Simplicial complexes reconstruction and generalisation of 3d lidar data in urban scenes

Stéphane Guinard

UNIVERSITÉ PARIS-EST

THÈSE

PRÉSENTÉE POUR OBTENIR LE GRADE DE :

Docteur en Philosophie de l'Université Paris-Est

MSTIC (ED532)
SPÉCIALITÉ: SCIENCES ET TECHNOLOGIES DE L'INFORMATION GÉOGRAPHIQUE

RECONSTRUCTION ET GÉNÉRALISATION DE COMPLEXES SIMPLICIAUX À PARTIR DE SCANS LIDAR DE SCÈNES URBAINES

SIMPLICIAL COMPLEXES RECONSTRUCTION AND GENERALISATION OF 3D LIDAR DATA IN URBAN SCENES

Stéphane A. GUINARD

Directeur de thèse : **Bruno VALLET**
Encadrants : **Laurent CARAFFA** & **Loïc LANDRIEU**

SOUTENUE PUBLIQUEMENT LE 19 JUIN 2020

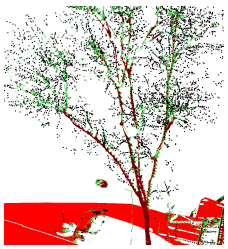
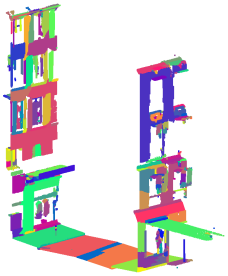
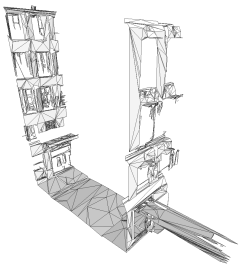
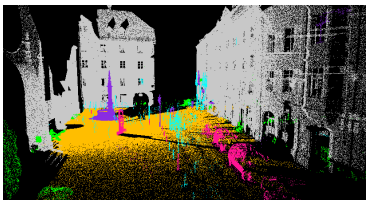
Composition du Jury

Paul CHECCHIN	Université Clermont Auvergne	Rapporteur
Roderik LINDENBERGH	Delft University of Technology	Rapporteur
François GOULETTE	Mines ParisTech	Président du jury
Sylvie DANIEL	Université Laval	Examinateur
Pierre GRUSSENMEYER	INSA Strasbourg	Examinateur
Pooran MEMARI	École Polytechnique	Examinateur
Laurent CARAFFA	IGN	Encadrant
Bruno VALLET	IGN	Directeur de thèse

Simplicial Complexes Reconstruction and Generalisation of 3D LiDAR Data in Urban Scenes

Stéphane A. Guinard

Advisor: **Bruno Vallet**
Supervizors: **Laurent Caraffa** & **Loïc Landrieu**

Jury members

Paul Checchin	Université Clermont Auvergne	Reviewer
Roderik Lindenbergh	Delft University of Technology	Reviewer
François Goulette	Mines ParisTech	President
Sylvie Daniel	Université Laval	Examiner
Pierre Grussenmeyer	INSA Strasbourg	Examiner
Pooran Memari	École Polytechnique	Examiner
Laurent Caraffa	IGN	Supervizor
Bruno Vallet	IGN	Advisor

LaSTIG – ACTE,
Institut National de l'Information
Géographique et Forestière (IGN)

Colophon

This disseration was typesetted using the LaTeX typesetting system, originally developed by Leslie Lamport, based on TeX created by Donald Knuth. The bibliography has been managed and compiled with the BibTeX software developed by Oren Patashnik and Leslie Lamport. This document has been compiled with the pdfLaTeX software, originally developed by Hàn Thế Thành.

This thesis was done at the ACTE (ACquisition et TraitEment de données Image / Lidar / Radar) team of the LaSTIG laboratory in Saint-Mandé, France, with the financial support of the DGA.

The slides for the defense are available at: https://fr.overleaf.com/read/qdwnvjrydzsq.

The bibtex entry to cite this dissertation is:

```
@phdthesis{guinard2020simplicial,
    author = {{Guinard, St\'ephane}},
    title = {{Simplicial Complexes Reconstruction and
    Generalization from 3{D} {LiDAR} Data in Urban Scenes}},
    school = {{Universit\'e Paris Est}},
    year = {2020},
    month = {June}
}
```

For any questions or remarks, feel free to contact me by email at: st.guinard@gmail.com.

August 31, 2020
Vincennes, France

À mes grand-parents, Pierre et Juliette.

Le monde a une réalité.
C'est d'elle qu'il faut partir, non d'un modèle idéal qu'il s'agirait d'approcher au plus près.
Le monde est. Le monde est ce qu'il est.
Peu importe ce qu'il pourrait être, ce qu'il aurait pu devenir, ce qu'il sera si...

Alain Damasio

Abstract

Thanks to their ever improving resolution and accessibility, Light Detection And Ranging (LiDAR) sensors are increasingly used for mapping cities. Indeed, these sensors are able to efficiently capture high-density scans, which can then be used to produce geometrically detailed reconstructions of complex scenes. However, such reconstruction requires organizing the scan with a fitting data structure, such as point clouds or meshes. Point clouds provide such a representation in a compact way, but their discrete nature prevents some applications such as visualization or simulation. Meshes allow for a continuous representation of surfaces, but are not well suited for representing complex objects, whose level of detail can exceed the resolution. To address these limitations, we propose to reconstruct a continuous geometry of the acquisition where sufficient geometric information is available only. This leads us to create a reconstruction mixing triangles, edges and points. We call such collection of objects a simplicial complex.

In this thesis, we study the creation of geometrically detailed 3-dimensional (3D) models of urban scenes, based on simplicial complexes. We show that simplicial complexes are a suitable alternative to such meshes. Indeed, they are fast to compute, and can be simplified while maintaining high geometric geometric fidelity with respect to the input scan. We argue that simplicial complexes convey valuable geometric information which can in turn be used for the semantization of 3D point clouds. We also think that they can serve as input for multi-scale reconstructions of urban scenes.

We first present an efficient algorithm for computing simplicial complexes from LiDAR scans of urban scenes. Since the reconstructed simplicial complexes can be very large, they can be difficult to process on a standard computer. To handle this challenge, we investigate different approaches for their spatial generalization by approximating large and geometrically simple areas with elementary primitives. To this end, we propose a new algorithm to compute piecewise-planar approximations of 3D point clouds, based on a global optimization approach. Next, we propose two different applications of simplicial complexes. The first one is a polygonalization method improving the creation of light yet geometrically accurate 3D models. The second one is a weakly-supervised classification method using 3D local and global descriptors.

Keywords: Point Cloud, Surface Reconstruction, Classification, Segmentation, Simplicial Complexes, LiDAR.

Résumé

Grâce à leur résolution et à leur accessibilité toujours meilleures, les capteurs LiDAR sont de plus en plus utilisés pour cartographier les villes. En effet, ces capteurs sont capables de réaliser efficacement des acquisitions à haut résolution, qui peuvent ensuite être utilisées pour produire des reconstructions géométriquement détaillées de scènes complexes. Cependant, une telle reconstruction nécessite d'organiser les données avec une structure de données adaptée, comme des nuages de points ou des maillages. Les nuages de points fournissent une représentation compacte des données, mais leur nature discrète empêche certaines applications telles que la visualisation ou la simulation. Les maillages permettent une représentation continue des surfaces, mais ne sont pas bien adaptés à la représentation d'objets complexes, dont le niveau de détail peut dépasser la résolution de l'acquisition. Pour remédier à ces limitations, nous proposons de reconstruire une géométrie continue uniquement lorsque suffisamment d'informations géométriques sont disponibles. Cela nous amène à créer une reconstruction mêlant triangles, arêtes et points. Nous appelons une telle collection d'objets un complexe simplicial.

Dans cette thèse, nous étudions la création de modèles 3D de scènes urbaines géométriquement détaillés, basés sur des complexes simpliciaux. Nous montrons que les complexes simpliciaux sont une alternative appropriée aux maillages. En effet, ils sont rapides à calculer et peuvent être simplifiés tout en conservant une grande fidélité géométrique par rapport aux données d'entrée. Nous soutenons que les complexes simples transmettent de précieuses informations géométriques qui peuvent à leur tour être utilisées pour la sémantisation des nuages de points 3D. Nous pensons également qu'ils peuvent servir de base pour des reconstructions multi-échelles de scènes urbaines.

Nous présentons d'abord un algorithme efficace pour le calcul de complexes simpliciaux à partir d'acquisitions LiDAR de scènes urbaines. Comme les complexes simpliciaux reconstruits peuvent être très lourds, ils peuvent être difficiles à traiter sur un ordinateur standard. Pour relever ce défi, nous étudions différentes approches pour les généraliser spatialement, en approximant de grandes zones géométriquement simples par des primitives simples. À cette fin, nous proposons un nouvel algorithme pour calculer des approximations planaires par morceaux de nuages de points 3D, basé sur une approche d'optimisation globale. Ensuite, nous proposons deux applications différentes des complexes simpliciaux. La première est une méthode de polygonalisation améliorant la création de modèles 3D légers mais géométriquement précis. La seconde est une méthode de classification faiblement supervisée utilisant des descripteurs 3D locaux et globaux.

Mots clés : nuages de points, reconstruction de surface, classification, segmentation, complexes simpliciaux, LiDAR.

Acknowlegements

It's hard to believe that 3 years have passed and that my PhD. came to an end. However, as short as it might seem for some, I feel like I lived an amazing adventure, and I owe so much to so many people that I don't know who to acknowledge first.

I want to start by thanking the French DGA for the finantial support they provided for this work. Thanks to them I was able to work full time and stay focused on my research. I more particularly thank Marie-Véronique Serfaty, Odile Ousset and Gwladys Theuillon for their interest and the discussions we had. I hope the DGA will find this work interesting and helpful for future applications.

I deeply thank Bruno, for always being here for me, from even before the start of the thesis, up to the defense. He always found time to help, teach and encourage me. Thanks to him, I was able to learn and experiment on various topics. His vast experience, along with his desire to impart his knowledge made me eager to always give the best, and I'm genuinely happy to had the opportunity to work with him.

Also, this work would not have been possible without the without the support of Loïc and Laurent. They were fully complementary as advisors. Loïc was one of the most rigorous and passionated researcher I ever met. Thanks to him, I was able to work on abstract concepts such as non-convex optimization, and develop the ℓ_0-plane pursuit algorithm, which really outperforms what I thought I was able to do before starting the thesis. Laurent was always ready to spend as much time as needed (too much I fear...) to explain abstract concepts, help me debugging or simply motivating me. I feel that all our talks (technical or not) early in the morning, when nobody else was at the lab., or when we were both in Barcelona working for ICGC, have been driving me for the whole thesis.

I want to sincerely thank François Goulette, for providing sound comments on this work, as a member of my PhD committee, and for accepting the presidency of my jury. I want to acknowledge Roderik Lindenbergh and Paul Checchin, for accepting to review this manuscript and providing valuable comments on this work. Also, I sincerely thank Sylvie Daniel, Pierre Grussenmeyer and Pooran Memari for taking part in the defence and the very interesting discussion we had. I also want to acknowledge Pascal Monasse for his important feedback, as a member of my PhD committee.

During this work, I had the opportunity to work with the amazing team of the LaSTIG lab. I especially want to express my gratitude to Oussama, Mohamed, Teng,

Raphael and Amine, which, more than colleagues, are really friends for me. They were always here, either for technical talks, lighter and funnier discussions, or encouraging me, which gave me the will to finish this work and the manuscript. Also, I want to thank Lâmân, Imran, Arnaud the druid, Mattia, Alexandre, Paul, Ali and Laurence for all the laughs during the (way too long) coffee breaks. David, Marie-Claude and Alain have to be acknowledged as well, as they are carrying the whole research team on their own shoulders. And simply for the friendly atmosphere and all the support that they provided me, I thank Manchun, Sébastien G., Luc, Olivier, Nathan, Yannis, Charly, Imane, Yilin, David, Laurent S., Clément, Vivien, Anatole, Kamel, Neelanjan, Quoc, JPP, Émile, Martyna, Margot, Valérie, Mathieu, Ewelina, Marc, Bahman, Sidonie, Arnaud the bordelais, Quy-Thy, Qasem, Evelyn, Nathalie, Bertrand, Mickaël, Tristan, Clément Junior, Zoumana, Pierre the bordelais, Pierre the lorrain, Dimitri, Guillaume, Sébastien M., Bénédicte, Ana-Maria, Julien, Marion, Giang and Nicolas.

I also want to thank the people at ENSG that helped me managing and giving courses: Clément Delgrange, Benoit Costes, Victor Coindet, Paul Bouquet and Sylvain Gonnet.

During this thesis, I had the opportunity to work for with the ICGC photogrammetric team in Barcelona. They are amazing guys, and I want to thank Jordi, Fernando, Vicenç, Ramon, Carlos, Dolors, Oscar, Juan Fernando, Lydia, Ariadna and Daniele for their friendliness and their technical support.

In the end, I want to thank my parents, my sisters, my in-laws and my nephews / niece for all their love and support during these 3 years (although some of them are less than 3 years old). I also direct special thanks to Pierre and Juliette, my grand-parents, whose abnegation and rigor inspired me, and to which this work is dedicated.

Last, my never-ending gratitude goes to Anne-Sophie, who, not only has been supporting me during all my studies, but also took care of me, especially during the hard times of writing this manuscript and preparing the defense.

Acronyms

2D 2-dimensional. 62, 63, 65, 73, 92, 94–96, 98, 103, 116, 117, 153, 190

3D 3-dimensional. xi, xxi, xxiv, xxv, xxxi, 60–65, 67–73, 77–79, 81–83, 87, 88, 92–96, 98, 113, 115–121, 123, 128, 129, 142, 146, 147, 150, 154, 155, 157, 160, 165, 168–172, 174–176, 190–193, 197

4D 4-dimensional. 70

ALS Aerial Laser Scanning. xxv, 70, 81, 83, 84, 86–89, 95, 116, 128–130, 142, 193

ASR Airport Surveillance Radar. 65, 66

BRDF Bidirectional Reflectance Distribution Function. 143

CDT Constrained Delaunay Triangulation. 160

CRF Conditional Random Field. 170, 171, 179, 180, 182, 186–188

DEM Digital Elevation Model. 83, 86, 88

DL Description Length. 157

DSM Digital Surface Model. 63, 157

DT Delaunay triangulation. 140, 141

DTM Digital Terrain Model. 86, 157

GPS Global Positionning System. xxv, 64, 77, 82, 83, 89

IGN Institut National de l'Information Géographique et Forestière. 83

IMU Inertial Measurement Unit. 64, 77, 82, 83, 89

ISAR Inverse Synthetic-Aperture Radar. 65

KNN k-nearest neighbors. xxv, 95

LASER Light Amplification by Stimulated Emission of Radiation. 64, 65, 67, 69, 71, 76–79, 81, 82, 85, 89

LiDAR Light Detection And Ranging. xi, xxiv, 60–65, 67–73, 76–89, 92–94, 98, 101, 102, 116, 119, 120, 142, 144, 157, 168, 170, 181, 190–193

LoD Level of Detail. 68, 70, 120, 150, 192, 193

MASER Microwave Amplification by Stimulated Emission of Radiation. 76, 77

MCMC Markov Chain Monte Carlo. 117

MDL Minimal Description Length. 157, 158, 160, 161, 165, 190

MLS Mobile Laser Scanning. xxiv, xxv, 69, 70, 77, 82–84, 87, 89, 92, 95, 97, 104, 116, 128, 130, 164

MRF Markov Random Field. 150, 170

NN Neural Networks. 63, 117, 170, 192

Radar Radio Detection and Ranging. xxiv, 60, 61, 63, 65–67

RANSAC RANdom SAmple Consensus. xxvii, 117–119, 125, 126, 135, 138–140, 143, 144

rjMCMC reversible jump Markov Chain Monte Carlo. 117

SAR Synthetic-Aperture Radar. xxiv, 65, 66

SC Simplicial complexes. 140, 141

SVM Support Vector Machine. 117, 170, 176

TLS Terrestrial Laser Scanning. xxiv, xxv, 62, 70, 82–84, 89, 92, 97, 116, 128–130, 142

UAV Unmanned aerial vehicle. xxv, 84–86, 193

VSA Variational Shape Approximation. 150, 160, 161, 164

Contents

List of Figures

List of Tables

Résumé étendu

Contents

0.1 Introduction

Le but de la télédétection est d'extraire des informations de scènes, sans contact physique, en utilisant des capteurs dédiés. Cela peut se faire par propagation d'ondes, soit dans le spectre de la lumière visible avec des caméras ou des LiDARs, soit à d'autres longueurs d'onde, comme pour les capteurs RaDAR. Cependant, ces capteurs fournissent des informations spatiales incomplètes, en particulier dans le cas des scènes urbaines, qui sont géométriquement complexes. Cela peut être dû à des occlusions, ou à la configuration géométrique de la scène. Par conséquent, un défi majeur de la communauté de la télédétection est de traiter les données manquantes ou incomplètes. La façon la plus simple de remédier à ces lacunes est d'effectuer plusieurs acquisitions de la même scène, avec des capteurs différents ou à des moments différents. Cependant, cela peut s'avérer coûteux, tant en terme de temps que d'argent. En outre, toutes les acquisitions doivent alors être enregistrées conjointement.

Une des applications de la recherche en télédétection est de fournir des modèles 3D de scènes réelles. Cela est particulièrement important dans les zones urbaines, dont la croissance rapide doit être surveillée par les acteurs étatiques, mais aussi pour effectuer des simulations de risques liés à l'environnement. Afin de réaliser cette tâche de modélisation, plusieurs méthodes ont été développées pour reconstruire des villes en 3D, à des échelles variant d'un seul bâtiment à une ville entière et avec des modélisations finales de qualité différente. Cependant, la reconstruction de scènes urbaines reste un défi, car elles sont composées de nombreux objets de formes et de tailles variées. Ces objets vont des routes, qui ont une grande étendue spatiale, aux poteaux, qui sont fins et ont une empreinte plus petite. De plus, lorsqu'il s'agit de grandes scènes, les méthodes de reconstruction 3D doivent souvent faire face à des données échantillonnées de manière irrégulière et à de nombreux petits détails géométriques. Les algorithmes de reconstruction doivent tenir compte de ces deux contraintes, ce qui complique la tâche de reconstruction. De plus, en raison des limites de la mémoire et des calculs, les ordinateurs standards actuels ne peuvent pas traiter des reconstructions 3D complètes de villes entières. Cependant, les scènes urbaines présentent généralement une certaine régularité géométrique pour des objets géométriquement simples, tels que les routes ou les façades. Cela a conduit les chercheurs à étudier les possibilités de simplifier ces modèles 3D afin de diminuer leur empreinte mémoire, tout en préservant la qualité géométrique du modèle. Toutefois, ce processus de simplification s'accompagne généralement d'une perte de la qualité géométrique de la reconstruction d'entrée.

Dans cette thèse, nous étudions la reconstruction de scènes urbaines, en restant le plus proche possible de la géométrie de l'acquisition, sans toutefois combiner plusieurs capteurs ou acquisitions. Nous soutenons que l'utilisation de toutes les informations disponibles d'un capteur LiDAR est suffisante pour une reconstruction géométriquement précise d'une scène. Nous souhaitons que notre approche reste aussi simple que possible, de sorte que nous n'inférions aucune information supplémentaire par rapport l'acquisition (pas de nouveaux points ou de points dupliqués) et nous voulons atteindre un compromis satisfaisant entre la taille de la mémoire du modèle 3D et sa fidélité géométrique à la scène réelle.

Nous commençons par présenter les différents types de capteurs utilisés pour l'acquisition de scènes urbaines et nous expliquons pourquoi nous avons choisi de nous concentrer sur les données LiDAR. Ensuite, nous expliquons certains choix faits pour concevoir notre méthode de reconstruction. Après cela, nous présentons quelques spécifications de nos reconstructions, en termes de remplissage des trous et de qualité géométrique. Enfin, nous présentons l'organisation de cette thèse.

0.1.1 Extraction de géométrie de scènes urbaines

L'extraction de géométrie dans des scènes urbaines se fait essentiellement à partir de trois types de capteurs différents : les appareils photos, les LiDARs et les RaDARs. Les appareils photos sont des capteurs passifs, peu chers et capables d'acquérir des scènes urbaines avec précision. Cependant, ils sont sensibles aux variations d'environnement, comme les changements de luminosité. De plus, des étapes de pré-traitement sont nécessaires afin de pouvoir utiliser les images pour reconstruire une scène en 3D.

Les capteurs LiDAR sont des capteurs actifs, contenant un laser émettant de la lumière à intervalles réguliers et enregistrant la lumière renvoyée par la scène. Ces capteurs sont onéreux mais sont capables d'acquérir des scènes urbaines avec une grande précision, tout en s'affranchissant des problèmes rencontrés par les appareils photos.

Le troisième type de capteur existant est le RaDAR. Ces capteurs sont aussi des capteurs actifs. Cependant, ils sont en général utilisés pour cartographier de très grandes scènes et produisent ainsi des acquisitions moins denses que les deux premiers capteurs. Pour toutes ces raisons, nous décidons de focaliser notre étude sur les données LiDAR.

0.1.2 Niveau de détail

Nous appelons *reconstruction 3D* la création d'une continuité géométrique dans une scène d'intérêt. Une reconstruction 3D doit respecter deux critères :

— compacité : la reconstruction doit être légère en termes de stockage informatique.
— fidélité géométrique : l'écart entre la surface reconstruite et la scène réelle doit être le plus petit possible.

Nous proposons aussi de caractériser le niveau de détail de nos reconstructions par la taille du plus petit objet géométrique reconstruit. Nous souhaitons utiliser la qualité géométrique de l'acquisition afin de reconstruire tous les objets possibles, tels <ue les arbres ou les poteaux.

0.1.3 Reconstruction 3D de données LiDAR

Le procédé d'acquisition LiDAR implique généralement la présence de trous dans les données. Ceux-ci sont dûs aux occlusions et à la grande variation de densité au sein même d'une acquisition LiDAR. Il s'agit d'une conséquence directe du théorême de Shannon-Nyquist [1].

Dans cette thèse, nous avons choisi de ne pas combler ces trous afin de ne pas ajouter trop d'informations qui pourraient significativement modifier la reconstruction finale. Cela signifie que nous ne pourrons donc pas créer une continuité géométrique dans toute la scène. Contrairement aux méthodes classiques de reconstruction 3D qui produisent des maillages, nous proposons de reconstruire des ensembles de points, arêtes et segments en fonction de la complexité géométrique locale de la scène. L'objet ainsi reconstruit s'appelle un *complexe simplicial*.

0.1.4 Organisation de la thèse

Dans cette thèse, nous étudions l'utilité des complexes simpliciaux pour représenter des scènes urbaines. Nous nous intéressons aussi aux possibilités d'utilisation desdits

1. Si la fréquence géométrique d'une scène est supérieure à la moitié de la fréquence d'acquisition, alors une partie de l'information est perdue.

complexes simpliciaux afin de généraliser et sémantiser des nuages de points.

Tout d'abord, nous présentons la composition et les principaux usages des capteurs LiDARs. Ensuite, nous proposons une nouvelle méthode pour la reconstruction de complexes simpliciaux à partir d'acquisitions LiDAR. Étant donné que nous travaillons sur des zones urbaines, composées majoritairement d'objets planaires, nous proposons donc un algorithme nommé ℓ_0-plane pursuit permettant de réaliser des simplifications planaires par morceaux des parties triangulées de nos complexes simpliciaux. Après cela, nous évaluons la qualité de reconstructions 3D basées sur les maillages simplifiés obtenus précédemment. Enfin, nous évaluons l'influence des complexes simpliciaux en tant que descripteurs pour la classification de nuages de points.

0.2 Capteurs et données LiDAR

0.2.1 Capteurs LiDAR

0.2.1.1 Composition

Les capteurs LiDAR sont des capteurs actifs : ils émettent de la lumière et détectent la quantité de lumière réfléchie vers le capteur. Afin d'émettre de la lumière, un capteur LiDAR utilise un laser. Cet outil est capable de produire de la lumière *cohérente*, ce qui signifie que tous les photons émis ainsi sont identiques (HAKEN, 1985). Cela permet d'obtenir un faisceau qui se diffracte peu, même sur de longues distances. La lumière émise est en général comprise entre 500 nm (vert) et 1550 nm (infrarouge). Nous appelons chaque émission de lumière par le laser un *pulse*. Chaque pulse dure de quelques nanosecondes à quelques microsecondes.

Le laser d'un capteur LiDAR pointe en direction d'un miroir. Ce miroir tourne sur un axe avec un ou deux degrés de liberté. Cela permet d'assurer un échantillonnage régulier et en trois dimensions de la scène.

L'autre composante essentielle d'un capteur LiDAR est le photodétecteur. En effet, il s'agit d'un appareil capable de générer un courant électrique lorsque stimulé par de la lumière. Cela permet de quantifier la quantité de lumière réfléchie et d'évaluer le temps écoulé entre l'émission et la réception. La figure 1 montre un exemple de capteur LiDAR.

Les capteurs LiDAR sont souvent équipés de capteurs complémentaires, afin de compléter l'acquisition ou de faciliter le recalage des données. Ainsi, les LiDARs mobiles, comme les MLS, sont équipés d'un GPS et d'une centrale inertielle, qui facilitent ce recalage. De plus, les capteurs LiDAR sont en général équipés d'un appareil photo, qui permet d'associer les deux moyens d'acquisition et facilite la texturisation du scan.

0.2.1.2 Création d'un nuage de points à partir d'un scan LiDAR

0.2.1.2.1 Cas d'écho unique Nous appelons *écho* le point créé à partir de la lumière réfléchie liée à une émission du laser. Afin de trouver la position dans l'espace de ce point, il nous suffit de connaître la position du laser au moment de l'émission, l'angle du miroir et la distance entre le laser et l'objet rencontré. Cette dernière est obtenue à partir de la vitesse de la lumière et du temps écoulé entre émission et réception de lumière. Ainsi, pour chaque émission de lumière, nous pouvons créer un point en 3D là où un objet physique a réfléchi la lumière.

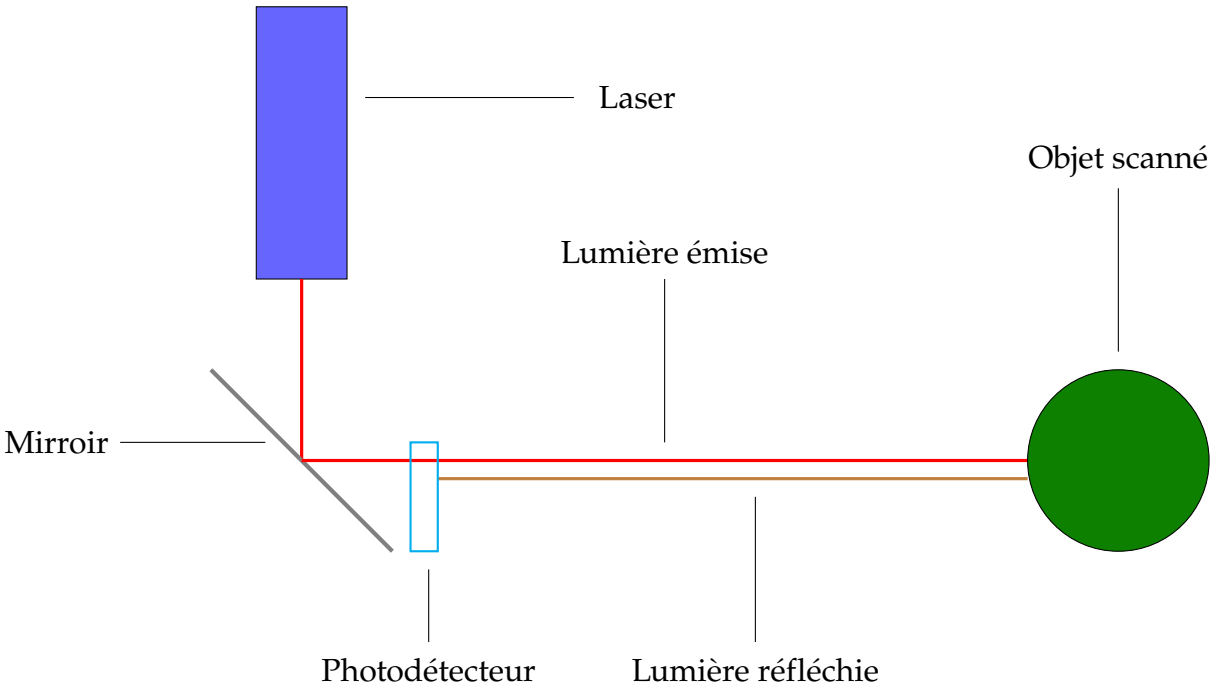


FIGURE 1 – Illustration d'un capteur LiDAR.

0.2.1.2.2 Cas d'échos multiples Cependant, la complexité géométrique des scènes urbaines fait que dans certains cas, seule une partie de la lumière est réfléchie au contact d'un objet et le reste continue de se propager. C'est notamment le cas lorsque le faisceau lumineux arrive au bord d'un objet, ou rencontre un objet qui transmet partiellement la lumière. Cela signifie que pour une seule émission de lumière, il peut y avoir plusieurs faisceaux réfléchis vers le LiDAR (correspondant à des objets différents). Dans ces cas-là, les LiDARs modernes sont capables de créer un point pour chaque faisceau réfléchi pour une unique émission de lumière.

0.2.2 Applications

Le domaine d'application historique du LiDAR est celui de la météorologie, notamment afin de suivre des objets tels les nuages. Ceci permet d'aider la cartographie des courants aériens et de prévoir des événements météorologiques, tels les tempêtes (SUOMI et al., 2017). Cela permet aussi d'estimer l'énergie éolienne (GOIT, SHIMADA et KOGAKI, 2019).

Les capteurs LiDARs sont aussi très utilisés pour la cartographie d'environnements forestiers. En effet, contrairement aux appareils photos, les acquisitions LiDAR peuvent traverser la végétation, ce qui permet de produire une acquisition plus rapide et complète des milieux forestiers. Cela permet entre autres d'estimer la quantité de biomasse (LUO et al., 2017 ; HOLM, NELSON et STÅHL, 2017) et de réaliser des MNT (BIGDELI, AMIRKOLAEE et PAHLAVANI, 2018) de ces milieux. Les LiDARs peuvent aussi être utilisés pour suivre les catastrophes naturelles, telles que les incendies de forêt (MCCARLEY et al., 2017 ; DELONG et al., 2018), les inondations (DE ALMEIDA et al., 2016), l'activité volcanique (BEHNCKE et al., 2016) et les tremblements de terre (ISHIMURA et al., 2019). À plus grance échelle, les capteurs LiDAR permettent aussi de suivre les mouvements des glaciers (PUTKINEN et al., 2017) et des plaques tectoniques (MEIGS, 2013).

Le fait de produire rapidement des acquisitions géométriquement détaillées et de pouvoir passer à travers la végétation rend les capteurs LiDARs particulièrement utiles

pour le suivi et l'entretien du patrimoine culturel. En effet, ces capteurs ont prouvé leur efficacité pour retrouver d'anciennes cités et routes aujourd'hui recouvertes par la végétation (RODRÍGUEZ-GONZÁLVEZ et al., 2017 ; INOMATA et al., 2018).

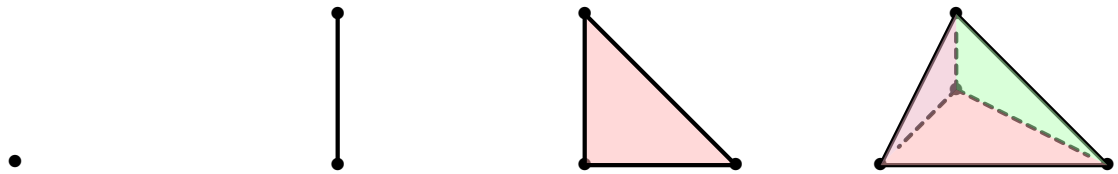
Enfin, les capteurs LiDAR sont tout particulièrement utiles pour scanner et suivre l'évolution de villes entières (WANG et al., 2019). C'est ainsi qu'a vu le concept de *jumeau numérique* : le fait de pouvoir représenter, suivre l'évolution et réaliser des simulations à l'échelle d'une ville entière est tout particulièrement utile dans le domaine de l'urbanisme (LIU et al., 2017 ; GOODING, CROOK et TOMLIN, 2015). De même, en milieu urbain, les capteurs LiDAR permettent d'aider à la conduite des voitures autonomes, en permettant une analyse géométrique précise et rapide de leur environnement (GHALLABI et al., 2018 ; MATTI, EKENEL et THIRAN, 2017).

0.3 Reconstruction de complexes simpliciaux

La première contribution technique de cette thèse est la mise en place d'une nouvelle méthode pour la reconstruction de complexes simpliciaux à partir d'acquisitions LiDAR de scènes urbaines.

0.3.1 Complexes simpliciaux

Pour $i \in [0, \dots, n]$, nous appelons i-simplexe de $\mathbb{R}^n$ l'enveloppe convexe de $i+1$ points indépendants de $\mathbb{R}^n$. Une face d'un i-simplexe de $\mathbb{R}^n$ est définie par l'enveloppe convexe de j points indépendants ($j \in [1, \dots, i]$).Des exemples de simplexes sont affichés dans la figure 2. Dans cette thèse, nous nous intéressons à la reconstruction de la surface visible de la scène et non pas à la reconstruction entière de la scène. Cela signifie que nous nous concentrons uniquement sur la reconstruction de $0-2$-simplexes (points, arêtes et triangles).



(a) Simplexe de dimension 0.

(b) Simplexe de dimension 1

(c) Simplexe de dimension 2

(d) Simplexe de dimension 3

FIGURE 2 – Exemples de simplexes en faible dimension.

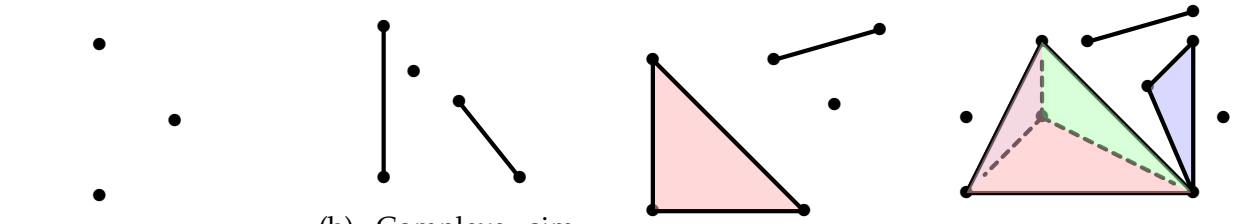
Pour $i \in [0, \dots, n]$, nous définissons un i-complexe simplicial de $\mathbb{R}^n$ comme un ensemble de simplexes de dimension inférieure à i. Tous les simplexes composant un complexe simplicial n'ont par forcément la même dimension Un i-complexe simplicial doit respecter les conditions suivantes :

— toutes les faces d'un i-complexe simplicial doivent lui appartenir.

— l'intersection de deux simplexes d'un i-complexe simplicial doit soit être une face partagée par ces deux simplexes, soit $\varnothing$.

La dimension d'un complexe simplicial est la dimension maximale des simplexes lui appartenant. Des exemples de complexes simpliciaux sont présentés dans la figure 3. Remarquons qu'un simplexe est aussi un complexe simplicial. Dans cette thèse,

nous nous intéressons uniquement à des complexes simpliciaux composés de $0-2$-simplexes.



(a) Complexe simplicial de dimension 0.

(b) Complexe simplicial de dimension 1.

(c) Complexe simplicial de dimension 2.

(d) Complexe simplicial de dimension 3.

FIGURE 3 – Exemples de complexes simpliciaux dans des espaces à faible dimension.

0.3.2 Reconstruction de complexes simpliciaux à partir de scans LiDAR 3D

0.3.2.1 Topologie capteur

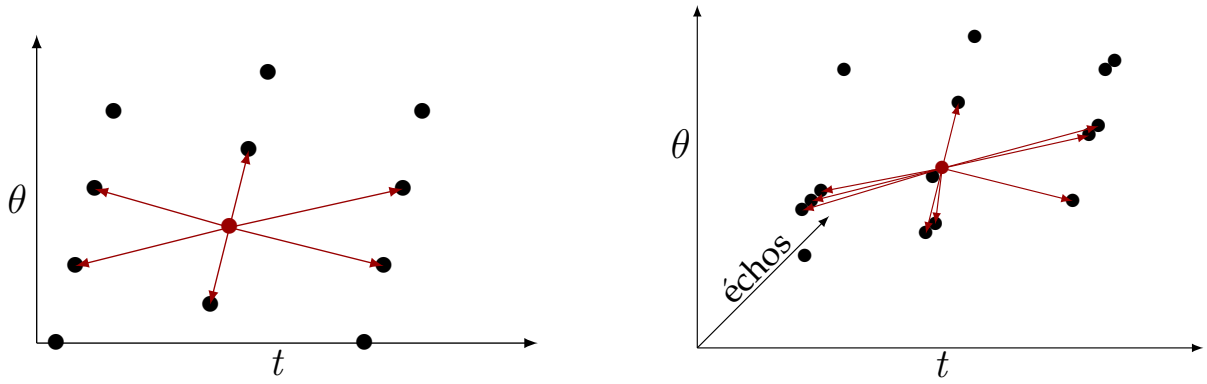
Un des premiers problèmes qui apparaît lorsque l'on veut traiter des données LiDAR est leur absence de structure interne : contrairement à une image, pour laquelle les pixels sont reliés entre eux suivant un 4- ou 8-voisinage (TOFFOLI et MARGOLUS, 1987, Section 7.2), il n'existe pas d'équivalent pour les nuages de points LiDAR. La plupart des travaux de l'état de l'art utilisent les k-plus proches voisins (WEINMANN et al., 2015b) ou une triangulation de Delaunay (BOISSONAT, 1984 ; ROMANONI et al., 2016). Cependant, ces méthodes ne sont pas capables de s'adapter aux variations de densité dans le scan et ont tendance à relier entre eux des points qui n'appartiennent pas aux mêmes objets dans la réalité (comme c'est le cas entre un poteau et la route, ou entre les différentes branches d'un arbre).

Nous remarquons que les capteurs LiDAR possèdent une topologie propre. En effet, chaque impulsion du laser peut être représentée dans un espace en deux dimensions, en fonction de la direction de la lumière émise et de l'instant d'émission. Ainsi, nous proposons d'associer à chaque impulsion, ses six voisins en topologie capteur. Cependant, comme évoqué précédemment, les capteurs LiDAR modernes sont capables d'enregistrer plusieurs échos pour une unique impulsion. C'est pourquoi nous décidons d'associer à chaque point / écho, tous les échos des impulsions voisines en topologie capteur. Ceci est illustré sur la figure 4. Notons qu'il existe peu de travaux dans la littérature qui utilisent la topologie capteur (XIAO, VALLET et PAPARODITIS, 2013 ; VALLET et al., 2015) et ceci vient essentiellement du fait que l'information brute venant du capteur est souvent perdue.

0.3.3 Reconstruction d'arêtes

Étant donné que les nuages de points peuvent être vus comme un complexe simplicial, composé uniquement de 0-simplexes, nous proposons d'ajouter en premier des 1-simplexes (arêtes), afin de reconnecter les points entre eux. Ensuite, nous ajoutons les 2-simplexes (triangles), à partir des arêtes créées. Ceci est illustré sur la figure 5.

Nous décidons de traiter chaque paire de points séparément. L'objectif est de déterminer dans quels cas une arête doit être créée entre deux points adjacents. Pour cela,



(a) Représentation du 6-voisinage induit par la topologie capteur pour les émissions lumineuses du laser. Chaque point correspond à une impulsion.

(b) Représentation de la topologie capteur dans le cas où chaque impulsion du laser peut amener à plusieurs faisceaux réfléchis. Chaque point représente un écho.

FIGURE 4 – Réprésentation de la topologie capteur et du voisinage qu'elle induit pour les émissions lumineuses (4a) et pour les échos reçus (4b).

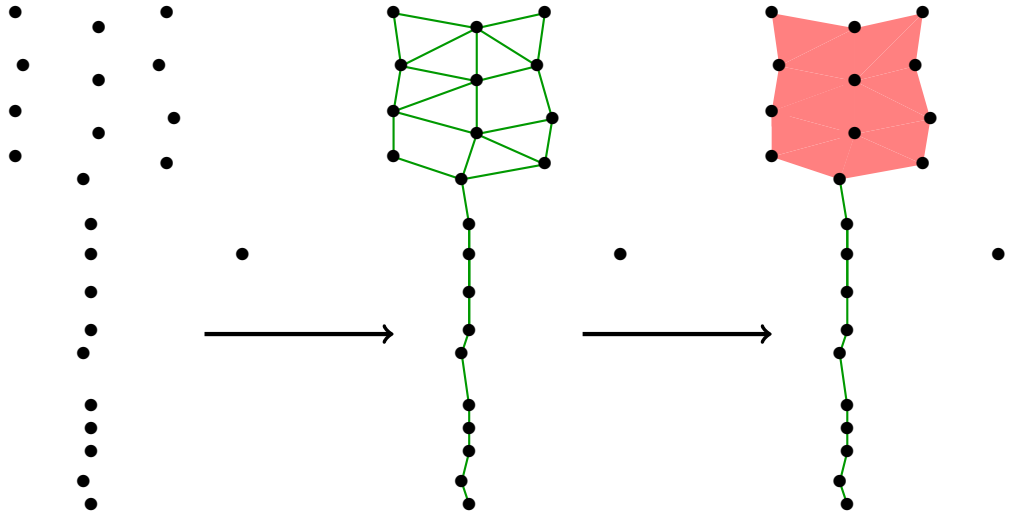
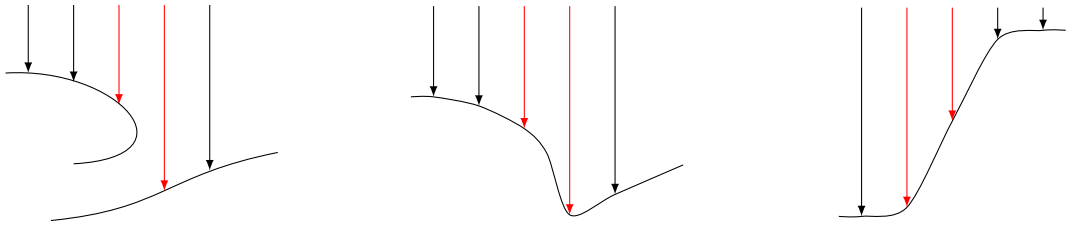


FIGURE 5 – À partir d'un ensemble de points (à gauche), nous ajoutons des arêtes entre chaque paire de points appartenant à un même objet dans la scène acquise. Ensuite nous ajoutons des triangles à partir des arêtes créées. Cela nous permet d'obtenir un complexe simplicial adapté à la géométrie locale de la scène. La scène représente un panneau de signalisation. Les points sont affichés en noir, les arêtes en vert et les triangles en rouge.

nous nous basons sur la différence de profondeur entre les deux points, par rapport à la position du capteur. Nous prenons aussi en compte la colinéarité de triplets de points adjacents. Ainsi, nous distinguons trois situations différentes représentées sur la figure 6 :

— le premier cas (figure 6a représente le cas où deux échos adjacents ont une grande différence de profondeur. Dans ce cas, nous considérons qu'ils n'appartiennent pas au même objet et les laissons séparés.

— le second cas (figure 6b représente la même acquisition que précédemment, mais dans ce cas, les deux échos font partie du même objet. Ici, la complexité géomé-



(a) Séparation (b) Cas limite (c) Non-séparation

FIGURE 6 – Représentation des trois différents cas abordés pour la reconstruction d'arêtes. Les échos considérés sont affichés en rouge.

trique de la scène est supérieure à la densité de l'acquisition et nous ne pouvons pas différencier ce cas du précédent, il est donc traité de la même manière.

— le dernier cas (figure 6c représente un cas ou trois échos ou plus sont presque alignés. Dans ce cas, nous pensons qu'ils appartiennent au même objet et souhaitons les reconnecter.

Afin d'encoder ces choix, nous mettons en place les deux critères suivants :

— C_0 régularité : nous empêchons la création d'arêtes entre deux échos avec une différence de profondeur trop importante.

— C_1 régularité : nous favorisons la création d'arêtes lorsque trois échos ou plus sont alignés.

Soient e_1 et e_2, deux impulsions du laser. Soient E_p et E_{p+1}, deux échos liés respectivement aux impulsions e_1 et e_2. Nous calculons les C_0 et C_1 régularités de la manière suivante :

$$C_0(p, e_1, e_2) = un - \vec{e_p}(e_1, e_2) \cdot \vec{l_p} \,, \tag{1}$$

avec $\vec{e_p}(e_1, e_2) = \frac{\overrightarrow{E_p^{e_1} E_{p+1}^{e_2}}}{||\overrightarrow{E_p^{e_1} E_{p+1}^{e_2}}||}$ et $\vec{l_p}$ la direction de l'impulsion laser. Remarquons que C_0 est proche de 0 pour des surfaces orthogonales et proche de 1 pour des surfaces parallèles à la direction de l'impulsion laser.

La régularité C_1 est calculée comme suit :

$$C_1(p, e_1, e_2) = min_{e=1}^{N_{p-1}} |1 - \vec{e}_{p-1}(e, e_1) \cdot \vec{e}_p(e_1, e_2)| \cdot min_{e=1}^{N_{p+2}} |1 - \vec{e}_p(e_1, e_2) \cdot \vec{e}_{p+1}(e_2, e)| \,, \tag{2}$$

avec N_p le nombre de pulses par ligne d'acquisitions [2] Nous décidons de créer une arête si et seulement si :

$$\begin{cases} C_0 & < \alpha_m \,, \\ & or \\ C_1 & < \frac{\lambda \cdot \alpha_m \cdot C_0}{\alpha_m - C_0} \,, \end{cases} \tag{3}$$

avec λ un paramètre servant de compromis entre les deux régularités.

2. Une ligne d'acquisition correspond à l'ensemble des impulsions effectuées dans le cadre d'une rotation de 2π radians du miroir.

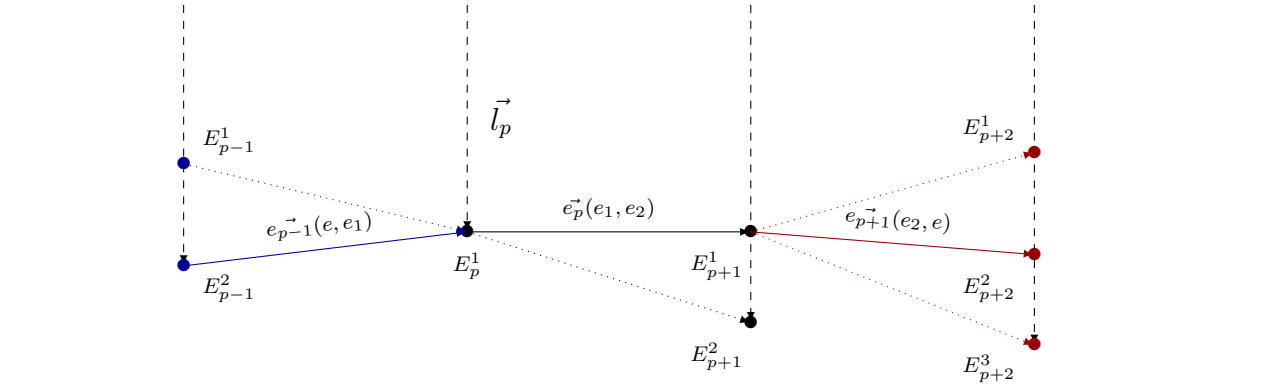


FIGURE 7 – Représentation des C_0 et C_1 régularités. Pour les échos bleus et rouges, nous sélectionnons les échos les plus proches de la ligne portée par $\vec{e_p}(e_1, e_2)$. Les échos ainsi sélectionnés sont utilisés pour calculer les deux régularités.

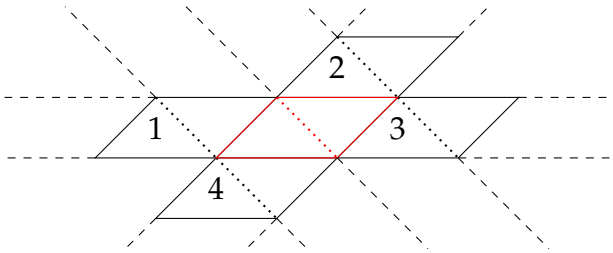


FIGURE 8 – Représentation d'une wedge (en rouge) et de ses wedges adjacentes. Les deux triangles formant une wedge sont séparés par une ligne en pointillé. Les lignes en tiret représentent les trois directions qui apparaissent dans le voisinage en topologie capteur. Les wedges adjacententes à la wedge rouge sont numérotées de un à 4.

0.3.4 Reconstruction de triangles

Maintenant que nous avons défini une méthode pour reconstruire des arêtes à partir d'ensembles de points et en utilisant la topologie capteur, nous devons nous intéresser à la création de triangles à partir de nos ensembles de points et d'arêtes.

La méthode la plus intuitive pour recréer des triangles est de considérer tous les triplets d'arêtes connectées entre elles et de reconstruire un triangle entre les trois échos qu'elles connectent. Cependant, avec cette méthode, l'absence d'une seule arête peut empêcher la création de deux triangles.

Nous décidons donc de procéder à la création des triangles dans les zones planaires du scan, indépendamment du fait que toutes les arêtes possibles aient été reconstruites ou non. Nous proposons d'étendre la notion de C_1 régularité pour les triangles. Ainsi, nous considérons non pas des triplets d'arêtes connectées, mais des quadruplets d'arêtes connectées, que nous appelons *wedge*. Lorsqu'une wedge est coplanaire avec au moins deux autres wedges, qui lui sont adjacentes, dans deux directions différentes, nous décidons de reconstruire les deux triangles formant la wedge. Ceci est illustré sur la figure 8.

0.3.5 Expériences

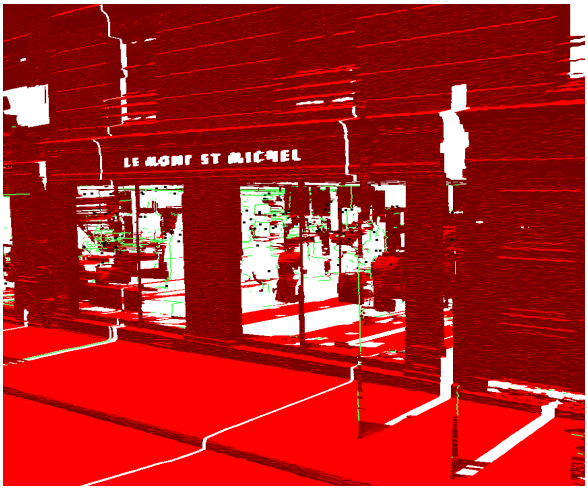
Les expériences réalisées pour évaluer cette contribution se basent sur une acquisition du véhicule de cartographie mobile Stéréopolis (PAPARODITIS et al., 2012). L'acquisition a eu lieu dans Paris. Nous avons comparé trois méthodes de reconstruction de complexes simpliciaux :

— Méthode naïve : dans cette méthode, toutes les arêtes reliant des points adjacents en topologie capteur dont la longueur est inférieure à un seuil sont reconstruites. Les triangles correspondent à des triplets d'arêtes connectés.

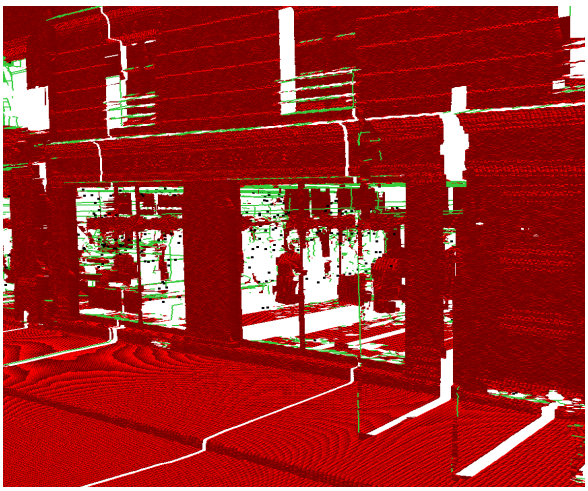
— Méthode avec arêtes filtrées : les arêtes sont créées à partir de la méthode décrite dans la section 0.3.3. Les triangles correspondent à des triplets d'arêtes connectés.

— Méthode avec triangles filtrés : les arêtes sont créées à partir de la méthode décrite dans la section 0.3.3. Les triangles sont créés à partir de notre approche basée sur des wedges.

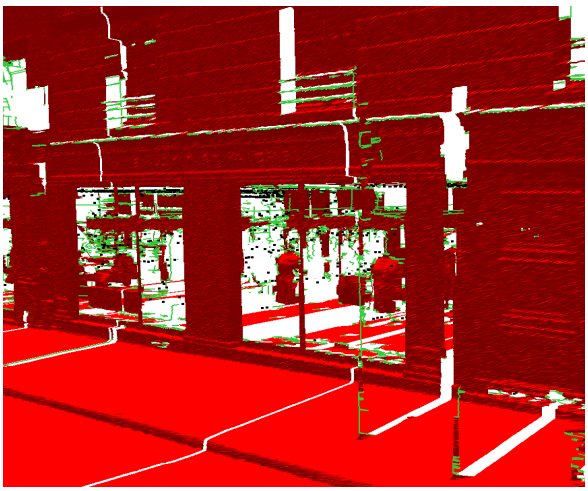
(a) Image de la scène dans GOOGLE (*View of the Mabillon street next to the corner of Lobineau street (Paris)*)



(b) Méthode naïve



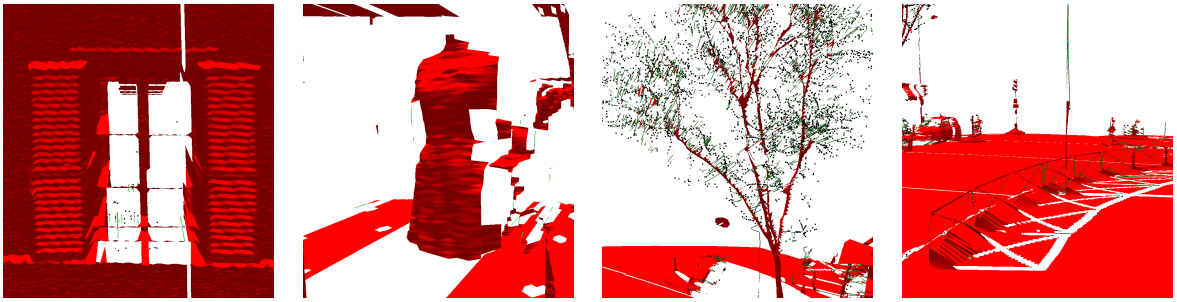
(c) Méthode avec filtrage d'arêtes



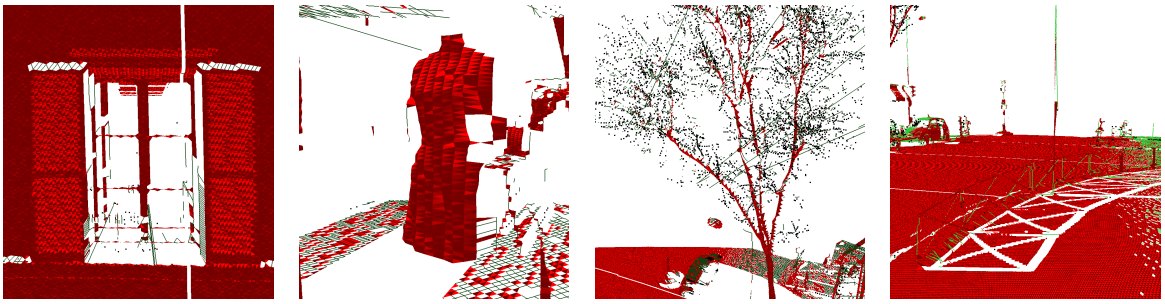
(d) Méthode avec filtrage de triangles

FIGURE 9 – Résultats sur une scène urbaine comprenant routes, façades, poteaux et piétons.

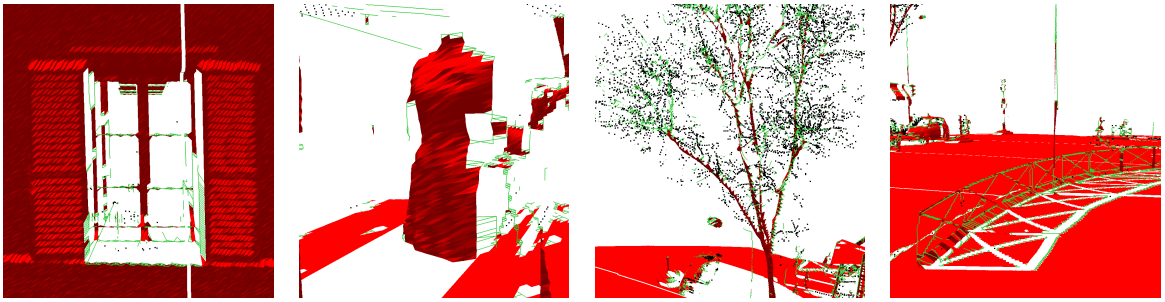
Nous observons que notre méthode, filtrant les arêtes et les triangles, nous permet



(a) Méthode naïve



(b) Méthode avec filtrage d'arêtes



(c) Méthode avec filtrage de triangles

FIGURE 10 – Comparaison de trois méthodes de reconstruction de complexes simpliciaux : la méthode naïve, la méthode avec filtrage d'arêtes et la méthode avec filtrage de triangles. De gauche à droite, les scènes représentent : une fenêtre, un mannequin, un arbre et une barrière sur un trottoir.

de reconstruire des complexes simpliciaux préservant la géométrie locale de l'acquisition. Ainsi, des objets souvent agrégés par des méthodes classiques de reconstruction sont séparés par notre méthode. C'est notamment le cas pour la base des poteaux, qui est ici séparée de la route. Nous remarquons aussi que notre méthode montre son efficacité pour la reconstruction d'arbres : le tronc et les branches principales sont composés de triangles, les branches plus fines sont visibles sous forme d'arêtes et les feuilles sont représentées comme un ensemble de points séparés les uns des autres.

Notons néanmoins que notre méthode est purement locale et que cela induit une reconstruction généralement bruitée. De plus, les zones géométriquement simples des scènes urbaines, telles que les routes ou certaines façades sont ici composées de plusieurs milliers de simplexes. Nous pensons que pour respecter l'objectif de compacité des méthodes de reconstruction, il nous faut simplifier nos reconstructions dans les régions géométriquement simples.

0.4 Généralisation planaire par morceaux de nuages de points

0.4.1 Simplification des complexes simpliciaux

Le processus de généralisation des complexes simpliciaux peut être divisé en trois sous problèmes :

— généralisation des points, basée sur un échantillonnage aléatoire ou un octree.

— généralisation des arêtes, basée sur des techniques bottom-up (Douglas-Peucker) ou top-down (edge collapse).

— généralisation des triangles.

Nous nous focalisons ici sur ce dernier problème, qui est le plus complexe des trois. Pour cela, nous nous intéressons aux méthodes de simplification de maillages. En particulier, nous observons que les scènes urbaines sont majoritairement composées d'objets planaires par morceaux. De plus, comme nous ne nous intéressons ici, qu'à la partie triangulée de nos complexes simpliciaux, nous pouvons supposer que les zones les plus géométriquement complexes de l'acquisition ne font pas partie du maillage. Nous proposons dans un premier temps de réaliser une segmentation planaire par morceaux de nos maillages, pour servir d'initialisation à un algorithme de décimation de maillages.

0.4.2 Représentations géométriques de scènes urbaines

La plupart des méthodes de l'état de l'art concernant la segmentation planaire par morceaux de maillages sont basées sur trois types d'algorithmes différents :

— RANSAC : cet algorithme permet d'extraire des primitives géométriques, telles que des plans, sphères ou cylindres,

— croissance de régions : cette méthode permet de préserver la géométrie locale des données mais est souvent plus lente et peine à passer à l'échelle,

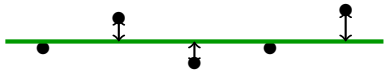
— découpe de graphes : cette méthode est rapide, mais peut être moins adaptée à la géométrie locale des données.

Ici, nous proposons de combiner RANSAC et une méthode de découpe de graphes, afin de réaliser une segmentation rapide de nos maillages, tout en préservant la géométrie locale.

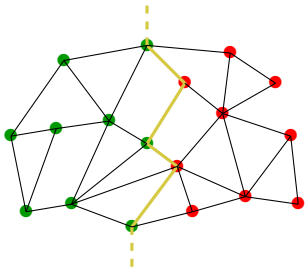
0.4.3 Approximations planaires par morceaux

0.4.3.1 Représentation de maillages sous forme de graphes

Nous représentons le maillage d'entrée sous forme d'un graphe $G = (V, E, w)$. V est l'ensemble des nœuds et chaque nœud correspond à un point 3D de notre maillage. E est l'ensemble des arêtes du graphe, ces arêtes se basent sur la structure du maillage. Enfin, chaque arête est pondérée en fonction de sa longueur par rapport à la longueur moyenne des arêtes du graphe, de telle sorte que les arêtes reliant deux points éloignés aient un plus faible poids que les arêtes reliant des points proches.



(a) Représentation du terme d'attache aux données dont le but est de minimiser la distance entre chaque point et le plan qui lui est associé (représenté ici par une ligne verte).



(b) Illustration du terme de régularisation pénalisant les régions avec des formes complexes. Les points verts et rouges appartiennent à deux régions différentes. La ligne jaune montre la frontière entre les deux régions.

FIGURE 11 – Représentation de notre modèle : nous voulons minimiser la distance entre chaque point et le plan qui lui est associé, tout en préservant des régions avec des formes simples.

0.4.3.2 Formulation du problème

Soit $\mathcal{P}$ l'ensemble des plans de $\mathbb{R}^3$. Soit $d(v, \pi)$ la distance euclidienne entre un nœud v et un plan $\pi \in \mathcal{P}$. Nous associons un plan π_v à chaque nœud v et notons $\Pi \in \mathcal{P}^V$ l'ensemble des plans ainsi définis. Ainsi, nous définissons une approximation de V par la projection de chaque nœud sur le plan qui lui est associé.

Nous proposons donc de minimiser l'énergie suivante $E : \mathcal{P}^V \to \mathbb{R}$ afin d'obtenir une approximation simple de V, tout en préservant la géométrie locale du maillage. Pour $\Pi \in \mathcal{P}^V$, l'énergie peut s'écrire de la façon suivante :

$$E(\Pi) = \underbrace{\sum_{v \in V} d\left(v, \pi_v\right)^2}_{Data\ term} + \mu \underbrace{\sum_{(u,v) \in E} w_{u,v} \left[\pi_u \neq \pi_v\right]}_{Regularizer}, \tag{4}$$

où $[\pi \neq \pi']$ est la fonction de $\mathcal{P}^2 \mapsto \{0, 1\}$ valant 0 quand π et π' sont identiques et 1 sinon. Le paramètre $\mu \in \mathbb{R}_+$ sert de compromis entre le terme d'attache aux données et la régularisation. Le premier terme de l'équation 4.2 correspond à l'éttache aux données, servant à minimiser la distance entre chaque point et son plan associé. Le second terme de l'équation 4 favorise les interfaces simples entre régions, favorisant ainsi les régions avec des formes simples. Le rôle de chaque terme est illustré sur la figure 11.

0.4.3.3 Algorithme ℓ_0-plane pursuit

Notre problème est similaire au *Generalized Minimal Partition Problem* présenté dans LANDRIEU et OBOZINSKI (2017). Ainsi, nous décidons de proposer un algorithme similaire à l'algorithme ℓ_0-cut pursuit introduit par LANDRIEU et OBOZINSKI (2017), afin de résoudre notre problème. Ainsi nous proposons l'algorithme ℓ_0-plane pursuit, qui diffère de ℓ_0-cut pursuit sur les points suivants :

— Initialisation : afin d'aider l'algorithme dans les premières itérations, nous proposons d'utiliser un RANSAC pour trouver les plans principaux de la scène.

— Raffinement : l'étape de raffinement de ℓ_0-cut pursuit est adaptée dans notre contexte : nous souhaitons trouver la partition binaire optimale $(Bi, A_i \backslash Bi)$ d'une

région $A_i \in \mathfrak{P}$. Cela revient à trouver :

$$\underset{B_i \subset A_i, (\pi,\pi') \in \mathcal{P}^2}{\arg\min} \sum_{v \in B_i} d(v,\pi) + \sum_{v \in A_i \setminus B_i} d(v,\pi') + \mu \sum_{(i,j) \in E_{B_i}} w_{i,j} \ , \tag{5}$$

avec $E_{B_i} = E \cap (B_i \times A_i \setminus B_i \cup A_i \setminus B_i \times B_i)$ l'ensemble des arêtes connectant B_i et $A_i \setminus B_i$.

— Fusion : l'étape complémentaire au raffinement est la fusion de régions adjacentes. Pour cela, nous calculons un plan commun aux deux régions et comparons l'énergie d'une région unifiée par rapport à celle des deux régions existantes avec leur frontière.

0.4.4 Expériences

0.4.4.1 Données, baseline et métrique

Nous avons testé l'algorithme ℓ_0-plane pursuit sur trois jeux de données différents, nommés :

— Paris : acquisition par un véhicule de cartographie mobile dans les rues de Paris,

— Chapelle : acquisition par LiDAR terrestre dans une chapelle,

— Barcelone : acquisition par LiDAR aérien au-dessus de la ville de Barcelone.

Les caractéristiques des trois acquisitions utilisées pour valider notre algorithme sont présentées dans le tableau 1.

Nom	Paris	Chapelle	Barcelone
Moyen d'acquisition	MLS	TLS	ALS
Capteur	RIEGL VQ-250	Leica ScanStation P40	Leica ALS50-II
Vitesse de rotation (Hz)	100	50	35
Multi-écho	< 8	Non	< 4
Densité (pts/m^2)	élevée : > 100	très élevée : > 1000	faible : quatre
Nombre de points	218,546	1,263,321	500,000
Nombre de triangles	418,254	7,313,760	872,372
Structure	Triangles de (GUINARD et VALLET, 2018a)	Triangles de (GUINARD et VALLET, 2018a)	Triangulatio de Delaunay

TABLE 1 – Caractéristiques des trois acquisitions utilisées pour tester l'algorithme ℓ_0-plane pursuit.

Nous avons comparé notre méthode à une méthode de segmentation planaire par morceaux basée sur une croissance de régions. Plus particulièrement, nous avons choisi la méthode introduite par COHEN-STEINER, ALLIEZ et DESBRUN (2004).

Nous avons choisi d'utiliser comme métrique la somme des distances euclidiennes au carré entre chaque point et le plan qui lui est associé. Cela nous permet de pénaliser fortement les points trop éloignés de leur plan respectif.

0.4.4.2 Résultats

Les résultats sur Paris sont présentés dans les figures 12 et 13. Nous remarquons que notre méthode est capable de s'adapter à la géométrie locale des données en produisant un faible nombre de régions pour les zones géométriquement simples et plus de régions pour les zones complexes de la scène. Ainsi, notre méthode produit des segmentations avec une faible erreur géométrique, mais est aussi plus rapide que les méthodes de type croissance de régions.

Nous observons aussi l'importance de l'étape de fusion de régions : elle permet diminuer le nombre de régions, tout en préservant la qualité géométrique de la segmentation. Ceci permet, à nombre de régions constant, d'obtenir de meilleures segmentations.

Des résultats plus détaillés sur ce jeu de données et sur les deux autres jeux de données sont présentés dans la section 4.3.3.

0.5 Polygonalisation

Maintenant que nous avons obtenu des segmentations planaires par morceaux de nos données, l'étape suivante pour la méthode de généralisation que nous mettons en place est de simplifier le maillage en un ensemble réduit de triangles, afin de retirer les données redondantes. Par exemple, les routes sont géométriquement simples et nous pensons qu'un bon algorithme de simplification de maillages devrait être capable de de représenter une telle zone de la scène par un faible nombre de triangles. Ce n'est pas le cas pour les parties maillées de nos complexes simpliciaux, mais nous comptons utiliser nos segmentations planaires par morceaux pour guider un algorithme de décimation de maillages, afin de décimer en priorité les zones géométriquement simples de la scène.

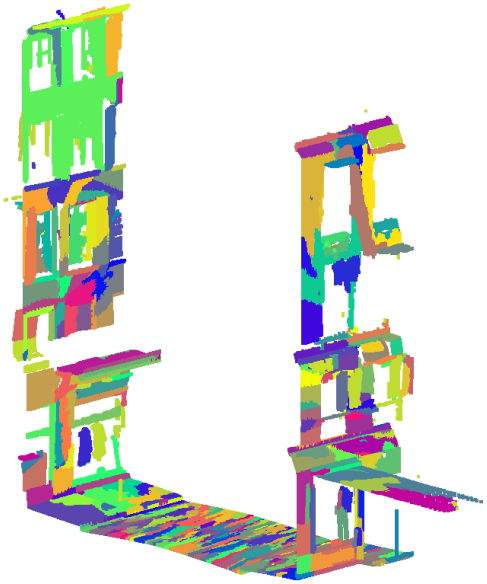
0.5.1 Projections planaires par morceaux de nuages de points 3D

Afin de pouvoir simplifier nos maillages, nous commençons par utiliser nos segmentations planaires par morceaux pour obtenir un maillage planaire par morceaux. La méthode la plus directe pour faire ceci est de projeter chaque point directement sur le plan qui lui est associé. Cependant, nous souhaitons préserver la qualité géométrique de l'acquisition dans les zones géométriquement complexes. Ainsi, si un point est trop éloigné de son projeté, nous choisissons de garder le point original. De plus, en ne prenant en compte que le plan associé à chaque point, des trous peuvent vite apparaître dans la reconstruction finale, au niveau des interfaces entre régions.

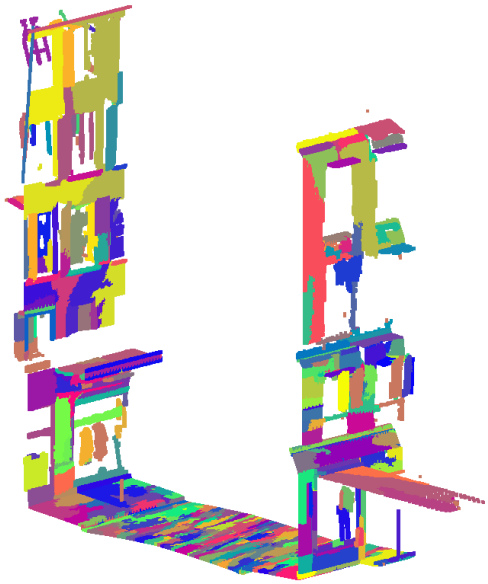
Pour résoudre ce problème, nous choisissons donc de considérer pour chaque point, l'ensemble de régions auquel appartient son voisinage. Ainsi, nous décidons de projeter un point, tant que la distance entre le point et son projeté est suffisamment faible, selon les règles suivantes :

i) si tous les points dans le voisinage du point considéré appartiennent à une même région : le point est projeté sur cette région,

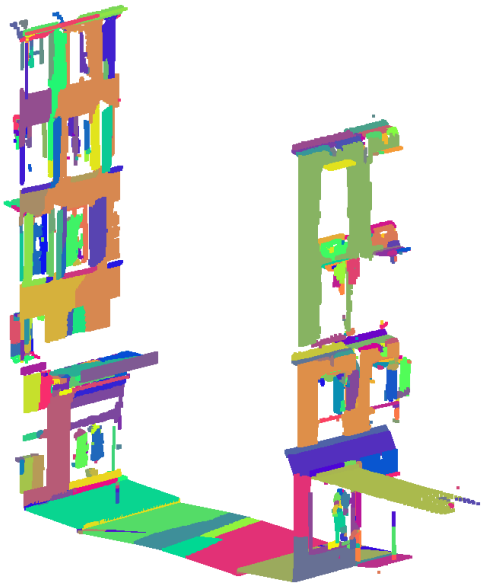
ii) si le voisinage direct d'un point est réparti entre deux régions différentes : le point est projeté sur l'intersection des deux plans. Si l'intersection n'est pas définie, ou si elle est trop loin du point d'origine, alors le point considéré est projeté sur le plan le plus proche.



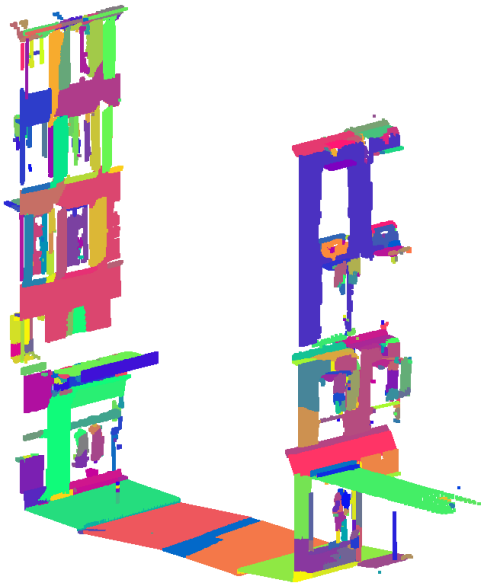
(a) Croissance de régions
492 régions
5 itérations
erreur : $18.3 \cdot 10^3$



(b) Croissance de régions
492 régions
15 itérations
erreur : $17.5 \cdot 10^3$



(c) ℓ_0-plane pursuit
pas de fusion de régions
514 régions
erreur : $1.6 \cdot 10^3$



(d) ℓ_0-plane pursuit
avec fusion de régions
492 régions
erreur : $1.6 \cdot 10^3$

FIGURE 12 – Comparaison de notre méthode de la baseline. Chaque couleur représente une région différente Les points sont projetés sur leur plan respectif. Notre méthode créée de grandes régions dans les zones géométriquement simples de la scène et de plus petites régions dans les zones géométriquement complexes.

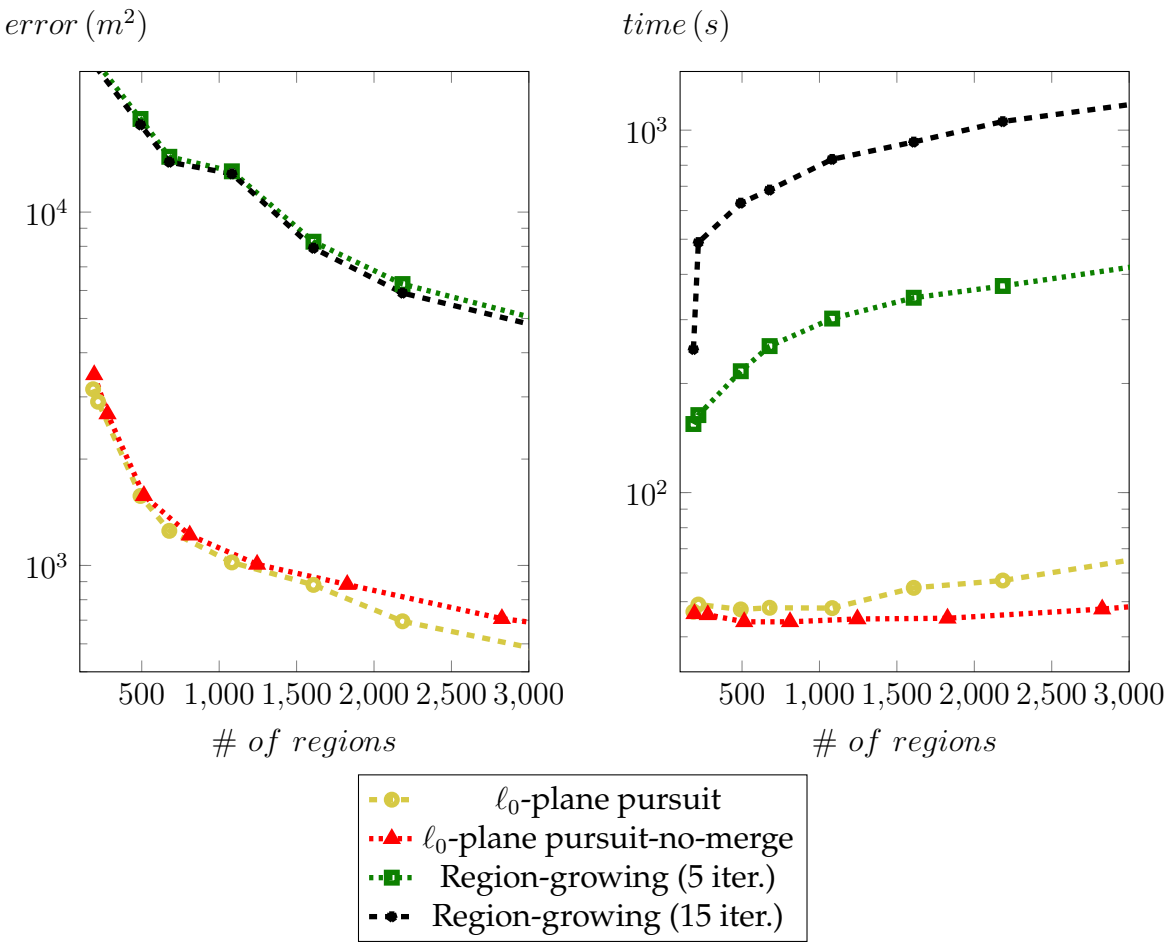


FIGURE 13 – Comparaison de l'erreur entre notre méthode et la méthode basée croissance de région. À nombre de régions fixé, notre méthode est capable de produire des segmentations ayant une plus grande fidélité géométrique, tout en étant plus rapide en termes de temps de calcul.

iii) si le voisinage direct d'un point appartient à trois régions différentes : le point est projeté sur l'intersection des trois plans supports. Cette intersection peut ne pas être définie [3]. Si l'intersection n'existe pas, on se rabat alors sur les trois paires de plans supports possibles et on projette le point sur la ligne d'intersection la plus proche. S'il n'existe aucune ligne d'intersection, on projette le point sur le plan le plus proche.

iv) si le voisinage direct d'un point appartient à quatre régions ou plus : nous choisissons de traiter tous les triplets de plans supports associés à des régions adjacentes et appliquons le même traitement qu'au point précédent.

0.5.2 Simplification de maillages 3D

Nous appelons *simplification de nuages de points 3D*, le fait de diminuer la quantité d'information utilisée pour représenter le nuage, tout en préservant sa géométrie. Le but est ainsi d'obtenir un nuage de points plus facile à manipuler, tout en étant le plus fidèle à la géométrie du nuage de points initial.

Dans la littérature, les méthodes les plus courantes pour simplifier des nuages de points ou des maillages se basent la suppression / fusion de points et la suppression d'arêtes dans les zones géométriquement simples. Nous pensons ici que l'algorithme ℓ_0-plane pursuit peut aider à identifier les régions géométriquement simples et servir à l'initialisation d'un algorithme de décimation de nuages de points. Nous avons sélectionné l'algorithme *Edge Collapse* tel que présenté dans LINDSTROM et TURK (1998).

0.5.3 Évaluation

Afin d'évaluer la qualité de nos approximations, nous proposons de les évaluer selon deux critères :

— Quantité d'information utilisée pour représenter la donnée. Cela est évalué en calculcant la longueur de description minimale permettant de stocker la donnée. Pour un maillage triangulaire, elle s'obtient de la manière suivante :

$$\mathrm{MDL}_{\text{polygons}} = \textit{sizeof(float)} \times (3 \cdot |V| - (\sum_{p \in P} |p| - 3)) \,. \tag{6}$$

— Écart entre l'acquisition initiale et l'approximation finale. Pour cela, nous calculons la somme des distances euclidiennes au carré entre les points de l'acquisition initiale et le maillage simplifié.

0.5.4 Expériences

Nous avons choisi de comparer notre méthode de simplification de maillages basée sur les algorithmes ℓ_0-plane pursuit et *Edge Collapse* à différentes méthodes de simplification de maillages, nommément : VSA (COHEN-STEINER, ALLIEZ et DESBRUN, 2004), Polyfit (NAN et WONKA, 2017) et Poisson (KAZHDAN et HOPPE, 2013). Nous avons aussi évalué l'influence de l'initialisation d'*Edge Collapse* avec une segmentation produite par ℓ_0-plane pursuit. Les résultats sont visibles dans la figure 14.

3. Voir `http://geomalgorithms.com/a05-_intersect-1.html` pour une illustration des différents cas possibles.

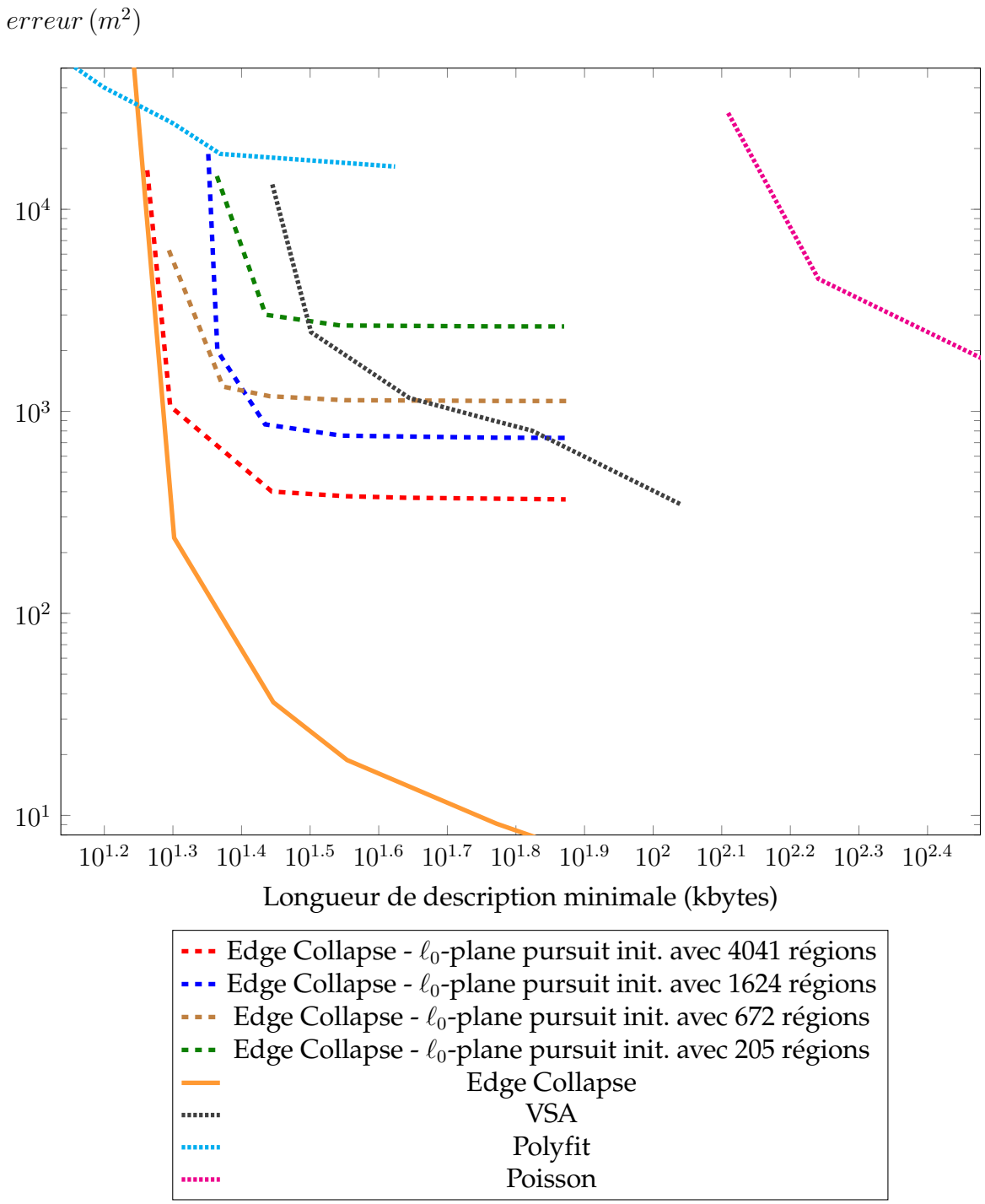


FIGURE 14 – Comparaison de notre méthode de simplification de maillages avec des méthodes de simplification de maillages de l'état de l'art. Lorsque l'algorithme ℓ_0-plane pursuit a été utilisé pour l'initialisation d'*Edge Collapse*, nous avons précisé le nombre de régions composant la segmentation.

Nos expériences montrent que notre méthode de simplification de maillage permet de respecter la donnée d'origine, tout en diminuant grandement la quantité d'information nécessaire pour représenter la donnée. De plus, nos résultats sont comparables à ceux produits par des méthodes de l'état de l'art.

Nous reconnaissons cependant qu'en l'état, notre méthode peut au mieux produire des résultats de la même qualité que ceux d'*Edge Collapse*. Cependant, nous pensons qu'une adaptation de notre méthode permettant de produire un ensemble de polygones et non un maillage permettrait d'obtenir de meilleurs résultats, en diminuant la longueur minimale de description.

0.6 Pré-segmentation pour la classification faiblement supervisée de nuages de points 3D

Dans cette dernière partie technique, nous nous intéressons à la généralisation sémantique des complexes simpliciaux. Pour cela, nous souhaitons évaluer leurs performances pour la classification de nuages de points 3D.

Cependant, nous remarquons que les classifications points par points de nuages de points 3D sont souvent bruitées. De plus, il est long et couteux de créer une vérité terrain suffisamment importante pour entraîner un algorithme de classification de nuages de points. Nous remarquons aussi que les scènes urbaines traitées dans cette thèse sont majoritairement composées d'objets avec des formes simples. Ainsi, nous proposons de capturer la géométrie sous-jacente d'une scène urbaine grâce à un algorithme de segmentation de nuages de points en régions géométriquement homogènes. Ensuite, nous proposons de régulariser une classification points par points bruitée, obtenue après avoir entraîné un algorithme de classification sur un nombre réduit d'exemples.

Nous nous intéressons dans un premier temps à la recherche de descripteurs permettant de décrire la géométrie de la scène. Ensuite, nous présentons l'algorithme de segmentation utilisé pour régulariser nos classifications bruitées. Enfin, nous détaillerons le processus de régularisation en lui-même.

0.6.1 Calcul de descripteurs

Afin de décrire la géométrie locale du nuage, nous proposons quatre descripteurs locaux. Ces descripteurs sont calculés à partir des valeurs propres et vecteurs propres de la matrice de covariance créée à partir de la position des voisins d'un point. Il s'agit de la linéarité, la planarité, la dispersion et la verticalité. La taille de voisinage choisie est le voisinage optimal tel que présenté dans WEINMANN et al. (2015b).

Nous proposons aussi deux descripteurs globaux : l'élévation et la position par rapport à la route. Ce dernier descripteur est calculé à partir de la projection du point sur l'α-shape (AKKIRAJU et al., 1995) des points classés en route. Cette classification est une classification binaire en route / non-route calculée à partir des quatre premiers descripteurs locaux.

À cela, nous ajoutons les complexes simpliciaux, calculés sur le nuage de point en entrée et convertis en un ensemble de trois descripteurs grâce à la technique du *one-hot* encoding.

L'ensemble de ces descritpeurs peut être utilisé pour entraîner un classifieur de type forêts aléatoires (BREIMAN, 2001).

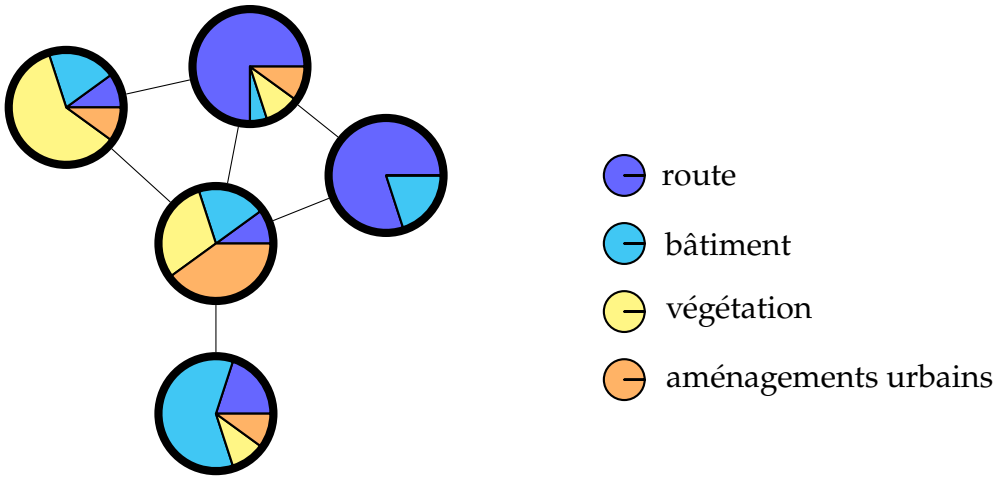


FIGURE 15 – Illustration de la classification des segments, basée sur l'aggrégation des probabilités de classification par points à l'intérieur de chaque segment. Chaque diagramme montre un segment différent. Chaque couleur représente une classe différente.

0.6.2 Segmentation en régions homogènes

Soit $G = (V, E)$ le graphe représentant les données d'entrée et encodant leur adjacence. Afin de réaliser une segmentation en régions géométriquement homogènes de la scène, nous avons décidé d'associer à chaque point le vecteur contenant les quatre descripteurs locaux $f_i \in \mathbb{R}^4$. Nous cherchons ensuite à obtenir une approximation constante par morceaux du signal $f \in \mathbb{R}^{V \times 4}$, en minimisant l'énergie suivante :

$$g^\star = \underset{g \in \mathbb{R}^{4 \times V}}{\arg\min} \sum_{i \in V} \|g_i - f_i\|^2 + \rho \sum_{(i,j) \in E} \delta(g_i - g_j \neq 0) \,, \tag{7}$$

avec $\delta(\cdot \neq 0)$ la fonction de $\mathbb{R}^4 :\mapsto \{0, 1\}$ valant 0 en 0 et 1 partout ailleurs. La première partie de cette énergie correspond au terme d'attache aux données et la seconde partie, à la régularisation, permettant de s'assurer que les régions ont des formes simples.

Remarquons que cette formulation du problème ne comporte aucun prérequis sur la taille des régions ou leur forme. Cela nous permet d'obtenir une segmentation adaptative à la géométrie locale des données. Afin de résoudre ce problème d'optimisation, nous proposons d'utiliser l'algorithme ℓ_0-cut pursuit, proposé par LANDRIEU et OBOZINSKI (2017).

0.6.3 Classification des segments

Afin de régulariser une classification par points bruitée à l'échelle des régions de notre segmentation, nous proposons d'agréger les probabilités de classification par classe et par points à l'échelle des segments. Le label associé à la classe ayant la plus forte probabilité à l'échelle du segment est ainsi attribué à tous les points composant le segment. Ceci est illustré sur la figure 15.

0.6.4 Expériences

Nous avons mené des expériences sur trois jeux de données différents. Le premier est déjà utilisé dans cette thèse et correspond à une acquisition dans les rues de Paris. Le deux autres jeux de données sont publics, il s'agit d'Oakland (MUNOZ et al., 2009)

et Semantic3D (HACKEL, WEGNER et SCHINDLER, 2016) (et plus particulièrement la scène Domfountain).

Dans une premirère expérience, nous avons évalué les performances des descripteurs locaux, globaux et des complexes simpliciaux pour la classification points par points des données. Ensuite, nous avons comparé une classification par points bruitée avec sa version régularisée par un CRF (NIEMEYER, ROTTENSTEINER et SOERGEL, 2014) et sa version régularisée par notre segmentation.

Les expériences sont évaluées avec les métriques suivantes : précision, rappel et FScore, par classe et globaux.

Pour la premirère expérience, la comparaison des descripteurs a montré que les complexes simpliciaux étaient trop bruités pour réllement améliorer les performances d'un algorithme de classification de nuages de points, en particulier dans un contexte faiblement supervisé. Les résultats détaillés pour cette expérience sont présentés dans le tableau 6.2. Ainsi, nous avons décidé de ne garder que les descripteurs locaux et globaux pour les expériences suivantes.

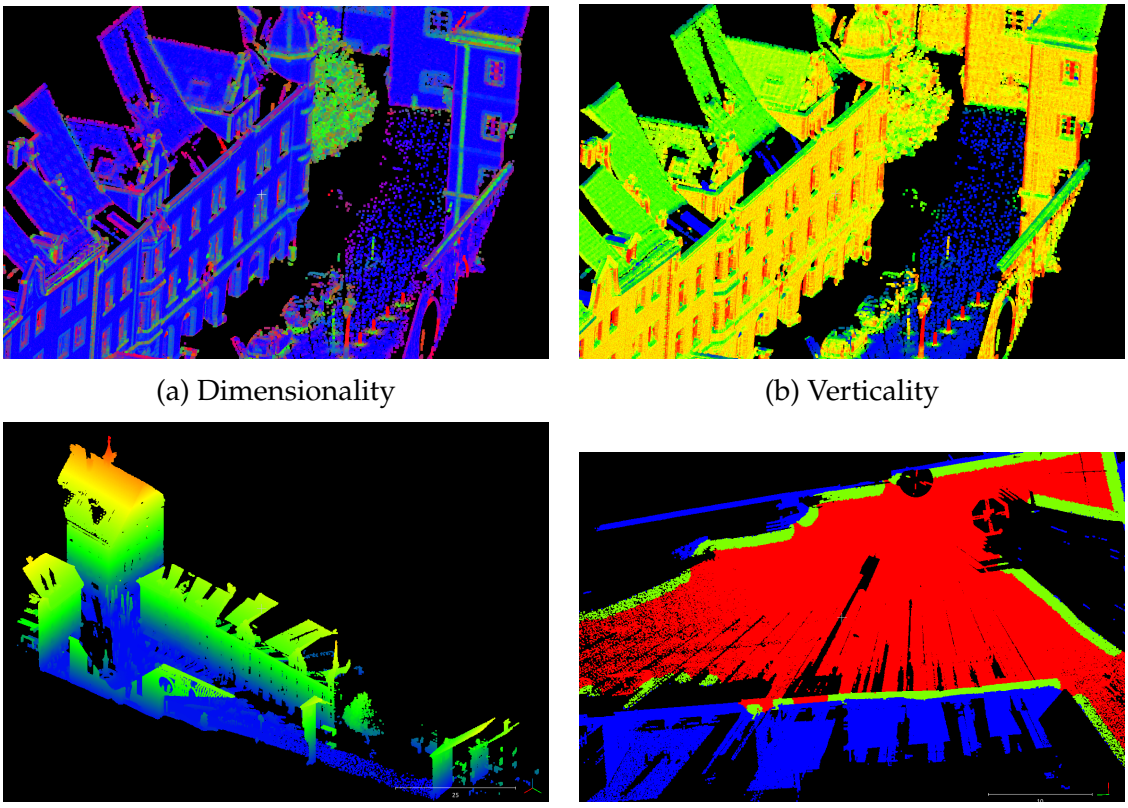
Nous avons ensuite comparé trois classifications différentes sur les jeux de données publics présentés précédemment. Les résultats pour Oakland sont présentés dans le tableau 6.3 et ceux pour Domnfountain dans le tableau 6.4. De plus, une illustration de notre méthode est visible dans la figure 16. Nous remarquons que même si notre méthode n'améliore le FScore global que de quelques pourcents par rapport à la régularisation basée sur un CRF, le FScore des classes difficiles à repérer (poteaux, voiture, . . .) est nettement meilleur.

0.7 Conclusion

0.7.1 Résumé des principales contributions

Dans cette thèse, nous avons étudié l'utilisation de complexes simplistes pour la reconstruction de scènes urbaines en 3D acquises avec un capteur LiDAR. Nous avons présenté les contraintes associées au traitement des données LiDAR et notamment les problèmes de données manquantes et le manque de structure interne reliant les points 3D. Dans les scènes urbaines, il existe différents cas dans lesquels un objet cache une partie de la scène. Cela peut entraîner des trous et des occlusions dans les scans LiDAR. Afin de récupérer la géométrie manquante, il existe certaines techniques, telles que l'inpainting ou la reconstruction continue, qui permettent de déduire la géométrie manquante et d'ajouter des informations plausibles à l'acquisition d'entrée. Dans cette thèse, nous avons choisi de ne pas ajouter d'informations aux données originales. L'objet résultant est une collection de points, d'arêtes et de triangles et est appelé un complexe simpliste. La reconstruction de complexes simpliciaux nous permet de préserver de petits détails géométriques et donne des informations significatives sur la géométrie locale de la scène. Nous reconnaissons que les complexes simpliciaux ne permettent pas une reconstruction étanche et que leur localisation élevée les rend plus difficiles à interpréter pour la visualisation.

De plus, les complexes simpliciaux des scènes urbaines sont composés de centaines de milliers de simplexes. Les objets géométriquement simples, tels que les routes ou les façades, sont composés de milliers de triangles coplanaires. Nous soutenons que cette géométrie peut être approchée avec précision par un petit nombre de primitives. C'est pourquoi nous avons décidé d'ajouter une généralisation axée sur les éléments simples de dimension deux (c'est-à-dire les triangles). Nous avons conçu une approche globale qui prend comme entrée un ensemble de points ayant une relation de contiguïté (tels



(a) Dimensionality (b) Verticality

(c) Elevation (d) Position with respect to the road

FIGURE 16 – Représentation des quatre descripteurs locaux et des deux descripteurs globaux. Pour (a), le vecteur dimensionnalité [linéarité, planarité, dispersion] est représenté par le vecteur de couleurs [rouge, vert, bleu]. Pour (b), la valeur de la verticalité est représentée avec un code couleur allant du bleu (faible verticalité - routes) au vert / jaune (verticalié moyenne - tois et façades) et au rouge (verticalité élevée - poteaux). L'élévation est représentée dans (c). La figure (d), illustre la position par rapport à la route, avec le code couleur suivant : intérieur de l'α-shape de la route en rouge, bordure en vert et extérieur en bleu.

que les triangles de nos complexes simplistes) et divise itérativement une scène en un ensemble de régions planes. Notre approche, appelée ℓ_0-plane pursuit, segmente itérativement un nuage de points d'entrée dans un ensemble de régions planes. Cette méthode s'adapte à la géométrie du nuage, préservant ainsi la géométrie hautement résolue en affinant localement la segmentation, tout en agrégeant des milliers de simples pour des zones géométriquement simples telles que les routes ou les façades. Notre méthode a donné des résultats prometteurs, tant en termes de vitesse de calcul que de qualité géométrique de l'approximation.

Sur la base des résultats de l'algorithme de segmentation ℓ_0-plane pursuit, nous avons proposé une approche pour fusionner les triangles coplanaires avec l'algorithme d'effondrement des bords. Nous avons évalué la qualité de l'approximation en projetant chaque point du nuage original sur le modèle simplifié. Nous avons également évalué le *degré de généralisation* de nos modèles en calculant leur MDL, qui correspond à la quantité d'informations nécessaires pour représenter les données. Notre approche a donné de meilleurs résultats que les approches classiques de modélisation 3D. Cela signifie que l'agrégation de simples 2D au sein de complexes simplistes est une méthode valable pour généraliser les nuages de points 3D des scènes urbaines.

Enfin, nous avons étudié les performances des complexes simpliciaux en tant que

structure de données. Plus spécifiquement, nous avons évalué les performances des algorithmes de classification opérant sur des nuages de points 3D. Nous avons montré comment les complexes simpliciaux pouvaient être utilisés comme descripteurs géométriques 3D. Conformément à notre objectif de nous appuyer sur le moins d'informations étrangères possible, nous avons formé une forêt aléatoire avec seulement quelques échantillons pour chaque classe sémantique. Nous avons fait valoir qu'une classification ponctuelle des nuages de points 3D n'est généralement pas de grande qualité, mais qu'ils peuvent être régularisés dans l'espace à l'aide d'un algorithme de segmentation, comme ℓ_0-plane pursuit. Cependant, nos expériences ont montré que les complexes simplistes étaient trop dépendants de la géométrie locale de la scène pour améliorer de manière significative une telle classification ponctuelle.

Au final, nous avons proposé une méthode simple et légère pour reconstruire les complexes simpliciaux à partir d'acquisitions de données LiDAR 3D de scènes urbaines. Nous avons fait valoir que les complexes simpliciaux sont une alternative pour les maillages qui permet de préserver toutes les informations géométriques sans modifier les données. Nous avons montré que les complexes simpliciaux peuvent être généralisés avec un nombre limité de primitives également et que des approches globales peuvent être utilisées pour produire des modèles géométriquement précis, mais compacts, de scènes urbaines en 3D.

0.7.2 Perspectives

0.7.2.1 Généralisation des complexes simpliciaux

Comme vue au chapitre 6, les complexes simpliciaux reconstruits dans cette thèse sont encore trop bruités pour réellement améliorer des algorithmes de classification. Nous pensons qu'une analyse incorporant plus de connaissances globales des données (dans la même veine que celle présentée dans la section 3.2.3) serait à même de générer des complexes simpliciaux plus géométriquement homogènes. De plus, nous pensons qu'une telle approche pourrait aussi être utilisée afin de combler les occlusions et les trous qui apparaissent dans les données.

0.7.2.2 Approximations multi-primitives

Dans cette thèse, nous nous sommes uniquement focalisés sur la généralisation planaire par morceaux des parties triangulées de nos complexes simpliciaux. Cependant, certains objets dans les scènes urbaines sont suffisamment grands pour être triangulés, mais n'ont pas une géométrie localement planaire. C'est le cas notamment des troncs d'arbres, qui sont plutôt cylindriques. Ainsi, nous pensons que l'algorithme ℓ_0-plane pursuit, présenté dans le chapitre 4 devrait être adapté afin de prendre en compte d'autres types de primitives (cylindres, sphères, tores, ...).

De même, dans cette thèse, nous ne nous sommes pas intéressés à la généralisation des points et segments de nos complexes simpliciaux. Cependant, nous pensons qu'une approche similaire à celle présentée ici peut être adaptée pour traiter ces cas.

0.7.2.3 Reconstruction de villes

Les travaux présentés dans ce manuscrit visent à reconstruire des scènes urbaines, tout en préservant la qualité géométrique de l'acquisition. Nous pensons qu'une approche à la VERDIE, LAFARGE et ALLIEZ (2015) peut être adaptée afin de générer des

reconstructions adaptées à la géométrie locale de nos complexes simpliciaux. Cette approche se baserait ainsi sur une classification directe des simplexes, avec une méthode similaire à celle présentée au chapitre 6.

Un autre problème à traiter est celui de l'absence d'information dans certaines zones de la scène, due aux occlusions. Nous pensons ainsi qu'une méthode d'inpainting à la DAI, DILLER et NIESSNER (2019) serait à même de recouvrer la géométrie manquante.

Enfin, une reconstruction à au niveau de détail LoD-3 de scènes urbaines avec une méthode capable de préserver la qualité géométrique de l'acquisition serait tout particulièrement utile afin de créer des modèles 3D de villes entières. Cela faciliterait le calcul de simulations (inondation, tremblement de terre, etc.), ou de rendu pour les films et jeux vidéos (PIEPEREIT et al., 2019 ; GABELLONE et al., 2017).

0.7.2.4 Traitements temps réel

Enfin, les reconstructions de complexes simpliciaux, telles que présentées dans la section 3.2.2.2 sont très rapides à calculer (environ 5 secondes sans la régularisation basée wedge pour un bloc d'une seconde d'acquisition) et permettent d'envisager des applications temps-réel, comme le positionnement de véhicules équipés de LiDAR, ou la détection de changement, par exemple sur des infrastructures ferroviaires (ARASTOUNIA, 2015).

1 Introduction

Contents

The purpose of remote sensing is to extract information from scenes using sensors without physical contact. This can be done through wave propagation, either in the visible light spectrum with cameras or LiDAR or at other wavelengths, as for Radar sensors. However, such sensors provide incomplete spatial information, especially in urban scenes, which are geometrically complex. This can be due to occlusions, or the geometric configuration of the scene. Hence, a major challenge of the remote sensing community is dealing with missing or incomplete data. The most straightforward way to overcome missing data is to perform several acquisitions of the same scene, with different sensors or at different times. However, this can be costly, both in terms of time and money. Furthermore, all the different acquisitions have then to be co-registered.

One of the applications of remote sensing research is to provide 3D models of real scenes. This is especially important in urban areas, whose rapid growth must be monitored by state actors, but also to perform simulations for environmentally-related hazards. In order to achieve this modeling task, several methods have been developed for reconstructing 3D cities at scales varying from a single building to a whole city, and with different final quality modelisations. However, reconstruting urban scenes remains a challenge, as they are composed of many different objects with varying shapes and sizes. These objects range from roads, which have a large spatial extent, to poles, which are short and have a smaller footprint. Moreover, when dealing with large scenes, 3D reconstruction methods often have to cope with irregularly sampled data and many small geometric details. Reconstruction algorithms have to take both of these constraints into account, which complicates the reconstruction task. Furthermore, due to memory and computational limitations, current desktop devices cannot display the full 3D reconstruction of a whole city. However, urban scenes usually exhibit some geometric regularity for geometrically simple objects, such as roads or facades. This has lead researchers to investigate the possibilities of simplifying such 3D models in order to decrease their memory footprints, while preserving the geometric quality of the model. However, this simplification process is typically accompanied by a loss of the geometrical quality of the input reconstruction.

In this thesis we investigate the reconstruction of urban scenes, with the smallest possible loss on the geometrical quality of the acquisition, without combining multiple sensors or acquisitions. We argue that using all the available informations of a sensor is enough for a geometrically accurate reconstruction of a scene. We want our approach to stay as simple as possible, so we do not infer any additional information to the acquisition (no new / duplicated points) and we want to achieve the compromise between the memory size of the 3D model and its fidelity to the real scene.

We start by presenting the different types of sensors used for the acquisition of urban scenes, and explain why we chose to focus on LiDAR data. Next, we explain some choices made to design our reconstruction method. Then, we present some specifications of our reconstructions, in terms of hole filling and geometric quality. Finally, we present the organisation of this thesis.

1.1 Extracting Informations from Urban Scenes

In the remote sensing community, acquisitions of urban scenes are most commonly performed with three different types of sensors: cameras, LiDARs, and Radar. Cameras are cheap sensors capturing images, which can be used to produce 3D reconstructions but lack robustness. LiDAR sensors are more expensive than cameras, and also perform geometrically accurate acquisitions, with a particular sensor topology. Radar sensors are also expensive and perform acquisitions with a low density. In this section we present each of these sensors, show their respectives strengths and weaknesses and present some applications.

1.1.1 Camera

1.1.1.1 RGB sensors

A camera is an optical sensor able to capture light through a small hole (called *aperture*) and to create an image. The ancestor of our modern camera is the so-called *camera obscura*. A *camera obscura* is a natural optical phenomenon which occurs when a scene faces a screen and is projected on the screen via a small hole. For a better quality of image, the room usually has to be darkened. It is hard to know when the first experiments involving this process were done; Schramm (1961) suggests that Anthemius of Tralles, a Greek mathematician of the 6^{th} century already understood the principle of *camera obscura*. Throughout the centuries, many scientists studied the *camera obscura*. Some famous works were done by Leonardo Da Vinci (Keele, 1955) or Johannes Kepler (Dupré, 2008), describing this natural optical phenomenon and showing how to create a *camera obscura*. The first attempt at taking a physical picture was done by Joseph Nicéphore Niépce in 1816, the process is explained in Gernsheim (1986). Niépce improved the process in the 1820's. With Charles and Vincent Chevalier, they designed a wooden box with a surface coated with Bitumen of Judea (Davenport, 1999). The light penetrating in the box hardens the bitumen (the more light, the more bitumen hardens). At the end of exposition time (in hours for the first experiments), the remaining soft bitumen is removed. An example of image acquired with this process is shown on Figure 1.1. Modern photography is divided in two different types: analog and digital photography.

Analog photographies (also called *argentic*) designate the presence of a physical film in the camera. In such camera, the surface opposed to the aperture is covered with silver halides [4]. Analog photography was popularised by the introduction of a film band in cameras by George Eastman (Curme and Rand, 1997). This type of photography is still actively used nowadays, especially because the film sensitivity allow for high quality images with a large dynamic range. Also, analog images can remain of high quality even after a few decades. It is also important to note that most of photogrammetric techniques and algorithms were first developed for processing analog images (Thompson et al., 1966; Wasil and Merchant, 1964). Analog images can serve as input for orthophotography generation (Otepka and Loitsch, 1979). Recent works using analog images usually focus on processing historical images for 3D modeling.

4. Silver halide's chemical notation is AgX. Photons absorbed by AgX stimulate the electrons of AgX, resulting in a crystallizing effect, forming silver metal atoms. These metallic atoms forms the latent image which is then made visible and insensitive to light thanks to the photographic development process. The photographic development consists in soaking the latent image in water in order to remove non-crystallized AgX. Then, the image is immersed in a specific acid to stop the chemical reactions, and a fixer is added to prevent the image to react again to light.

Maiwald et al. (2018) used the images of the Saxon State and University Library Dresden for modeling the city of Dresden (Saxony, Germany). Analog images are also used for monitoring events at a large time scale (usually several decades), for instance Mölg and Bolch (2017) monitor the movement of a glacier using the earliest available images, which were produced with an argentic camera.

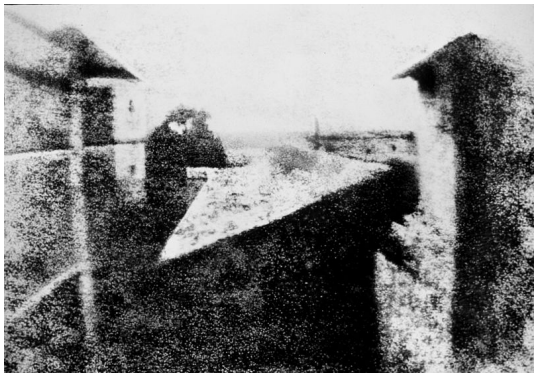


Figure 1.1 – View from the window at Le Gras, Joseph Nicéphore Niépce (1826-1827)

Unlike analog cameras, digital cameras use an array of photodetectors to capture the light and create an image. The first digital image was produced by Russel Kirsch in 1957 [5]. The first applications for digital images are for TV broadcasting and space observation: unlike analog images, digital images could be taken from a satellite far from Earth, automatically encoded and sent back to Earth (O'Handley and Green, 1972). For instance, the Mars Viking Lander captured images of the planet that were then processed on Earth (Ruiz et al., 1977). These images greatly improved our knowledge of Mars' surface. Digital cameras became more and more used since the 1990's. They have the advantage of allowing people to take their own photographies, without any technical knowledge and without waiting for the photographic development. Also they allow to take many more photos in a single survey than analog cameras. Modern digital cameras allow to directly pre-process images in the camera. This kind of pre-processing step became possible due to the fact that digital images are entirely numerical. However, digital images suffer from aliasing and usually have a smaller dynamic range than analog cameras. Nowadays, digital images are used for nearly all photogrammetric processes (Rupnik, Daakir, and Deseilligny, 2017). For instance, Balsa-Barreiro and Fritsch (2018) used digital images for creating 3D city models. Digital images are also easily combined with LiDAR data (Yastikli, 2007), and some platforms (e.g. TLS) can combine both sensors (Paparoditis et al., 2012). Leberl and Thurgood (2004) reviewed the differences between analog-vased photogrammetry and digital photogrammetry.

1.1.1.2 Digital Images

Images can be viewed as a 2-dimensional (2D) array, where each cell of the array is a pixel. In this case, each cell contains:

— Panoptic image: an intensity information corresponding to the quantity of light acquired in the whole visible range RGB information if the image is in color,

— RGB image: a quantity of red, green and blue light acquired by each pixel,

5. https://www.nist.gov/news-events/news/2007/05/fiftieth-anniversary-first-digital-image-marked

— Depth image: an image can be used to represent the geometry of a scene by storing the depth of the scene in the direction corresponding to each pixel. Some devices such as the Kinect (Zhang, 2012) can produce directly such depth images while dense matching allows to produce such depth maps from tuples of images (which is called stereovision in the case of 2 images).

As images can be viewed as 2D arrays, a natural topology arises, where each cell is connected to its direct neighbors vertically and horizontally (Von Neumann's neighborhood) or even in diagonal (Moore's neighborhood) as explained in Toffoli and Margolus (1987, Section 7.2). This topology is illustrated on Figure 3.3.

1.1.1.3 Applications

The remote sensing community use image processing techniques for various applications. Image processing has been researched since the 1960's for image enhancing (Rosenfeld, 1969). The improving quality of images, in particular with digital cameras, allowed for using image-based data for various purposes, including photogrammetry and computer vision (Jain, 1989). The computer vision community used images for investigating various challenges, such as object recognition (Radovic, Adarkwa, and Wang, 2017), semantisation (Badrinarayanan, Kendall, and Cipolla, 2017; Chen et al., 2018) and navigation for autonomous driving or blind persons (Martinez et al., 2017).

In this section we focus on 3D reconstruction from images. This process usually relies on the *stereophotogrammetry* technique to extract 3D informations from a set of images (typically two) (Thompson, 1908; Hotine, 1930). In fact, when a scene has been acquired by two or more images, it is possible to reconstruct the scene in 3D using the *stereo-matching* technique, as shown on Figure 1.2. Stereo-matching consists in computing a disparity map, encoding the horizontal difference of corresponding image points. This allow to compute the relative depth variation of the considered pixels. Also, stereo-matching requires the following pre-processing steps: the images must be projected on the same plane and the distorsion due to the camera should be corrected. Stereophotogrammetry lead to the production of orthoimages, 3D point clouds and Digital Surface Model (DSM) (Hallert, 1960; Mikhail, Bethel, and McGlone, 2001). In order to perform the stereo-matching in an urban scene, stereophotogrammetry usually requires dozens of images, notably for 3D information extraction and object identification from several point of views. This is due to the limited portion of a scene that a single image is able to capture. Recently, many researchers focused on extracting 3D information from a single image using Neural Networks (NN) (Ledig et al., 2017; Groueix et al., 2018). Such methods give promising results. However, NN requires millions of paired image and point clouds for training, as opposed to *stereophotogrammetry*, which does not rely on learning attributes. Also, NN may not be able to perform well on scenes not represented in the training set.

In the end, we want to emphasize on the fact that stereophotogrammetry usually requires dozens images for reconstructing a single 3D scene. These images have to be co-registered, which is time-consuming. Also, images taken from cameras suffer from distorsion and saturation due to irregular quantities of light throughout the image. These flaws have to be corrected before processing images. We think that all of the above constraints can lead to 3D reconstructions of lower quality. In this thesis, we will focus on sensors that produce data which can be processed directly. Hence, we investigate the 3D reconstruction of scenes acquired with active sensors such as LiDAR and Radar.

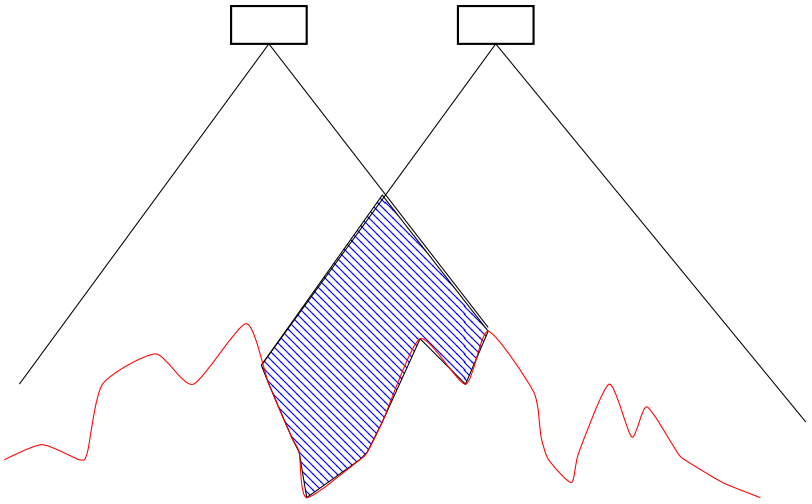


Figure 1.2 – Illustration of an acquisition of a 3D scene using cameras. The hatched area is viewed from two images, taken from different places. This allows for a depth estimation in this area.

1.1.2 LiDAR

Another type of sensor for extracting informations on real scenes is LiDAR scanners.

1.1.2.1 LiDAR Sensors

LiDAR sensors are active sensors relying on light emitted by a Light Amplification by Stimulated Emission of Radiation (LASER) and recording the light reflection on real objects. It is then possible to find the 3D position of the object that reflected the LASER beam by considering the direction of the sensor at the emission time and the elapsed time between light emission and reception. The sensor itself is described on Section 2.1.3. As the emitted light reaches an object, part of the light is reflected, but in some situations (semi-transparent object, or light reaching the edge of an object) that part of the light reaches further objects. This situation gives an important information that two objects are in front of each other.

LiDAR sensors can be mounted on various platforms, from trolleys to aircrafts. They are often combined with a GPS and a Inertial Measurement Unit (IMU) for georeferencing purposes. Last, LiDAR sensors can be coupled to a camera in order to produce a simultaneous acquisition of a scene with both sensors.

1.1.2.2 LiDAR Data

LiDAR data is usually processed as a set of 3D points with no connectivity relationship. Also the sampling density varies with the distance between the sensor and the scanned objects. Processing LiDAR data leads to a first problem: there is no straightforward approach for connecting points. In order to add an adjacency relationship between points, most of the works in the litterature focus on *Delaunay Triangulation* or *Nearest Neighbors* approaches. These approaches are purely geometric and based only on the 3D point cloud. In the litterature, the raw sensor data (comprising the time and angle of each light emission and the number of physical objects encountered) is often lost. We argue that this information can be used for representing LiDAR point cloud in a two-dimensionnal space that we call the *Sensor Topology*. Also, using these infor-

mations, one can divide the data in *acquisition lines*. One acquisition line correspond to all the LASER beams and their returned echoes for a 2π rotation of the sensor. This particular topology is further explained and illustrated in Section 3.2.1.

1.1.2.3 Applications

LiDAR sensors are increasingly used in more and more diverse applications. LiDAR-based applications range from 3D modeling (Caraffa, Brédif, and Vallet, 2016; Hoppe, 1996; Nan and Wonka, 2017) to dam or tunnel monitoring (Han, Guo, and Jiang, 2013). LiDAR sensors are also commonly used in forestry, with applications to biomass estimation or tree inventory (Hyyppä et al., 2017). Some autonomous cars are also equipped with LiDAR sensors (Ibisch et al., 2013). A detailed state-of-the-art for LiDAR applications is presented in Section 2.2.

1.1.3 Radar

Kirscht and Rinke (1998) showed that 3D building reconstruction could be done using Radar data. Hence, we now present Radar sensors and their different applications.

1.1.3.1 Radar Sensors

A radar is a system using radio waves [6] to compute the range, the angle and the velocity of objects. A Radar system is composed of a trasmitter, a waveguide, and a receiver. The role of the transmitter is to emit radio waves. The waveguide transfers the radio waves to the antenna. When these radio waves reach an object, part of them are reflected in the Radar direction. This reflected signal is then captured by the receiver and can be interpreted in the same way as the received light for a LiDAR for detecting objects. In the case where the same antenna is used for emission and reception, Radar systems use a duplexer to redirect, either the emitted signal to the antenna or redirect the received signal to the receiver.

First experiments using Radar-like systems were done at the beginning of the 20^{th} century, by Christian Hülsmeyer [7]. In 1904 he showed that it was possible to detect the presence of a metallic object using radio waves, though he was unable to find its position. Experiments in the 1930's showed that it was possible to detect ships, but also planes using radio waves (Taylor, Young, and Hyland, 1934; Blanchard, 2016). Radars were first developed for military purposes in the WWII context. Most of the involved countries tried to develop their own system, using radio waves, to detect distant or moving objects such as ships. Radar prototypes include the American (Colton, 1945), Soviet (Kostenko, Nosich, and Tishchenko, 2001), Japanese (Sato, 1991) or New Zealander ones (Unwin, 1992).

Modern Radar sensors include Airport Surveillance Radar (ASR) (Radar sensors for tracking airplanes close to airports) or imaging Radars such as SAR or Inverse Synthetic-Aperture Radar (ISAR). SAR is a mobile Radar that uses the *Doplller effect* to create 2D images and 3D reconstructions of large scenes (Ramakrishnan et al., 2002). ISAR is the opposite of SAR: it is a fixed Radar designed to detect moving objects based on the *Doplller effect* (Chen and Martorella, 2014). An scheme showing the acquisition process of an imaging SAR is presented in Figure 1.3.

6. Radio waves vary between 30 Hz and 30 GHz.
7. `http://www.radarworld.org/huelsmeyer.html`

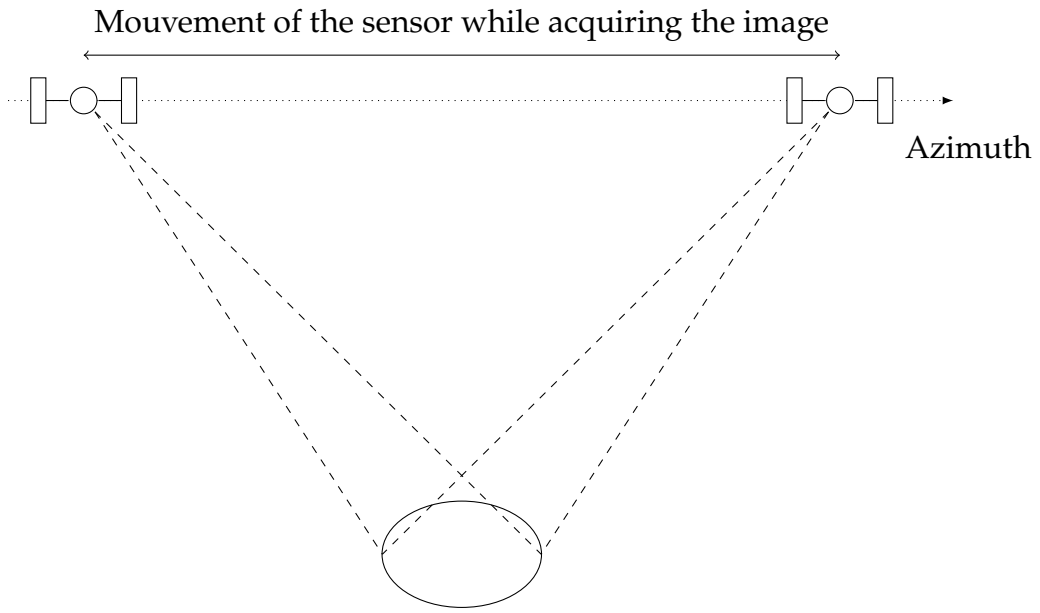


Figure 1.3 – Representation of the process used by a SAR for creating an image.

1.1.3.2 Radar Data

Radar sensors provide two distinct informations: a dephasing information and an amplitude information. The first one gives an information on the type of material encountered [8]. The latter is related to the signal quantity received by the sensor, which gives an information on the object rugosity [9]. For more details on Radar raw data processing, we refer the reader to the work of Emery and Camps (2017).

1.1.3.3 Applications

As said in Section 1.1.3.1, Radar sensors were first developed for military purposes: it was supposed to help the detection of ships and planes in bad weather conditions (Guerlac and Boas, 1950; Crisp, 2004). For instance, Marthaler and Heighway (1979) used a Side Looking Airborne Radar for detecting icebergs and ships. An exemple of ship detection based on Radar images is shown on Figure 1.4. Radar are also used for detecting smaller objects than ships, such as aircrafts (Khan and Power, 1995). This lead to the development of ASR that allow the tracking of planes along most current airports (Taylor and Brunins, 1985; Galati et al., 2010).

However, Radar applications are not limited to ship or plane detection. Graber and Hassler (1962) suggests to use an airborne Radar for bird detection. In Dokter et al. (2011), the authors use a set of Radar to monitor bird migrations. At a larger scale, Radar are used for ocean monitoring (Barrick, Evans, and Weber, 1977), from current mapping (Goldstein and Zebker, 1987) to phytoplankton estimation (Esaias, 1980).

At an even larger scales, Radar are used for analysing the structure of outer planets (Hapke, 1990). For instance, the Magellan mission [10] consisted in sending a Radar orbitting Venus in order to study the surface of the planet (Johnson, 1991). More recent studies focus on icy satellites, such as Europa (Aglyamov, Schroeder, and Vance, 2017) thanks to the RIME sensor (Bruzzone et al., 2013) and the CLIPPER mission (Phillips

8. Each type of material has a different *penetration* rate given a fixed wavelength.
9. `https://planet-terre.ens-lyon.fr/article/interferometrie-radar.xml` (in French)
10. `https://nssdc.gsfc.nasa.gov/planetary/magellan.html`

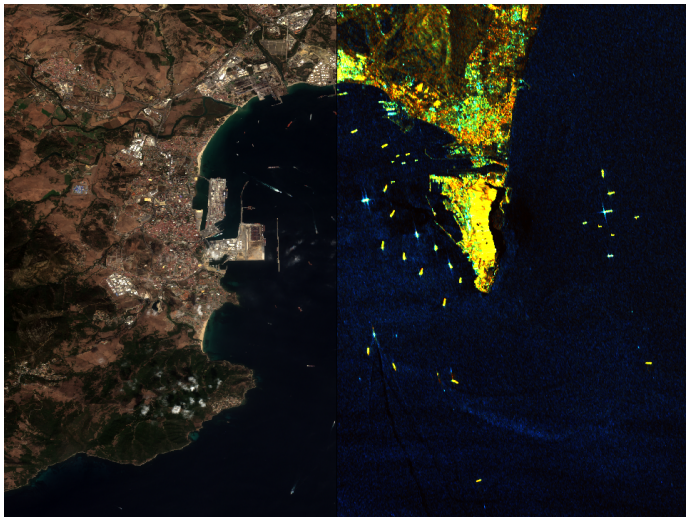


Figure 1.4 – Optical image (left) and Radar image (right) used for ship detection in the Gibraltar straight. The geometric complexity of ships makes them easily identifiable from a Radar point of view. Courtesy of Anatol Garioud (IGN-LaSTIG-STRUDEL).

and Pappalardo, 2014).

1.1.4 Sensor Choice

We choose to study LiDAR sensors as they allow for a geometrically accurate representation of a continuous reality. Unlike cameras, LiDAR sensors are robust to light changes and do not require any pre-processing step. Also, the density of the acquisition is higher for LiDAR-based acquisitions than for Radar-based ones. LiDAR also provide an inherent topology, based on the time of pulse emission and angle of the LASER, that we can use in order to improve the reconstruction of 3D point clouds based on the local geometry of the data. We argue that an approach based on the local geometry of the cloud and the sensor topology will result in an accurate geometric representation of the scene. Also, urban scenes usually display a local geometric regularity. This is due to the fact that most objects in such scenes are designed by humans. We think that a 3D reconstruction of an urban scene should display this geometric regularity. This means that such reconstructions can be approximated with a limited number of primitives.

1.2 Level of Detail

We now focus on the geometric quality that our reconstruction should achieve. In order to contextualise our observations, we first introduce some definitions, and then explain the choices made to design our reconstruction method.

1.2.1 3D Reconstruction

We call *3D reconstruction* the creation of a geometric continuity in a scene of interest. The 3D reconstruction aims at respecting two priors: compacity and geometric fidelity:

— Compacity: the compacity corresponds to the amount of information needed to represent the data. In computer science, this can translates by the number of bytes used to store the data.

— Geometric fidelity: this translates by the gap between the original scene and the resulting 3D model. This geometric fidelity is dependent of the chosen evaluation metric.

To our knowledge, there exists no standard representation for all the objects in an urban scene. However, there exists some standard representation for buildings (Biljecki, Ledoux, and Stoter, 2016; Löwner et al., 2013), one of the most used being CityGML (Kolbe, Gröger, and Plümer, 2005), but we do not review them, as they have not been generalized for all objects composing 3D urban scenes.

1.2.2 Positioning of our Study

In this thesis, we propose to characterize the Level of Detail (LoD) of our reconstructions as the size of the smallest geometrical details reconstructed. We do not use the CityGML or other building-based LoD definitions because urban areas comprises objects for which LoD are not defined, such as trees or cars. We aim at reconstructing all the geometrical details of the acquisition. This includes small objects like tree branches and traffic signs. We acknowledge that due to missing data, it is not possible to reconstruct full objects without adding meaningful information. However, we do not want to worsen the quality of the data. Thus, we will not focus on full object / building reconstruction, but on reconstructing surface only where enough information is available. Our goal is to exploit all the informations acquired by a sensor. This means that we do not want to restrain ourself to the reconstruction of building footprints or 3D shapes like Cheng et al. (2011). Instead, we study the reconstruction of more global shapes than just buildings. We want to retrieve the geometrical details up to building interiors if available.

In the next section, we investigate the 3D reconstruction from LiDAR data in urban scenes, and show how such data can be used to reconstruct a surface when not enough geometric information is available.

1.3 3D Scene Reconstruction from LiDAR Data

There exists two main possible representations of LiDAR data:

— point clouds: they have a high geometric accuracy but have a very low compacity and do not display the continuous reality.

— meshes: they have a higher compacity and represent a continuous surface between points.

Point clouds do not fill our needs, hence, in the remaining of this section, we focus on mesh-based reconstructions. In urban scenes, from a sensor point of view, there are various objects on different levels. This means that it is possible that an object is hiding part of another object. When acquiring such objects, LiDAR sensors will miss the geometric information behind the foreground objects. Hence, a typical LiDAR scan in an urban scene will show some holes behind foreground objects. In order to design our reconstructions from LiDAR data, we first investigate the hole problem in LiDAR data. Then, we study the LoD that we want our reconstructions to display.

1.3.1 Processing Holes

One of the main problems when processing LiDAR data is the presence of holes where geometric information is missing. Such holes can appear for various reasons. First of all, LiDAR sensors perform a discrete acquisition of the world. Also, for some platforms (such as MLS) the sampling density can be highly irregular due to the speed variations and the rotations of the platform. Then, there are many occlusions caused by the fact that LASER beams cannot cross physical objects: if the light reaches a non-transparent object (such as a pedestrian in front of a building), it is reflected and we will have no information on what is behind the object. In order to perform a 3D reconstruction of LiDAR data in urban scenes, we have to tak into account the holes that can appear in the scan.

In the litterature, there exists two main methods for overcoming holes related problems in LiDAR scans: creating a continuous representation of the data or filling identified holes.

1.3.1.1 Creating Continuous Information

3D reconstruction from point clouds can be viewed as finding the interface between interior and exterior of objects. In this case, the reconstruction itself is supposed to represent this interface. In computer science, this interface is usually represented as a surface mesh. The reconstruction is said watertight if it contains no hole: this means that every part of the considered space can be labelled as either *interior* or *exterior*.

When reconstructing 3D urban scenes, some work focus on creating watertight reconstruction by interpolating a continuous reconstruction between the points of the cloud. For instance, Caraffa, Brédif, and Vallet (2016) proposed an algorithm for interpolating the global minimal surface between adjacent points. The minimal surface between two points is a line, and between three or more points, it is a plane. As they can not ensure the presence of a surface between adjacent points in some cases, they use the Dempster-Shafer theory (Shafer, 1992) for associating a level of confidence on the occupancy of the scene (occupied, empty or unknown). There approach allow for reconnecting points that belong to a same object but are separated with holes due to occlusions.

1.3.1.2 Hole-Based Filling

Another approach used in the litterature is to add geometric information in places where data may be missing. This process is called *inpainting*. It consists in identifying missing data and fill this data according to the existing data in its neighborhood. In this section, we focus on *point cloud inpainting*, which consists in adding a set of points with a regular density to fill such areas. This technique originally comes from the image-processing community (Bertalmio et al., 2000). Doria and Radke (2012) propose to use depth-gradient to adapt image-based inpainting algorithms to 3D LiDAR point clouds. Recently, Biasutti et al. (2019) proposed a patch-based method to improve the density of point clouds and fill holes.

We want our reconstruction method not to use any pre-processing step, as we think that they can decrease the geomtric quality of the final reconstruction. This means that we do not add any information that is not already present in the scan and we will

only consider LiDAR data at a local scale. Hence, in this thesis, we will not focus on recovering missing data or producing watertight surface reconstructions.

1.3.2 Reconstruction at different LODs from LiDAR point clouds

Several work focused on the 3D reconstruction task from LiDAR point cloud based on the LoD definitions (Arefi, 2009). For instance, Malambo and Hahn (2010) proposed a new method for reconstructing LoD-1 and LoD-2 buildings from LiDAR point cloud. Bauchet and Lafarge (2019) propose an approach, based on LiDAR point cloud classification for identifying buildings and reconstruct them at a city-scale at LoD-1.

The task of reconstructing buildings from a LiDAR acquisition is still a challenge in the remote sensing community. This is mainly due to the fact that LiDAR sensors provide incomplete informations of the building shape, as seen on Figure 1.5. In most cases, roofs are missing from the acquisition. Also, we have nearly no information on the building shape when it is not directly visible by the sensor. However, some information may be available on the interior of the building thanks to windows and openings. All of this contraints makes it hard to find the precise 3D shape of the building and reconstruct it.

Some researchers decided to focus only on the facade part of the building (Becker and Haala, 2009), which makes sense as it is the part of the building on which we have the most information. Other works propose to combine different point of views (MLS and ALS for instance) for reconstructing full buildings (Caraffa, Brédif, and Vallet, 2016).

Recent works also focus on reconstructing 3D models in a 4-dimensional (4D) space. Ohori et al. (2015) explains that 3D models can be represented at various LoDs along a 4^{th} perpendicular direction. This allows them to handle at the same time different reconstructions with various LoDs at the same time.

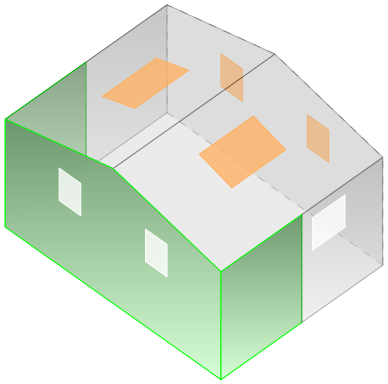


Figure 1.5 – LiDAR acquisition of a building (green) compared to the original building (light grey). Notice that due to openings (in white), a MLS or a TLS is able to retrieve parts of the interior of the building (background wall and roof portions, in orange).

1.3.3 Shannon-Nyquist Theorem

In this thesis, we want to solve the reconstruction problem from LiDAR point clouds, which have an irregular sampling density and for urban scenes, which are typically composed of many objects of varying sizes and shapes. One problem occuring in our case is the presence of small or thin objects such as poles or tree leaves. Such objects have a geometric complexity that is close to or higher than the sampling density of

the scan. This means that the resolution of the scan is not sufficient for reconstructing triangles, which is what traditionnal reconstruction methods aim at. This is illustrated on Figure 3.9. Traditionnal methods tend to either add information (new points in this case) or reconnect thin and geomtrically complex objects to nearby other objects (for instance by creating triangles between the road and a pole).

This problem is a direct consequence of the Shannon-Nyquist Sampling Theorem. This theorem translates in our context as the following statement: if the geometric frequency of the scene is higher than half the sampling frequency, then some signal will be lost. This is exactly what happens for thin objects such as poles: these objects usually appear on a single line of acquisition (see Section 1.1.2.2) of the sensor and it is not possible to interpolate a 3D shape.

In this thesis, as stated in Section 1.2.2, we do not want to add any additionnal information that do not belong to the reality, so we will not use inpainting techniques. We also want to keep the geometric quality and the high accuracy of LiDAR point clouds. In this context, unlike traditionnal reconstruction methods, we choose not to add any triangles between two different unrelated objects (except in areas of contact). This means that there are some cases where we will not be able to reconstruct triangles and that our final reconstruction will not map the whole scene, but only areas where enough information is available, hence the reconstruction will not be watertight.

1.3.4 Topological Reconnections

We now focus on the sensor topology that will be leveraged in this thesis. The goal of this section is to find a workaround for reconstructing physical objects when there is not enough information to derive a 3D shape os such object.

1.3.4.1 Simplicial Complexes

The Shannon-Nyquist Sampling Theorem can be interpreted, as the fact that, due to the resolution of the sensors, there may be parts of the data where we do not have enough information to perform a 3D reconstruction. Thus, in this thesis we propose to reconstruct edges in areas where enough geometric information is available, and triangles when enough edges have been reconstructed. This means that, we aim at reconstructing triangles only when the Shannon condition is met in two dimensions. We will reconstruct only edges when the geometric frequency is too high in 1 dimension and points when the geometric frequency is too high in the 2 dimensions.

As a result, we obtain an object that is composed of points, edges, and triangles. Such object is called a *simplicial complex* (see Section 3.1 for a mathematical definition). Note that simplicial complexes are not watertight and that is coherent with the fact that we do not try to fill holes or add new information, as stated in Section 1.2.2.

1.3.4.2 Vietoris-Rips Complexes

Simplicial complexes are highly based on the local topology of the data. In the LiDAR topology, we say that two pulses are adjacent if they belong to consecutives LASER beams in the same acquisition line, or if they have been emitted in two consecutive acquisition lines, and at a similar LASER angle. Two adjacent pulses can be associated to points far from one another. For instance, one of them can land on a thin foreground object, while the other reached a further object. In this case, we may try to reconnect points that are not connected via a real surface in the scene. Hence, we have to set constraints preventing the connection of such points. An approach to solve

the issue can be to set a maximum diameter for reconstructing a given simplex. Such simplicial complexes are called *Vietoris-Rips complexes*.

A *Vietoris-Rips complex* is a simplicial complex in which all finite subset of points whose diameter is lower than a given threshold forms a simplex (Gromov, 1987; Hausmann, 1995). The diameter of a set of 2 points is the length of the edge connecting them. The diameter of a set of 3 points is the maximum edge length in the triangle connecting them. This can be generalized for simplices of higher dimension. An example of Vietoris-Rips complex is shown on Figure 1.6. Vietoris-Rips complexes have been used as a topological structure for representing 3D point cloud (Chazal et al., 2009; Chambers et al., 2010). They can improve clustering algorithms or reconstruction of point clouds (Beksi and Papanikolopoulos, 2016). For instance, Upadhyay, Wang, and Ekenna (2019) use Vietoris-Rips complexes for reconstructing 3D scenes and automatically finding paths to let a robot exit a maze.

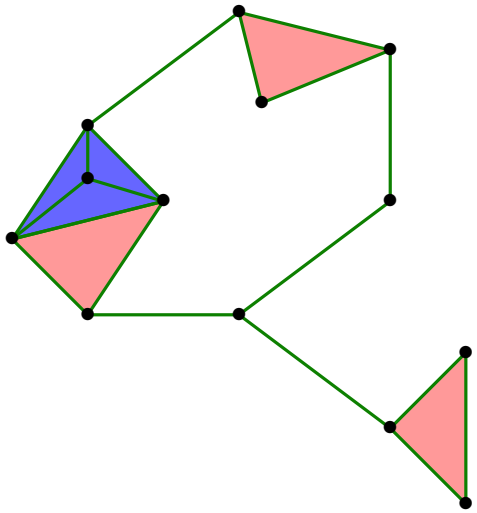


Figure 1.6 – Illustration of a Vietoris-Rips complex. 0D simplices colored are in black, 1D simplices in green, 2D simplices in red and 3D simplices in blue.

Vietoris-Rips complexes are can be employed as a structure for efficiently representing and processing 3D point clouds. However, we argue that they are not adapted when it comes to process LiDAR data. This is mainly due to the highly varying density of LiDAR data: point density is more important for areas close to the sensor than for distant areas. This means that if we set an absolute area threshold for reconstructing Vietoris-Rips complexes, all the points close enough to the sensor will be connected, whereas points in distant areas will remain unconnected. This is why we will use simplicial complexes but not Vietoris-Rips complexes.

1.4 Overview of this Thesis

1.4.1 Problematic

In this thesis, we want to evaluate the suitability of simplicial complexes for representing 3D point clouds. We also investigate their performance as data structure for point cloud generalisation and semantisation.

1.4.2 Organisation of the Thesis

We start by presenting LiDAR sensors, the challenges related to LiDAR data processing and the various applications in which LiDAR sensors are involved in Chapter 2.

The first technical part of this thesis will focus on the reconstruction of simplicial complexes from a local point of view. To achieve this, we use the sensor topology of LiDAR sensors and reconstruct simplicial complexes, first by creating edges between adjacent points, and then by creating triangles when enough edges have been reconstructed. This work is developed in Chapter 3.

As simplicial complexes are created from a very local perspective, they are sensitive to the noise of the acquisition. Thus, we propose an approach for generalising simplicial complexes, separating points, edges and triangles. We then focus on the triangles of our simplicial complexes, as we think that urban scenes are mostly composed of piecewise-planar objects, which can be approximated with a few planar primitives. We propose a global approach for simplifying the representation of large sets of 2D-simplices. This approach is presented in Chapter 4.

We then investigated the use of simplicial complexes as a data structure for 3D reconstructions. We evaluated the influence and the biais introduced by simplicial complexes when reconstructing and simplifying 3D point clouds. We compared our approach to state-of-the-art reconstruction methods and proved that simplicial complexes provide useful information for the 3D reconstruction of urban scenes. This work is detailed in Chapter 5.

In Chapter 6, we propose to improve the classification task by regularizing descriptors at object scale thanks to a pre-segmentation step. We also investigated the influence of simplicial complexes for point cloud classification. We investigate whether the geometric information they provide can help the classification task by providing an additional information to standard 3D decriptors.

2 LiDARs: Sensors and Applications

Contents

In this Chapter, we present the LiDAR sensor and its usages. First, we discuss the origin of the sensor and its composition. Then, we present different types of LiDAR sensors and platforms. Finally, we show various applications for LiDAR data, ranging from forest mapping to autonomous driving.

2.1 LiDAR sensors

2.1.1 History

The first occurence of the word *"lidar"* appears in Middleton and Spilhaus (1953). In their paper, the authors use the term *Light Detection and Ranging* to refer to a ceilometer. A ceilometer [11] is a device that can be used for cloud height estimation. The ceilometer uses light emissions to perform measurments. The light emission can be performed using a LASER, but other sensors can be used as well. In fact, modern ceilometers are a kind of LiDAR sensors.

LiDARs were first designed for satellite tracking. This system is able to compute distances measurements by emitting a signal and measuring the elapsed time between the emission and the returned time of the signal (Stitch, 1961).

The pinciple of the LiDAR sensor has been described by Schawlow and Townes (1958) and Maiman (1960). The LiDAR sensor relies on a LASER. The LASER is a device which emits light, by stimulating the emission of electromagnetic radiations. The light is obtained after a photon stimulation (Einstein, 1917) and is then sent as a beam outside the sensor. First experiments were performed using stimulated microwaves [12] rather than visible light, and they lead to the creation of the Microwave Amplification by Stimulated Emission of Radiation (MASER).

2.1.2 First LiDARs

The first LiDARs were only built at the end of the 1960's (Northend, Honey, and Evans, 1966; Collis, 1969). In the meantime, there were already some works focusing on the usability of LiDAR data in a meteorological context (Fiocco and Smullin, 1963; Fiocco and Grams, 1964). Goyer and Watson (1963) reviews the possible applications of LiDAR data for meteorology, such as studying the internal structure of clouds or for stratospheric measurements (McCormick, 1977). Barrett and Ben-Dov (1967) focused on the evaluation of air pollution in Chicago.

2.1.3 Composition

In this section, we detail the different components of a LiDAR sensor. A simplified scheme of a LiDAR is presented on Figure 2.1.

2.1.3.1 Laser

The LiDAR is an active sensor: it produces and emits the light that is recorded. The light used by the LiDAR comes from a LASER. The LASER technology has been developed after experiments with the MASER (Weber, 1953; Basov and Prokhorov, 1954). The difference between a MASER and a laser is that the first one uses photons from at microwave frequency, whereas the latter uses photons from light (visible or infrared).

11. `https://www.weather.gov/asos/`
12. Microwaves have wavelengths ranging from about one meter to one millimeter.

The first MASER was built by Gordon, Zeiger, and Townes (1954). MASERs have been used in atomic clocks (Ramsey, 1983) and radio telescopes (Jelley and Cooper, 1961; Levy et al., 1986).

The light emitted by the LASER is a *coherent light*. This means that all photons are identical: they have the same level of energy and the same direction (Haken, 1985). The fact that every photon has the same level of energy leads to a monocolor light. The physical *coherence* of the emitted light allows a laser beam to stay narrow, even at great distances. This makes LASERs suitable for performing distance measurements. The frequency of the emitted light usually varies between 500 nm (green) and 1550 nm (infrared, eye-safe), depending on the application and the zone of acquisition. We call *pulse* one emission of light from the laser. This emission usually last a few nanoseconds to a few microseconds. For instance, the laser of the RIEGL VQ-250 [13] has a pulse rate up to 300 kHz, which corresponds to a pulse every 3.33 µs.

2.1.3.2 Mirror

The LASER points toward a rotating mirror. This mirror is used to redirect the emitted light in different directions, thus allowing the sampling of a scene. Between every pulse, the mirror rotates, hence orienting the next pulse to a new direction. The scheme presented in Figure 2.1 shows a sensor with one flat mirror. If the mirror can rotate in the 3 dimensions, then a full scene can be scanned. In fact, most LiDARs are built with a single mirror.

In order to acquire a whole 3D scene, most sensors usually include a mirror driven to rotate along different axis (Koganov, Shuker, and Gordov, 2005; Nguyen et al., 2017; Dong, Anderson, and Barfoot, 2013). Some sensors, such as the RIEGL VQ-250 use a mirror rotating on one axis, while other sensors use mirrors rotating along two orthogonal directions. Another possibility is to use an optical lens, coupled to a polygonal mirror (Niclass et al., 2015). Last, the sampling of a 3D scene can be done by coupling two (or more) lasers and using a single mirror (Muhammad and Lacroix, 2010).

2.1.3.3 Photodetector

The photodetector is a crucial component of a LiDAR sensor. The photodetector is a sensor able to convert light into current (Morrison et al., 1998; Berman et al., 2006). When receiving current from the photodetector, the sensor can acknowledge that the latest pulse reached a physical object that returned the emitted pulse. The most common types of photodetectors used in LiDAR sensors are the photodiodes and the photomultipliers.

2.1.3.4 Other Sensors

2.1.3.4.1 GPS and IMU LiDAR sensors can be coupled to other sensors for complementary informations. For instance, coupling a LiDAR to a GPS will help the registration of the scan (Teo and Huang, 2014; Zheng et al., 2019). Also, for mobile sensors (such as MLSs), the use of a IMU will help the registration as well (Gao et al., 2015; Meng, Wang, and Liu, 2017; Caltagirone et al., 2017).

2.1.3.4.2 Camera Most modern LiDAR sensors do not only include a LASER, but also a RGB-camera (Mirzaei, Kottas, and Roumeliotis, 2012; Gong, Lin, and Liu, 2013),

13. `http://www.riegl.com/uploads/tx_pxpriegldownloads/10_DataSheet_VQ-250_rund_25-09-2012.pdf` - Accessed on 14/01/2020

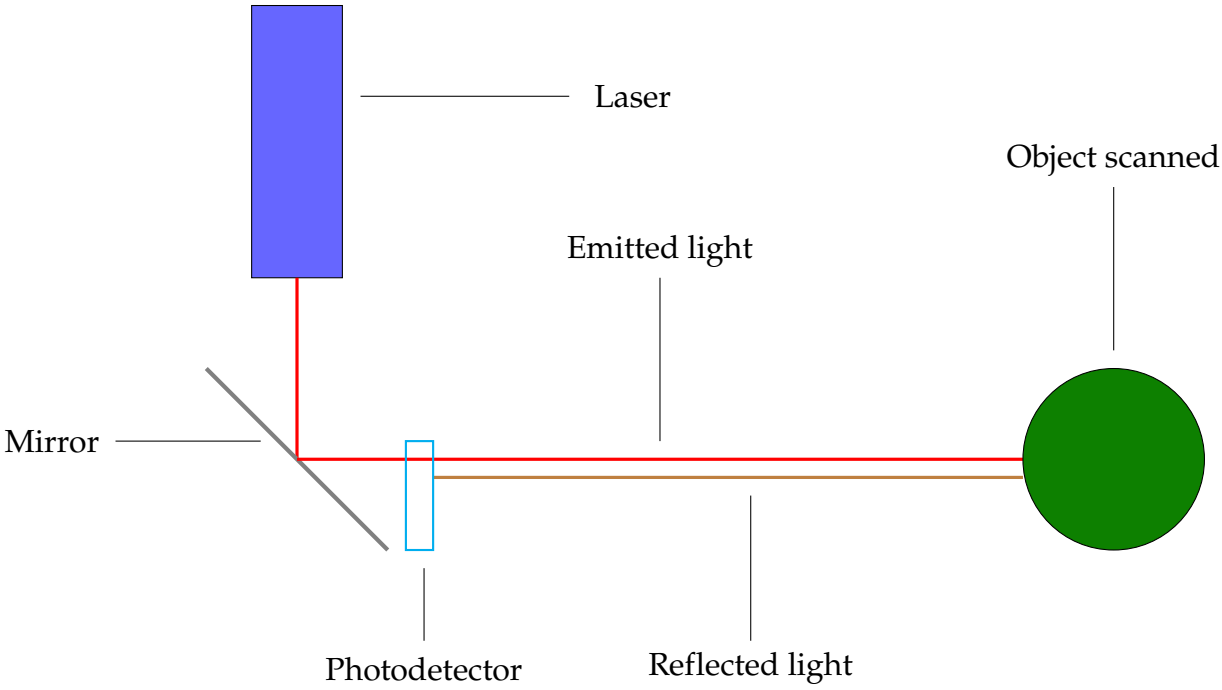


Figure 2.1 – Illustration of a LiDAR sensor.

which provides a useful complementary information to the LASER. The full sensor is usually named a *multi-sensor platform*. The fusion of camera and lidar data has been widely investigated (Zhang, Clarke, and Knoll, 2014; Caltagirone et al., 2019), especially in the context of autonomus driving (Premebida et al., 2014; Cho et al., 2014) where the detection of other cars or pedestrians is overriding. Last, the famous KITTI dataset combine image and laser data (Geiger, Lenz, and Urtasun, 2012).

2.1.4 Retrieved Pulses

We now present how a LiDAR is able to produce a 3D point cloud, based on the time of light emission, the angle of the mirror(s) and the time of activation of photodetectors.

2.1.4.1 Single Echo

To obtain the position where the emitted light reached a physical object, one needs to know: the distance between the sensor and the physical object, and the angle of the laser at the pulse time. The angle can be derived from the current orientation of the mirror. Let c be the speed of light and δt the elapsed time between the light emission and the signal from the photodetector. The distance d between the sensor and the scanned object can be derived via the following equation:

$$d = \frac{c \cdot \delta t}{2} \,. \tag{2.1}$$

We call *echo* the point created when, for a single pulse, the emitted light reached a physical object and was reflected back to the photodetector.

2.1.4.2 Multi Echo

In the previous section, we showed how a single light pulse can be used to create a 3D point. However, when the emitted light reaches an object reflecting only part of

the light (e.g. water), or the edge of an object (e.g. a leaf), part of the emitted light beam is reflected in the sensor direction and the rest of the light beam can continue to propagate until it reaches another object. This is illustrated on figure 2.2. In this case, we observe that a single pulse can have several *echoes*.

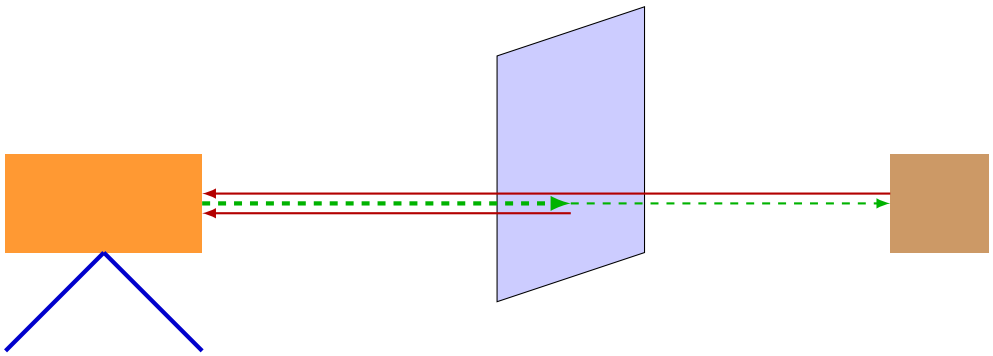


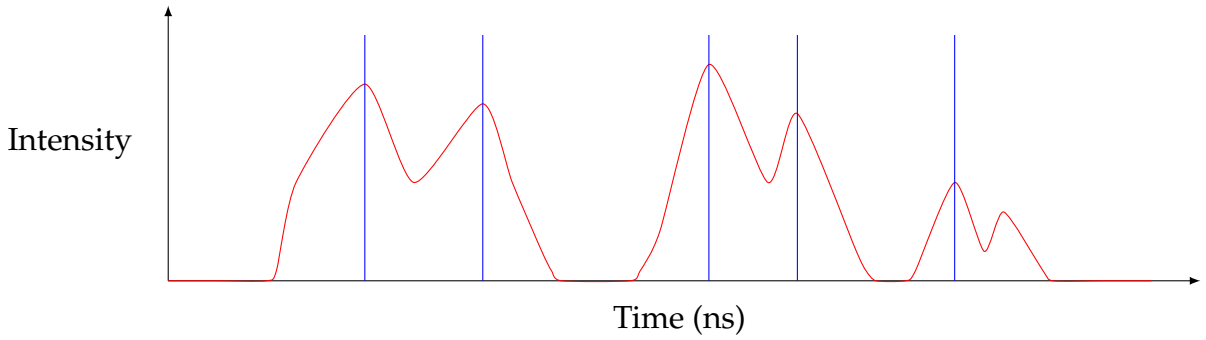
Figure 2.2 – Illustration of the multiple echoes returned for a single light emission. The LiDAR sensor is represented as the orange box, the emitted light in green, the reflected light is in red. First, the emitted light reaches an object that reflects only part of the light: . Then, the light that crossed the first object reaches another object: , before being reflected in the sensor direction.

Following the method introduced in Section 2.1.4.1, we can create several 3D points for a single pulse. To achieve this, for a single pulse, we consider the full returned signal that triggered the photodetector. This returned signal is called a *waveform*. If more than one peak is observed, as shown in Figure 2.3, each separate peak can be considered as a distinct echo. All the points created this way are aligned on the LASER beam trajectory. Taking into account multiple echoes can be crucial, as it gives an information on the scene geometry behind the first echo. This is especially useful in geometrically complex areas such as forests or urban scenes, and help reducing the number of occlusions. However, there is not a single method for interpreting the returned waveform.

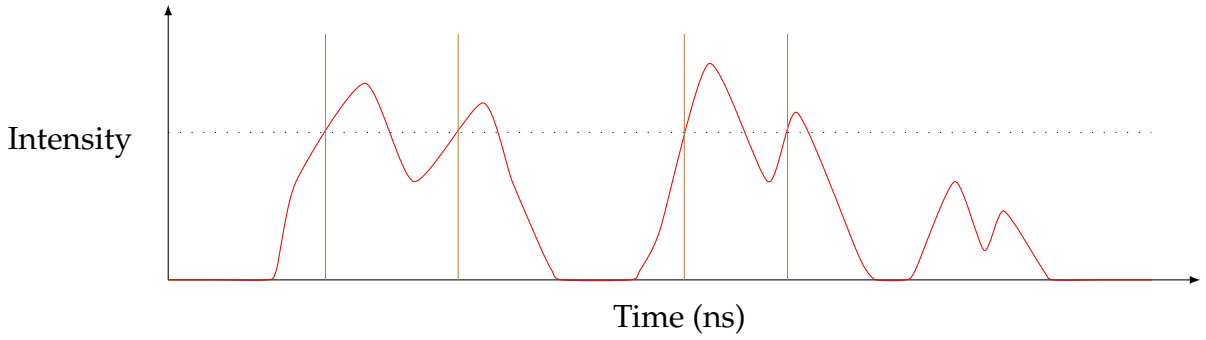
Historically, most LiDAR constructors did not reveal the peak detection method they used. However, the peak detection method chosen can greatly influence the quality of the study (Wagner et al., 2004; Jutzi and Stilla, 2005). Hence, it is interesting to focus on the processing of the returned wavelength itself (Mallet and Bretar, 2009; Chauve et al., 2008). We now focus on two different types of methods for retrieving echoes for LiDAR sensors: *full waveform LiDAR* and *single photon LiDAR*.

2.1.4.3 Full waveform

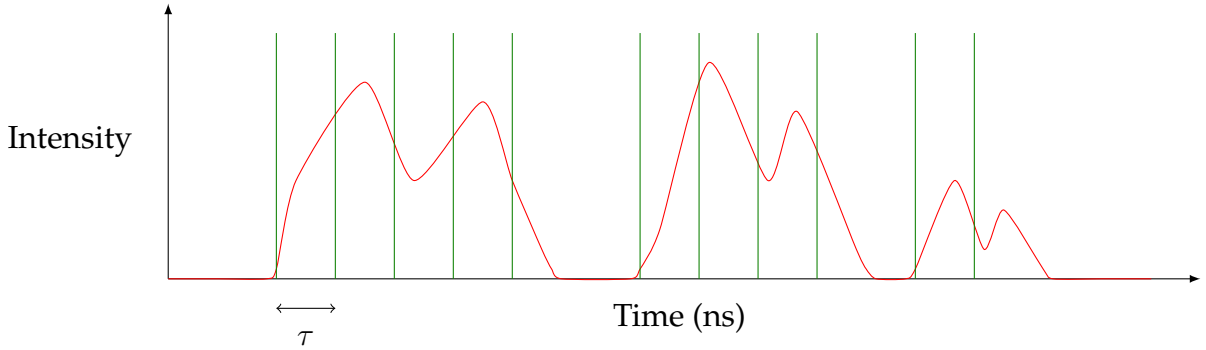
Full waveform LiDARs have been developped at the end of the 1980s (Guenther and Mesick, 1988). However, studying the full waveform became popular only 15 years later (Jutzi and Stilla, 2003; Jutzi and Stilla, 2006). Two different approaches exist for processing the wavelength. The first one consists in decomposing the signal as a sum of echoes (one for each object reached). This method's goal is to detect as much relevant peaks as possible in the waveform. Each peak is used to generate a 3D point. In the end, a dense 3D point cloud is created. The second approach consists in performing a *"spatio-temporal analysis to extract features within a 3D waveform space"* (Mallet and Bretar, 2009). The authors argue that this method is more suitable for processing complex areas such as urban scenes. Mallet and Bretar (2009) show that it is possible to



(a) The blue lines — show the timestamps of peak responses taken for creating points based on the full waveform. If a peak is too close to another one, or if it is not high enough, it can be ignored.



(b) The brown lines —. show the timestamps of peak responses taken for creating points based on the full waveform. The dotted line illustrates the *detection treshold*.



(c) The green lines —. shows the timestamps of peak responses used with a single photon LiDAR. τ is the minimal time between two echoes of a single pulse.

Figure 2.3 – From the same wavelength, displayed in red —, different methods can lead to different echoes created. On Figure a, selected echoes correspond to the highest peaks of the wavelength. On Figure b, selected echoes are obtained after reaching a *detection threshold*. On Figure c, an echo is created everytime a photon is returned, with an elapsed time due to mechanical constraints.

extract *weak pulses* [14] and to improve the distance estimation between physical objects and the sensor by processing the full waveform. For a complete description of both methods and a comparison of their strengths and weaknesses, we refer the reader to Mallet and Bretar (2009).

Full waveform LiDARs have historically been used for mapping large areas such as forests thanks to the precise extraction of multiple echoes for each pulse, as described

14. Weak pulses are pulses originating from object edges or objects that reflects only a small part of the light.

in the previous paragraph. Reitberger et al. (2009) and Heinzel and Koch (2011) used a full waveform ALS for individual tree segmentation and classification. At a larger scale, full waveform LiDARs are suitable for canopy height estimation (Nie et al., 2017; Shen et al., 2018). The study of the full waveform, as described by Mallet and Bretar (2009) strengthen the performances of such sensors in urban areas (Mallet et al., 2011). For instance Azadbakht, Fraser, and Khoshelham (2018) used the *L-Curve* method introduced by Azadbakht, Fraser, and Zhang (2015) to extract their own echoes from the waveform and compute 3D descriptors on them, at different scales. These descriptors are in turn used as input for a Random Forest classifier (Breiman, 2001), which is trained for the pointwise classification of urban scenes.

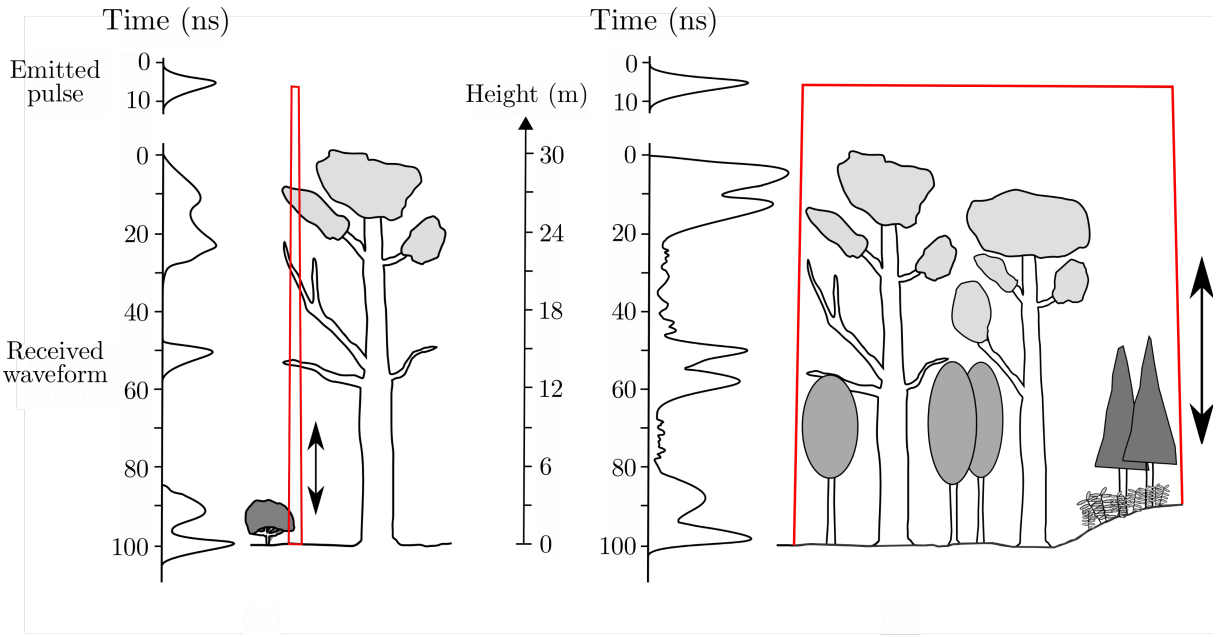


Figure 2.4 – Illustration of the returned wavelength for a scan of trees. The scanned area is displayed in red —. The waveform associated to the scanned area is represented on the graph on the left. The first peak of this graph correspond to tree foliage. If the first peak is used to create an echo, as with methods processing the full waveform, the resulting 3D point will be placed a few centimeter below the actual top of the foliage. This shows the interest of detecting the very first returned photons. This image is taken from Mallet and Bretar (2009).

Lastly, a LiDAR sensor can be equipped with several LASERs. In this case, we call it a *multiple wavelength LiDAR*. Multiple wavelengths LiDARs are interesting for scanning areas where objects have a different response depending on the wavelength of the beam (Sasano and Browell, 1989; Althausen et al., 2000). In this case a 3D point cloud can be built from the peaks of each waveform. In this case, the information that a given peak obtained for a given wavelength shows that the target have some particular physical properties (Woodhouse et al., 2011; Vauhkonen et al., 2013). For instance, (Hakala et al., 2012) built an hyperspectral LiDAR for analysing trees. Multispectral LiDARs have also been used for air-quality analysis: Wirth et al. (2009) investigated the water vapor absorption whereas Murayama et al. (2004) detected dust over Tokyo. This approach was also adopted by the NavVis company for the NavVis M6 indoor mapping sensor [15]: a trolley comprising 6 LASERs which can be used for mapping interiors of buildings as well as small urban scenes.

15. `https://www.navvis.com/m6` - Accessed on 14/01/2020

2.1.4.4 Single Photon

Single photon LiDARs (Aull et al., 2004; Clifton et al., 2015) work differently from full waveform LiDARs. In fact, standard LiDAR systems need approximately 1000 photons to trigger the photodetector. A single photon LiDAR, on the other side, comprises several photodetectors, that can be triggered by a single photon (Priedhorsky, Smith, and Ho, 1996; Degnan et al., 2007). This allows for a higher density of acquisition, which is especially useful in dense and noisy areas such as forests. At the time of writing this thesis, recent single photon LiDARs are able to record several echoes for each photodetector.

Single photon LiDARs are still recent, and most work based on such LiDARs focuses on vegetation processing (Wästlund et al., 2018). In fact, the estimation of canopy height and coverage is of higher quality when using Single photon LiDARs (Swatantran et al., 2016; Tang et al., 2016). This is due to the fact that Single photon LiDARs capture the very moment when the laser beam hits an object. In the case of a forest, this will correspond to the leaf at the top of a tree. This allows for retrieving the full canopy of the forest with a better precision than when using *full waveform* LiDAR as shown on Figure 2.4.

Single photon LiDARs have also been used for bathymetry. In fact, for the same reason as for the canopy height estimation, Single photon LiDARs can be used for water surface mapping (Mandlburger and Jutzi, 2019; Degnan, 2016).

A few studies evaluated the performances of Single photon LiDARs, compared to full waveform LiDAR (Li et al., 2016; Bernard et al., 2019; Mandlburger, Lehner, and Pfeifer, 2019). Theses studies tend to favor *full waveform LiDARs* for most applications, while acknowledging the performances of *Single photon LiDARs* for forestry related purposes. However, the large amount of noise due to the recording technique can be a limiting factor for using Single photon LiDARs.

2.1.5 Different Platforms for LiDAR Sensors

LiDAR sensors can be mounted on different platforms, depending on the usage. The platforms range from a trolley to a plane or a satellite. We present here the main platforms used in the research area and in the industry.

2.1.5.1 Terrestrial Lidar

We start by presenting one of the most used platforms: the TLS. It consists of LiDAR sensor fixed on a tripod. The LASER can be coupled with a camera and a GPS as shown on Figure 2.5. Usually, TLSs are used for scanning a small scene, such as a room or a building. The acquired data can then be used for 3D reconstruction, texturation (if camera acquisition is performed as well) or semantic segmentation (Qi et al., 2017; Landrieu and Simonovsky, 2018). The Semantic3D benchmark (Hackel, Wegner, and Schindler, 2016) and the S3DIS benchmark (Armeni et al., 2016) provide TLS data too.

2.1.5.2 Mobile Lidar

MLSs are used for larger zones than TLSs, and usually in an urban environment. MLS platforms include cars and trolleys (Chung et al., 2017). Usually a GPS and an IMU are coupled to the LiDAR in this case (Toth, 2009; Vlaminck et al., 2016). This allows for simplifying the registration process, which can possibly be done in real time

Figure 2.5 – Illustrattion of a TLS coupled to a GPS. Courtesy of Wikipedia Commons.

(Chen and Cho, 2016; Kim, Chen, and Cho, 2018). For an extensive review of registration algorithms for MLS platforms, we refer the reader to the work of Pomerleau, Colas, and Siegwart (2015).

A few mobile mapping systems were built for research purposes (Kukko et al., 2012; Glennie et al., 2013). The emergence of mobile mapping systems lead to breakthroughs in various domains such as 3D reconstruction and autonomous driving (Tao and Li, 2007; Becker and Haala, 2009; Monnier, Vallet, and Soheilian, 2012). Following these advances, Institut National de l'Information Géographique et Forestière (IGN) built its own mobile mapping system, called Stereopolis (Tournaire, Soheilian, and Paparoditis, 2006; Paparoditis et al., 2012). Stereopolis is a mobile multiplatform sensor. It contains: 16 cameras (10 for a panoramic head and two triplets of cameras, one at the front and one at the back of the vehicle), 3 LiDARs (1 on each side for facade acquisition and another one for the bottom part of the scene) and a set of navigation devices (two GPSs and an IMU). Stereopolis has successfully been used in an urban context for pedestrian detection (Paparoditis et al., 2012) or road side detection (Hervieu and Soheilian, 2013). Last, some studies focused on inpainting for completing missing data in Stereopolis acquisitions (Biasutti et al., 2017). This sensor's acquisitions are used throughout this whole thesis for various experiments.

2.1.5.3 Aerial Lidar

When it comes to mapping vast areas, ALSs are the best solution: they allow to perform fast acquisitions of large areas, with a sampling ratio of several points per square meters, enabling various applications, from Digital Elevation Model (DEM) generation (Fissore and Pirotti, 2019) to forest inventory (Hyyppä et al., 2017). Alamús et al. (2018) used an ALS over the city of Barcelona for evaluating the light pollution and investigation solar panels installastion.

Aerial platforms are also used for underwater analysis. In this case, a special LiDAR is equipped: a bathymetric LiDAR (Quadros, Collier, and Fraser, 2008; Kim et al.,

2016). In this context, a bathymetric LiDAR allows to detect the bottom of water. This has been used for mapping riverbeds (Wang and Philpot, 2007; Lague et al., 2016). Collin et al. (2018) focused on coral detection and extraction. However, bathymetric LiDARs have not been tested in this thesis, and we refer the reader to the work of Kim et al. (2019) for more details on this technology.

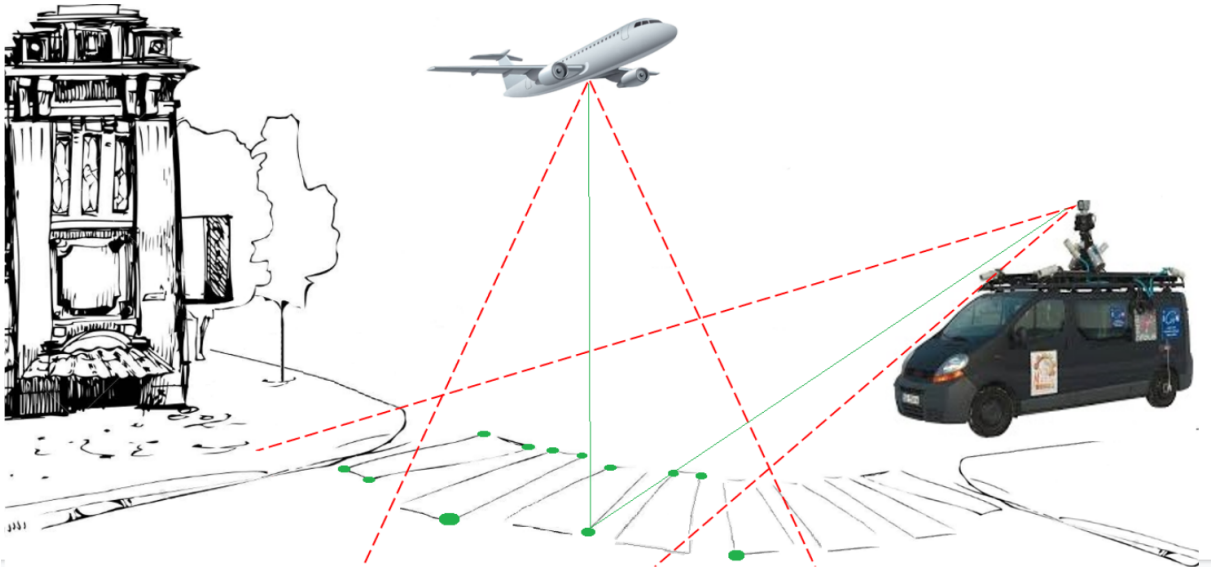


Figure 2.6 – Illustration of an ALS and a MLS acquisition on an urban scene. Courtesy of Imane Fikri (IGN-LaSTIG-ACTE).

2.1.5.4 UAV-based Lidar

ALSs are one of the most used sensors for mapping large areas or when a top-down point of view is needed. However it can be rather expensive and time-consuming to set-up a full ALS system. ALSs also suffer from occlusions (due to height differences in the scene). Hence, there has been a need to develop a simpler and lighter system that could be used to map areas unaccessible for classic TLS and MLS systems while having the possibility to map an entire complex scene. In this context UAV-based LiDARs systems have been developed (Lin, Hyyppa, and Jaakkola, 2010; Wallace et al., 2012).

UAV-based LiDARs have been used to map forests below the canopy (Chisholm et al., 2013; Wallace, Lucieer, and Watson, 2014). In fact, UAVs are especially interesting to use in forested areas due to the accessibility difficulties and the complexity of the scenes. They have been used in various forests, from the Amazon rainforest (Khan, Aragão, and Iriarte, 2017) to Chinese forests (Guo et al., 2017). As for many other LiDAR-based systems, UAV-based LiDAR surveys can be completed by imagery (Sankey et al., 2017; Sankey et al., 2018).

Although we presented various platforms in this section, this list is not comprehensive, as there exists many very specific LiDAR-based platforms have been created for a limited number of applications. In this thesis, we will mostly focus on MLS sensors. However, some experiments were done on ALS and TLS too, for generalization purposes.

Figure 2.7 – Illustrattion of a UAV-based LiDAR sensor. Courtesy of Wikipedia Commons.

2.2 Applications

We now present some applications of LiDAR data. We first focus on meteorology as it is the historical field of interest for processing LiDAR data. Next, we demonstrate that LiDAR devices have been useful for environmentally-related challenges, such as biomass estimation (see Section 2.2.2) or glacier monitoring (see Section 2.2.3). We then present some applications in cultural heritage, a fast-growing area of research at the time of writing this thesis. Lastly, we present a brief state-of-the-art on applications relating to autonomous driving and city mapping, two areas of interest related to this thesis in Sections 2.2.5 and 2.2.6.

2.2.1 Meteorology and Space-Based Observations

The first application we present in this section is the historical area of interest for LiDARs: meteorology and space-based observations. One of the main challenges in meteorology is tracking moving objects (such as clouds) and estimating their speed and direction. This allows for predicting wind-related events such as tornadoes. To this end, a special type of LiDAR has been developped: the *Doppler LiDAR* (Chanin et al., 1989; Grund et al., 2001). The development of this sensor is motivated by the fact that moving objects add a Doppler shift to the photons of the LASER beam [16] (Blaugrund, 1966; Takeda, 1986). This shift depends on the speed and the direction of the moving object. A Doppler LiDAR is able to analyse this shift and compute the direction and shift of moving objects such as clouds. Such sensor can then been used to compute wind gusts (Suomi et al., 2017) and evaluate wind energy (Goit, Shimada, and Kogaki, 2019; Henderson and Jacob, 2018). LiDAR sensors have also been used for studying water vapor (Wulfmeyer and Bösenberg, 1998), cloud radiative effects (Guélis et al., 2017), and cosmic dust influx in the atmosphere (Gardner et al., 2014), improving our knowledge of the different atmospheric layers of the Earth.

The second historical field of interest for LiDAR sensor is space-based observations. To this end, a few sensors have been developed, especially by NASA's teams (McCormick et al., 1993; Winker, Couch, and McCormick, 1996). One of the most recent ones is the CALIOP sensor (Winker et al., 2009). This sensor has been mounted on the CALIPSO satellite [17]. Its goal is to perform vertical measurements (in Earth di-

16. It is possible to measure this Doppler shift because the light emitted by a LASER is *coherent*.

17. This satellite was built by French and American space agencies. It comprises a LiDAR, a camera and a radiometer. CALIPSO was launched in 2006 and is still active. For more informations about the project, see: `https://www-calipso.larc.nasa.gov/`

rection) to study aerosols and clouds. Space-based LiDAR studies include wind measurements (Baker et al., 1995; Marseille and Stoffelen, 2003) or sea surface estimation (Hu et al., 2008). Space-borne LiDARs have also been used for phytoplankton biomass estimation (Behrenfeld et al., 2017; Hostetler et al., 2018). In the following sections, we present environment-based applications using LiDAR data.

2.2.2 Forestry

LiDAR sensors are increasingly used for the study of forests. This comes from the fact that, unlike for cameras, LiDARs are able to penetrate through vegetation (Lim et al., 2003; Simard et al., 2011). Hence, LiDARs have been used for biomass estimation (Zolkos, Goetz, and Dubayah, 2013; Luo et al., 2017; Holm, Nelson, and Ståhl, 2017) in various types of forest, ranging from boreal (Askne, Soja, and Ulander, 2017) to tropical forests (Lau et al., 2019). These sensors are also suitable for forest inventory (Bauwens et al., 2016; Hyyppä et al., 2017). Some works also focus on terrain detection in forests, for Digital Terrain Model (DTM) estimation (Bigdeli, Amirkolaee, and Pahlavani, 2018; Maguya, Junttila, and Kauranne, 2014) or thalweg detection (Guilbert, Jutras, and Badard, 2018).

Most works using LiDAR data on forest areas rely on acquisitions from ALS sensors (Richardson and Moskal, 2011; Bock et al., 2017). ALS is preferred in this context as it allows for performing a fast acquisition of the forest. Moreover, it can be used to scan unaccessible or remote areas. However, most of the emitted pulses cannot cross the canopy, which results in an incomplete acquisition of the forest's structure below the canopy. Hence, some researchers recently proposed to use UAV-based LiDAR for mapping forests (Wallace et al., 2012; Liu et al., 2018). In this context, UAVs can be used to scan the interior of the forest, below the canopy, providing complementary information to ALS.

Last, an interesting application of LiDAR sensors in a forestry context is for the evaluation of natural disasters, such as wildfires or floodings. In fact, LiDARs have been used during the fires in the Amazon rainforest in 2019,[18] and might see an increasing use in areas where such disasters are frequent.[19] In this context, an increasing number of scientists are working on fire detection and evolution on forested areas (De Almeida et al., 2016; McCarley et al., 2017; DeLong et al., 2018), using a LiDAR to survey burning areas. Similar works have been conducted for flooded areas (Malinowski et al., 2016).

2.2.3 Geology

Another environment-related application for LiDAR data is geology (Hartzell et al., 2014; Buckley et al., 2008). In fact, LiDAR sensors can be used to monitor large objects such as glaciers (Fischer et al., 2015; Putkinen et al., 2017) or even tectonic plates (Bruhn et al., 2006; Meigs, 2013). LiDAR sensors have been useful for mapping and monitoring volcanoes (Csatho et al., 2008; Kereszturi et al., 2018). In this context, Behncke et al. (2016) used LiDAR data to estimate the eruptive activity of Mt. Etna (Italy) by regularly producing DEMs of the volcano's slopes. They use this data to compute the volumetric evolution of the volcano and estimate the quantity of lava at the summit of the volcano.

18. https://news.mongabay.com/2019/12/2019-the-year-rainforests-burned/ - Accessed on 15/01/2020

19. https://magviral.com/more-than-600-new-jobs-for-firefighters-proposed-for-california-the-ukiah-daily-journal/ - Accessed on 15/01/2020

At a larger scale, LiDAR sensors have been used to monitor natural disasters (Dunham et al., 2017) such as earthquakes. Nissen et al. (2017) studies the coastal deformation attributed to Kaikōura earthquake (New Zealand). Ishimura et al. (2019) used a LiDAR to evaluate the surface rupture traces created by the Nagano earthquake (Japan). Last, Bose et al. (2016) evaluate building damages after an earthquake in Nepal. This shows that even in the case of a natural disaster, LiDAR sensors can be used promptly and at both local and global scales to monitor and evaluate the disaster. They can also be used for monitoring the Earth and prepare to future disasters such as volcanic eruptions.

2.2.4 Cultural Heritage

Another application for LiDAR data developped here, is cultural heritage. In fact, LiDAR sensors have been used to survey large areas in order to find ancient cities or roads (Rodríguez-Gonzálvez et al., 2017; Cheng et al., 2016; Chase, Chase, and Chase, 2017). For instance, Inomata et al. (2018) used an ALS for identifying Mayan cities and evaluate their evolution through time in the Seibal region in Guatemala. This study was part of the *"Uxul Archaeological Project"* (Grube et al., 2012; Vincent et al., 2015), a project aiming to studying the Mayan civilization by using photogrammetric and lasergrammetic means. Johnson and Ouimet (2014) used LiDAR data to detect archaeological sites in New England, a region in the northeastern part of the USA. The authors were able to retrieve old habitations and crops from Native Americans.

LiDAR data can be used to recreate full 3D ancient cities, which in turns can be used for interactive visualization (Abdelmonem, 2017). Such information can be used in a historical map, displaying the changes through centuries of a geographical region. This is, for instance, the scope of the *European Time Machine* [20] project: a European project for creating a historical map of all Europe. A first experiment was done in the single city of Venice [21]. In the following sections we focus on applications more related to the topic of this thesis.

2.2.5 City Mapping

The ability to acquire large areas in a relatively short period of time, while maintaining a high geometrical precision implies that LiDAR sensors are suited for city-scale analysis (Zhou, 2012; Wang et al., 2019). This comprises urban planning, or city visualization. In this context *Digital City Twins* have been developed (Knight, Rampi, and Host, 2017). A Digital City Twin is a digital model of a city. Examples of Digital City Twins include Singapore. [22] This model is based on a large-scale LiDAR acquisition, completed with a collaborative platform. Digital City Twins often include various and complementary data sources: MLSs are good for street and facade reconstructions, while aerial sensors are better for roof extraction (Javanmardi et al., 2017). Moreover, the combination of images and LiDAR point clouds is important for the texturation of 3D models (Boussaha et al., 2018; Meinhardt-Llopis and d'Autume, 2019). Digital City Twins allow for various applications, from flood simulations to wind energy estimation (Millward-Hopkins et al., 2013; Adam et al., 2016). In this case, LiDAR data is used as input for reconstructing urban corridors favorable for wind movements.

20. `https://www.timemachine.eu/discover/` - Accessed on 15/01/2020

21. `https://www.epfl.ch/research/domains/venice-time-machine/inbrief/` - Accessed on 07/04/2020.

22. `https://www.nrf.gov.sg/programmes/virtual-singapore` - Accessed on 16/01/2020

LiDAR sensors have been used for tree identification and management at city scale (Alonzo, Bookhagen, and Roberts, 2014; Liu et al., 2017). For instance, the city of Washington D.C. used a LiDAR to count the trees in the city.[23] In this context, a LiDAR can be combined with hyperspectral imagery for better accuracy (Alonzo et al., 2016). For this application, most works used an ALS, as it allows a rapid acquisition and can easily be coupled with aerial and satellite imagery (Guo et al., 2011; Dogon-Yaro et al., 2016).

ALSs have also been used for roof extraction in urban areas. The extracted roofs can in turn be used for DEM production (Fissore and Pirotti, 2019) or investigation of solar panels installation (Lee and Zlatanova, 2009; Gooding, Crook, and Tomlin, 2015). LiDARs data help the cadastral mapping at a city-scale, by identifying the footprints of buildings and the number of floors (Giannaka, Dimopoulou, and Georgopoulos, 2014). For instance, Ribeiro et al. (2019) used an ALS over Sao Paulo City (Brazil) to identify *favelas* and estimate their size. Cadastral mapping relies on single building extraction. Building extraction can be performed by detecting the edges of each building (Wei, 2008) or by facade detection (Dorninger and Pfeifer, 2008).

City mapping often requires to identify each object of the city, as typical objects, such as roads or buildings, will not be reconstructed in the same way. Hence, some works focus on the semantic segmentation of LiDAR data at city-scale (Dohan, Matejek, and Funkhouser, 2015; Landrieu and Simonovsky, 2018; Tchapmi et al., 2017). These segmentations can be used as input for 3D reconstruction algorithms (Lin et al., 2013).

2.2.6 Autonomous Driving

One of the growing challenges in urban areas is autonomous driving (Wei et al., 2013). Indeed, self-driving cars could increase the safety for drivers, pedestrians, or cyclists. Moreover, it could improve the global traffic and reduce traffic jams (Franke et al., 1998; Schellekens, 2015). Self-driving cars would be useful for disabled or old people (Yang and Coughlin, 2014), unable to drive, by offering them a higher independence. Last, driverless trucks could be used to reduce transportation costs and delivery times.

Self-driving cars can include (among many sensors) a LiDAR. This LiDAR is used for real-time analysis of the car environment. Its goal is to detect objects defining driving rules comprising traffic signs (Gargoum et al., 2017) and road lanes (Zhang, 2010; Zhu et al., 2012; Ghallabi et al., 2018). Autonomous cars should also be able to detect moving objects such as cyclists or pedestrians (Kidono et al., 2011; Matti, Ekenel, and Thiran, 2017). Futhermore, driverless cars have to detect and avoid potential accidents. This implies that the car must know the place and speed of other cars on the road (Ibisch et al., 2013; Asvadi et al., 2017). All these expectations shows how crucial is fast and accurate interpretation of the urban environment. The large quantity of information acquired by a LiDAR device compared to the processing efficiency required in this context shows that being able to simplify the acquired scene while losing as little geometric information as possible is an important subject.

23. `https://www.govtech.com/fs/news/Lidar-Data-Is-Becoming-an-Increasingly-Valuable-Tool-for-Cities.html` - Accessed on 16/01/2020

2.3 Conclusion

In this chapter, we presented an overview of the LiDAR technology. This sensor relies on the LASER technology to emit a signal whose return triggers a photodetector. This produces a *return wave* which can be used to derive the position of physical objects encountered by a single emitted signal. LiDAR sensors can be combined with cameras and other sensors such as a GPS or an IMU to complete the acquisition and help the global registration of a scan. LiDARs can be mounted on various platforms, from planes to trolleys, allowing the use of such sensor in nearly all environments.

We then presented different applications of this technology and showed that LiDAR sensors were used at many different scales (from a local building to entire geographic region) and for various usages, such as fire monitoring or autonomous driving. This shows that LiDAR sensors are more and more used and help face many technological challenges, from archaeological studies to wind measurements. We want to emphasize on the fact that there is not a *single* LiDAR type or a *single* platform that outperforms all others. The choice of the sensor and the platform is highly dependent on the application and the budget (it is cheaper to use a single TLS than to launch a satellite equipped with a LiDAR into space).

For the remaining of this thesis, we will mostly use data from MLSs, but will also evaluate some of our algorithms on ALS and TLS data. In the next chapter, we will show how raw data from a MLS can be used for the reconstruction of simplicial complexes.

3 Simplicial Complexes Reconstruction from 3D LiDAR Data

Contents

In this chapter we introduce the notion of simplicial complexes and explain why they are an approximate structure to represent the continuous nature of a scene scanned by a LiDAR. Then, we design a method for simplicial complexe reconstruction from a LiDAR point cloud and demonstrate its versatility by applying it to MLS and TLS datasets. We argue that the use of simplicial complexes allow us to be more adaptive than triangulated mesh reconstructions, especially on complex parts of the scene, such as trees or urban furniture.

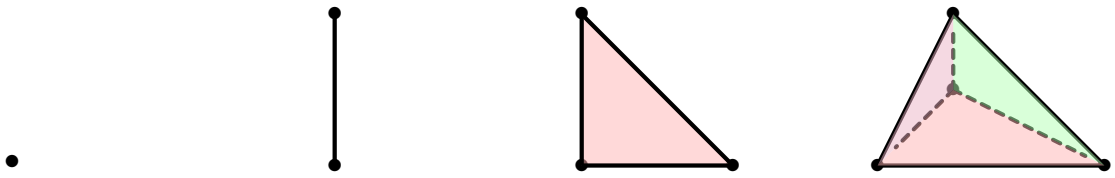
3.1 Simplicial Complexes

In this section, we first introduce the notion of simplex, on which simplicial complexes are built. We then present some works on 3D geometry processing based on simplicial complexes.

3.1.1 Definition

3.1.1.1 Simplex

For $i \in [0, \dots, n]$, we define a i-simplex of $\mathbb{R}^n$ as the convex hull of $i+1$ independent points of $\mathbb{R}^n$. We define a face of a i-simplex of $\mathbb{R}^n$ as the convexhull of j independent points ($j \in [1, \dots, i]$) from the original i-simplex. Examples of simplices are shown on figure 3.1. In this thesis, we focus on reconstructing the exposed surface of the scene and not the full 3D scene. Thus we will focus on 0-, 1- and 2D-simplices (points, line segments and triangles).



(a) Simplex of dimension 0.

(b) Simplex of dimension 1

(c) Simplex of dimension 2

(d) Simplex of dimension 3

Figure 3.1 – Examples of simplices in low-dimensional spaces

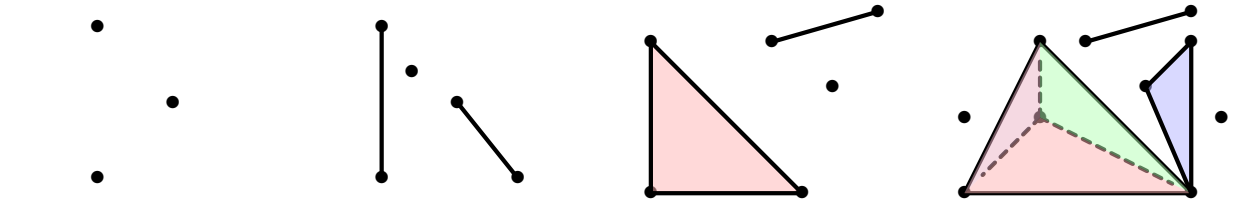
3.1.1.2 Simplicial Complex

For $i \in [0, \dots, n]$, we define a i-simplicial complex of $\mathbb{R}^n$ as a set of simplices which dimension is lower than i. All the simplices composing a simplicial complex do not necessarily have the same dimension. A i-simplicial complex satisfies the following:

— all the faces of a i-simplicial complex belong to itself,

— the intersection of two simplices of a i-simplicial complex is either a shared face of each simplex, or $\varnothing$.

The dimension of a simplicial complex is the maximum dimension of its simplices. Examples of simplicial complexes are shown of figure 3.2. Note that a simplex is a simplicial complex too. For the same reason as mentionned in the previous paragraph, we will only consider simplicial complexes comprised of $0-2$-simplices.



(a) Simplicial complex of dimension 0.

(b) Simplicial complex of dimension 1.

(c) Simplicial complex of dimension 2.

(d) Simplicial complex of dimension 3.

Figure 3.2 – Examples of simplicial complexes in low-dimensional spaces

3.1.2 Early Uses with 3D Data

The use of simplicial complexes for 3D point cloud reconstruction has been expressed by Popović and Hoppe (1997) as a generalization manner to simplify 3D meshes. In this paper, the authors use an alternative set of vertex unifications and splits to generalize an input mesh. The core idea of their approach has been introduced in an earlier paper (Hoppe, 1996). In this work, they introduced a method for simplifying meshes with the edge-collapse technique. They extended the principle to simplicial complexes, by arguing that the generalization process may require some topological changes: in some cases the simplicification process of a surface mesh can be stopped due to the lack of triangles in the final output; however, one can allow the collapse of triangles in such case and keep only an edge instead of the triangle, or even just a point. This produces a simplicial complex. Guibas and Oudot (2008) used witness complexes (a simplicial complex whose faces are “witnessed" by a set of points) to generalize meshes.

Simplicial complexes are also used to simplify defect-laden point sets as a way to be robust to noise and outliers using optimal transport (De Goes et al., 2011; Digne et al., 2014) or alpha-shapes (Bernardini and Bajaj, 1997). Applications of simplicial complex reconstruction include forest canopy reconstruction Vauhkonen (2015) or road network reconstruction from GPS traces (Ahmed and Wenk, 2012; Dey, Wang, and Wang, 2017). Simplicial complexes have also been used for computing trajectory to help robots move from one place to another (Pokorny, Hawasly, and Ramamoorthy, 2016).

In this thesis, we are looking for a 3D structure that is adaptive to the geometry of the scene, while being independent from the sampling or any other additional information. Simplicial complexes can fill this role: planar surfaces (roads, facades, ...) will be represented with sets of triangles, while linear structures (poles, wires, small branches) will be displayed as sets of edges. Lastly, geometry will not be interpolated in complex areas of the scene, such as tree foliage. In this case, the input points will simply be preserved in the output simplicial complex.

3.2 Simplicial Complex Reconstruction from 3D LiDAR Data

In this section, we introduce the reconstruction approach that we used. We want our approach to connect points that belong to a same object in the original scene. This will allow us to select, at the point scale, which simplex should be added to the reconstruction. Ideally, each edge will encode the existence of an object between the two echoes it connects. This principle can be extended to triangles as well.

We first present the different structures for 3D LiDAR point cloud and explain why

simplicial complexes fit our need. Then we show the rconstruction process for each dimension: we start by adding simplices of dimension 1 to our reconstruction and then add simplices of dimension 2. Lastly, we present a quick hole filling process to recover missing simplices.

3.2.1 Structure of 3D LiDAR Data

Unlike images, one of the biggest problems of 3D point clouds is the absence of a trivial connectivity structure. Images rely on pixel adjacency to derive the Von Neumann's or Moore's neighborhood (Toffoli and Margolus, 1987, Section 7.2) as shown on Figure 3.3. These 2D neighborhood definitions can be extended to 3D by connecting a point to the closest points in the 3 dimensions (Monica and Aleotti, 2018). Point clouds are highly irregular and noisy. Point clouds don't have the array structure of images. This implies that we cannot directly apply the same neighborhood than for images.

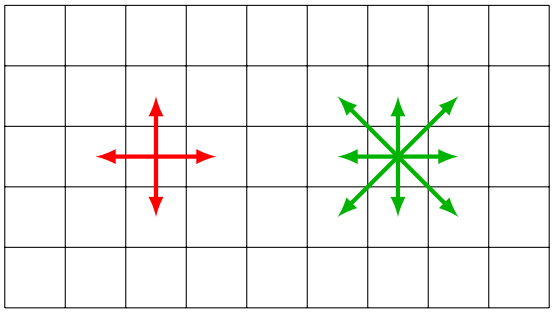


Figure 3.3 – Illustration of Von Neumann's and Moore's neighborhoods (respectively red and green) on an image. Each cell represents a pixel.

We now present the different kinds of structures that can be applied to 3D data, from depth images to LiDAR data.

3.2.1.1 RGB-D images

A first step toward the structuration of 3D point clouds is to add a third dimension to an image: a depth information is added to each pixel. This structure has been used for camera pose estimation (Shotton et al., 2013; Cavallari et al., 2019) or 3D scene reconstruction (Zollhöfer et al., 2014; Steinbrucker, Kerl, and Cremers, 2013). However, LiDAR data usually has a highly varying density which makes it hard to sample as an image. Moreover, for a single laser pulse, several echoes can be returned (e.g. when the emitted ligth encounters a window, part of the light is refracted, but the rest crosses the window and can hit an object behind, resulting in two returned echoes for a single emitted pulse as shown on Figure 2.2). The fact that LiDAR data can comprise several echoes for a single pulse means, if interpreted as an RGB-D image, that there can be several points with a significant depth difference on the same pixel.

3.2.1.2 Projection on 2D images

One can derive images from 3D point clouds by taking artificial photos of the scene (Boulch, Le Saux, and Audebert, 2017). This way, we can directly apply images-based structure and algorithms. This is useful for images relocalization in point clouds, with applications in historical images positioning (Russell et al., 2011; Aubry, Russell, and Sivic, 2014). However, this method suffers from the same problems than RGB-D images. Boulch, Le Saux, and Audebert (2017) try to overcome this drawback by using

different point-of-views for generating images, but without strong knowledge on the data, we have no guarantee to process all the points of the cloud.

3.2.1.3 k-nearest neighbors

A naive extension of the image structure to 3D point clouds is to look, for each point, to its k-nearest neighbors. This way, we consider a fixed number of neighbors for each point. However, this structure is highly dependent of the density of the scan. If there are some strong density differences in different directions (like for MLS or ALS), this structure will be heavily biaised, as showned on figure 3.4. Moreover, it is not easy to find the optimal neighboring size. Some works (Gressin et al., 2013; Weinmann, Jutzi, and Mallet, 2014; Weinmann et al., 2015b) suggest to use an adaptive threshold, related to the local geometry of the neighborhood.

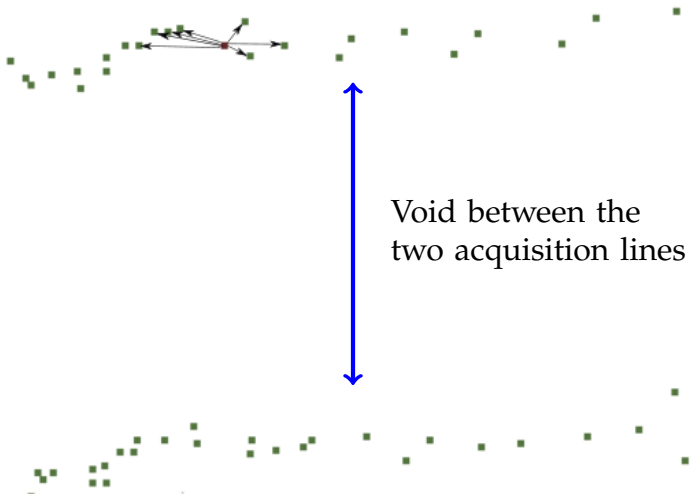


Figure 3.4 – Illustration of the biais induced by a KNN on a point cloud from MLS. The scene is a zoom on the corner of a facade. The red points is connected to its 7 nearest neighbors. The closest neighbors of the red point are nearly aligned. This lead the neighborhood to be mostly linear, even if the point cloud shown here is part of a planar facade.

3.2.1.4 Geometric neighborhoods

One can look for neighborhoods with fixed geometrical shapes (usually spheres or cylinder) to create a neighborhood relationship (Demantke et al., 2011; Blomley and Weinmann, 2017). An example of a spherical neighborhood on a point is displayed on Figure 3.5. Nevertheless, like for KNN, it is hard to parameterize and can be sensitive to highly varying point density as shown on Figure 3.4.

3.2.1.5 Delaunay triangulation

In order to avoid the problems of highly varying sampling density, we can structure point clouds with a 3D Delaunay triangulation (Delaunay, 1934). It was first defined for 2D point clouds. Given a set of points, its Delaunay Triangualtion tries to maximize the minimum angle of each triangle. A Delaunay triangulation ensures that there are no points in the circumcircle of any triangle of the triangulation. Such triangulation is very useful when processing data with irregular density. Boissonat (1984) adapted

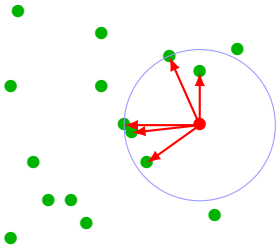
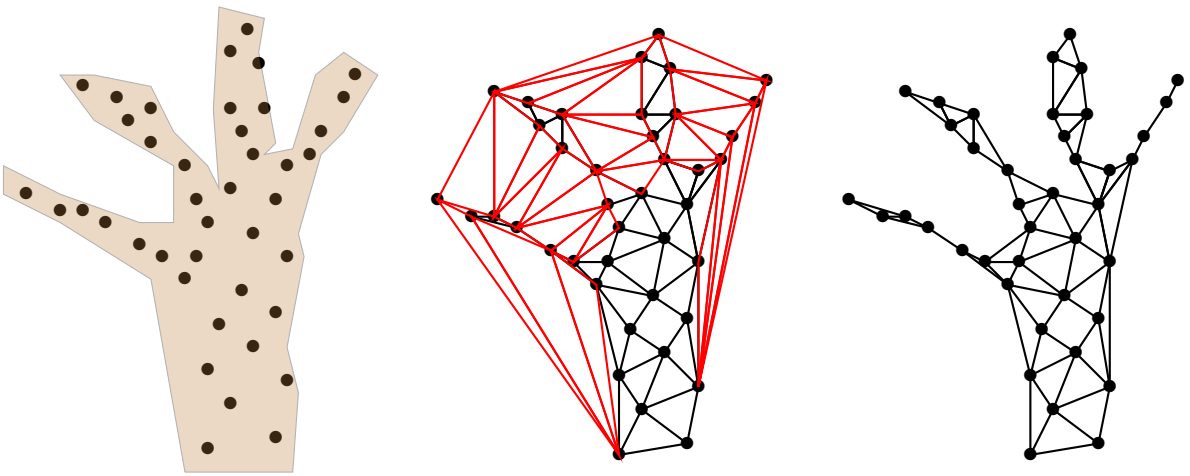


Figure 3.5 – Illustration of the geometric neighborhood. Here we represent a point cloud with the spherical neighborhood of a single point (in red).

the original 2D algorithm for 3D triangulations. It has been extensively used in the literature (Golias and Dutton, 1997; Du and Wang, 2006; Romanoni et al., 2016). Some work also focused on Delaunay tetrahedrization to reconstruct 3D simplices (Si, 2015). This tetrahedrization works similarly to the Delaunay triangulation, but generalized to tetrahedrons and spheres. Even if this structure is computationally efficient with geometrically simple areas, such as buildings or roads, it is not precise enough on more complicated areas, like tree foliage, as shown on Figure 3.6b. On this figure, representing the top of a tree with a few branches, we can remark that points at the end of the branches are linked to points of the trunk, even if we know that they cannot be directly connected in the real scene. Another example of a reconstruction of a tree that do not respect our criterion is shown on Figure 3.7. In this thesis, we do not want to reconstruct such triangles. Instead we want to create simplices between points only when they are directly connected in the original scene.



(a) Scan of the top of a tree. For simplicity we removed the foliage and only kept the top of the trunk and some branches.

(b) Illustration of a Delaunay triangulation on the scan of the top of a tree. The red triangles show an examples of triangles that we want to avoid because they connect points that are not directly connected in the original tree.

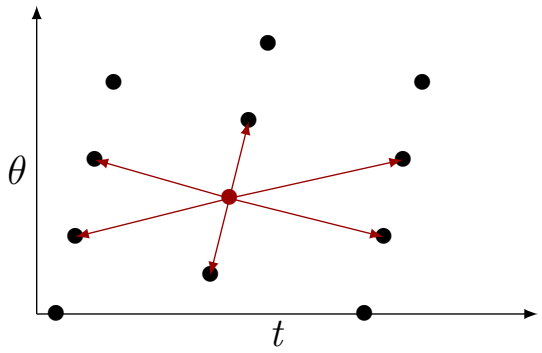
(c) Illustration of the neighborhood we want to build: the main branches are separated and when there is not enough points to add triangles, we only add edges.

Figure 3.6 – Illustration of the differences between a Delaunay triangulation and the neighborhood definition we want to build.

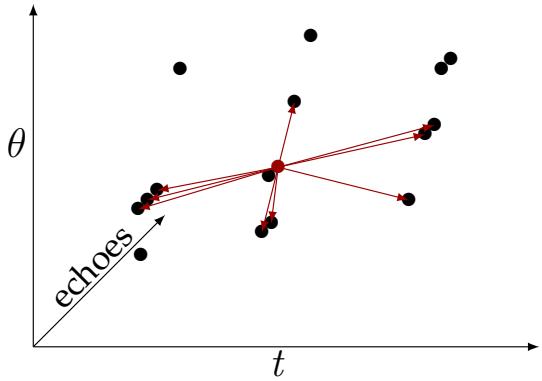
Figure 3.7 – A zoom on Google Maps' reconstruction of a tree. In this reconstruction, the different branches of the tree are connected together in order to form a simpler polygon which do not represent the original geometry of the scene.

3.2.1.6 Sensor topology

The sensors used for MLS or TLS often have an inherent topology. These sensors sample a regular grid in (θ,t), where θ is the rotation angle of the laser beam and t the time of acquisition. In general, the number N_p of pulses for a 2π rotation in θ is not an integer, so a pulse P_i has six neighbors: P_{i-1}, P_{i+1}, P_{i-n}, P_{i-n-1}, P_{i+n}, P_{i+n+1}, where $n = \lfloor Np \rfloor$ the integer part of the number of pulses per line. This illustrated on Figure 3.8a. However, this topology concerns emitted pulses, not recorded echoes. One pulse might have 0 echo (no target hit) or up to 8 as most modern scanners can record multiple echoes for one pulse if the laser beam intersected several targets, which is very frequent in the vegetation or transparent objects for instance. We chose to tackle this issue by connecting an echo to each echoes of its pulses' neighbors as illustrated in Figure 3.8b because we should keep all possible edge hypotheses before filtering them. There is little work in the litterature that uses this structure (Xiao, Vallet, and Paparoditis, 2013; Vallet et al., 2015). This mainly comes from the fact that the raw data of a MLS is often lost during pre-processing steps. However, we think that this data can provide useful information, and that the sensor topology build with it is the most efficient structure for our work.



(a) Representation of the 6-neighborhood induced by the sensor topology. Each point correspond to a laser pulse.



(b) Representation of the adaptation of the 6-neighborhood to a laser recording multiple echoes. Each point correspond to a returned echo.

Figure 3.8 – Illustration of the sensor topology-based neighbohood for pulses (3.8a) and returned echoes (3.8b).

3.2.2 Edge Reconstruction

As a point cloud can be interpreted as a set of 0D-simplices in a 3D space, we decide to add 1D-simplices (edges) between all points that must be connected in the real scene. This way we have a first hint of the local geometry of the scene: its dimensionality is at least 1. Then we add 2D-simplices (triangles) for all triplets of points that are in a local 2-dimensional neighborhood. The pipeline is illustrated on figure 3.9.

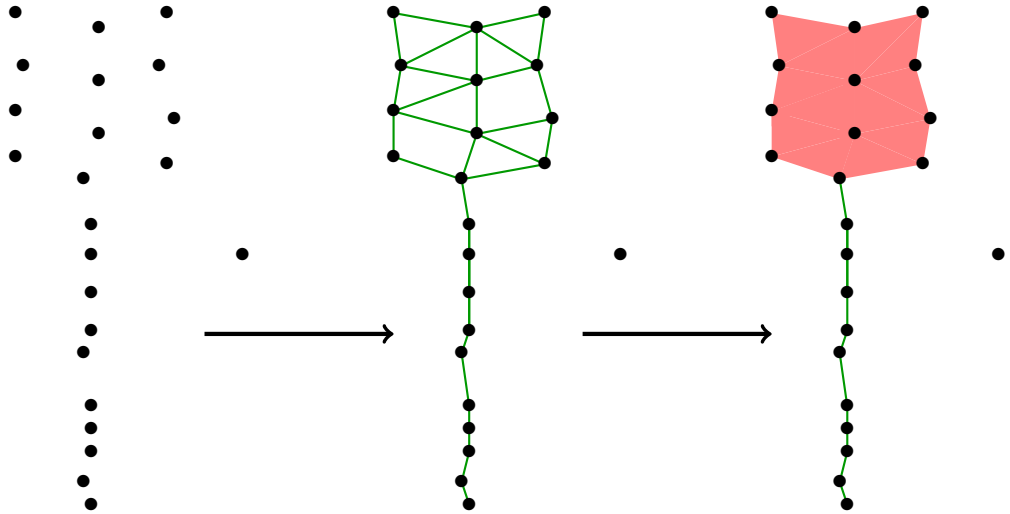


Figure 3.9 – From a set of points (left), we want to find all real edges (center), in order to obtain a final simplicial complex (right), that is adaptive to the local geometry of the scene. The scene represents a traffic sign. Initial points are shown in black, edges in green and triangles in red.

First, we start by presenting the edge reconstruction process in the case where each LiDAR pulse receives exactly one echo back. Then we extend the principle the muliple echo case.

3.2.2.1 Single Echo Case

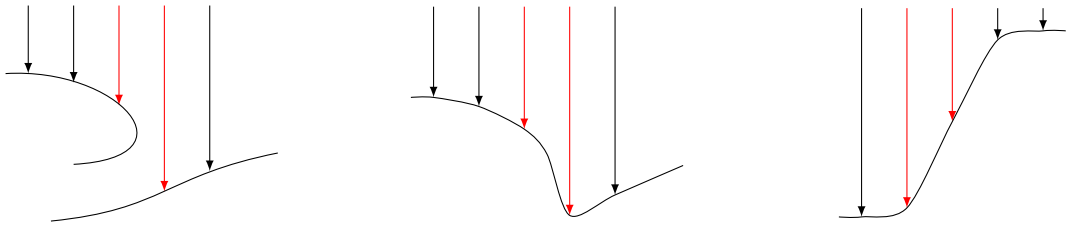
We first present the pipeline in the case where each pulse of the laser receives only one echo. This means that the nieghborhood considered in this case is the 6-neighborhood defined on Figure 3.8a.

We process each pair of neighboring echoes separately. The goal now, is to decide whether we should build an edge between two adjacent echoes or not. Our main criteria for creating an edge will be the depth difference between the two considered echoes. This depth difference is computed from a sensor viewpoint. If their depth difference is small, this means that both points were approximately at the same distance from the sensor. In this case, we assume that they are part of a same object and we create an edge between them.

On the other side, when two adjacent echoes have a huge depth difference, this means that the points are far one from another. To help the edge filtering process, we distinguished three different situations for this case. They are shown on Figure 3.10:

— in the first case (3.10a), we have two echoes with a huge depth difference falling on two different objects. In this case, we want to separate them.

— the second case (3.10b) shows two echoes falling on the same object. In this case, the geometry of the object is too complex compared to the density of acquisition



(a) Separation (b) Limit case (c) Non-separation

Figure 3.10 – Illustration of the 3 different cases considered for the edge reconstruction process. The 2 considered echoes are showned in red.

and we can't decide if we should connect the points or not. By default they will remain disconnected.

— the third case (3.10c) displays a particular case where we have three or more consecutive echoes with a huge depth difference that are nearly aligned. In this case, the laser pulses probabilby landed on a grazing surface, and we want to link the three echoes.

To achieve the edge filtering step, we propose the two following criterion:

— C_0 regularity: we want to prevent the reconstruction of an edge when the euclidian distance between its source and target is too high.

— C_1 regularity: we want to favor the reconstruction of an edge when three or more echoes are nearly aligned.

In order to be independent from the sampling density, we propose to express the regularities in an angular manner. Moreover, the sensor topology has an hexagonal structure, and we propose to treat each line in the 3 directions of the structure independently. We then express the regularities as:

— C_0 regularity: for an edge (E_p, E_{p+1}) between two neighboring pulses e_1 and e_2:

$$C_0(p, e_1, e_2) = 1 - \vec{e_p}(e_1, e_2) \cdot \vec{l_p} \,, \tag{3.1}$$

where $\vec{e_p}(e_1, e_2) = \frac{\overrightarrow{E_p E_{p+1}}}{||\overrightarrow{E_p E_{p+1}}||}$ and $\vec{l_p}$ is the direction of the laser beam of pulse p (cf. Figure 3.11). C_0 is close to 0 for surfaces orthogonal to the LiDAR ray and close to 1 for grazing surfaces, almost parallel to the ray.

— C_1 regularity, for an edge (E_p, E_{p+1}) between two neighboring pulses:

$$C_1(p, e_1, e_2) = |1 - \vec{e}_{p-1}(e, e_1) \cdot \vec{e}_p(e_1, e_2)| \cdot |1 - \vec{e}_p(e_1, e_2) \cdot \vec{e}_{p+1}(e_2, e)| \,. \tag{3.2}$$

C_1 is close to 0 is the edge is aligned with at least one of its neighboring edges, and close to 1 if it is orthogonal to all neighboring edges.

Given two adjacent echoes, we consider that if the C_0 regularity is high enough, we can ensure the real existence of the edge, and don't have to compute the C_1 regularity. We denote this threshold α_m. For all the other cases, we compute the C_1 regularity and filter the edges according to the C_0 and α_m. To summarize, we add an edge in the reconstruction if and only if:

$$\begin{cases} C_0 & < \alpha_m \,, \\ & or \\ C_1 & < \frac{\lambda \cdot \alpha_m \cdot C_0}{\alpha_m - C_0} \,, \end{cases} \tag{3.3}$$

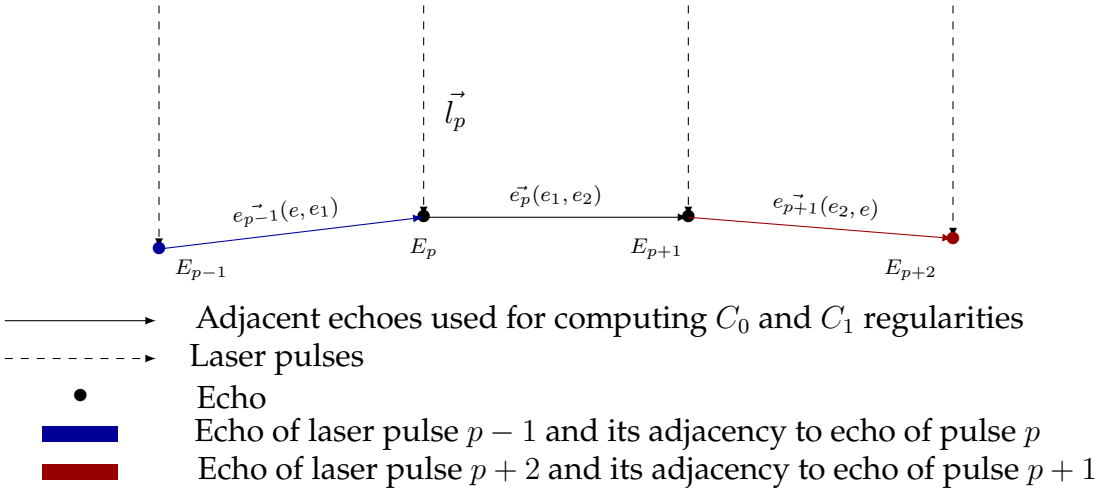


Figure 3.11 – Illustration of the computed regularities C_0 and C_1.

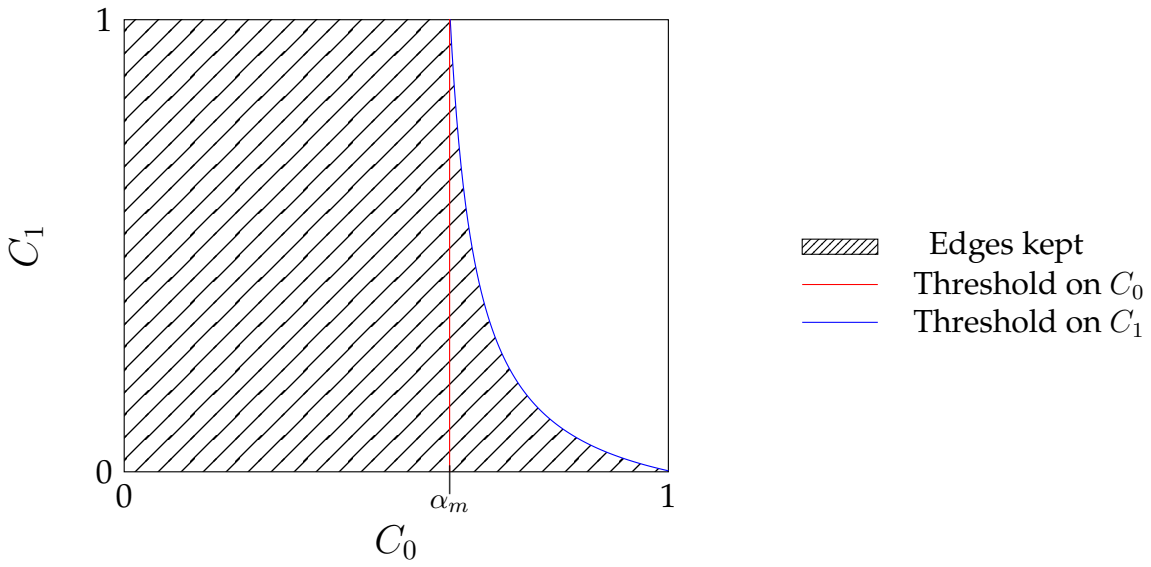


Figure 3.12 – Filtering of edges knowing C_0 and C_1.

where λ sets how much C_1 regularity can compensate for C_0 discontinuity. A high value of λ allows more edges to be kept. This criteria is illustrated on figure 3.12. The red line represent the α_m threshold and the blue line corresponds to the limit cases between removing and keeping the edges depending on C_0 and C_1.

We also want to filter edges that would remain unconnected to any other edge in the cloud, or just connected to one other edge but with different directions. Actually this often occurs on noisy areas where an edge can pass the regularity criteria "by chance". However, it is very improbable for this to happen for two neighboring edges. Hence, we propose an additional criterion to favor a reconstruction which keeps only points instead of isolated or unaligned edges.

Let e be an edge and $\{e_1, \ldots, e_n\}$ its adjacent edges. We consider that if we find an edge $e_i \in \{e_1, \ldots, e_n\}$ such that:

$$1 - \vec{e} \cdot \vec{e_i} < \epsilon \quad , \tag{3.4}$$

where ϵ is the tolerance on edge reconstruction, e is not alone or unaligned and we keep it in the simplicial complex.

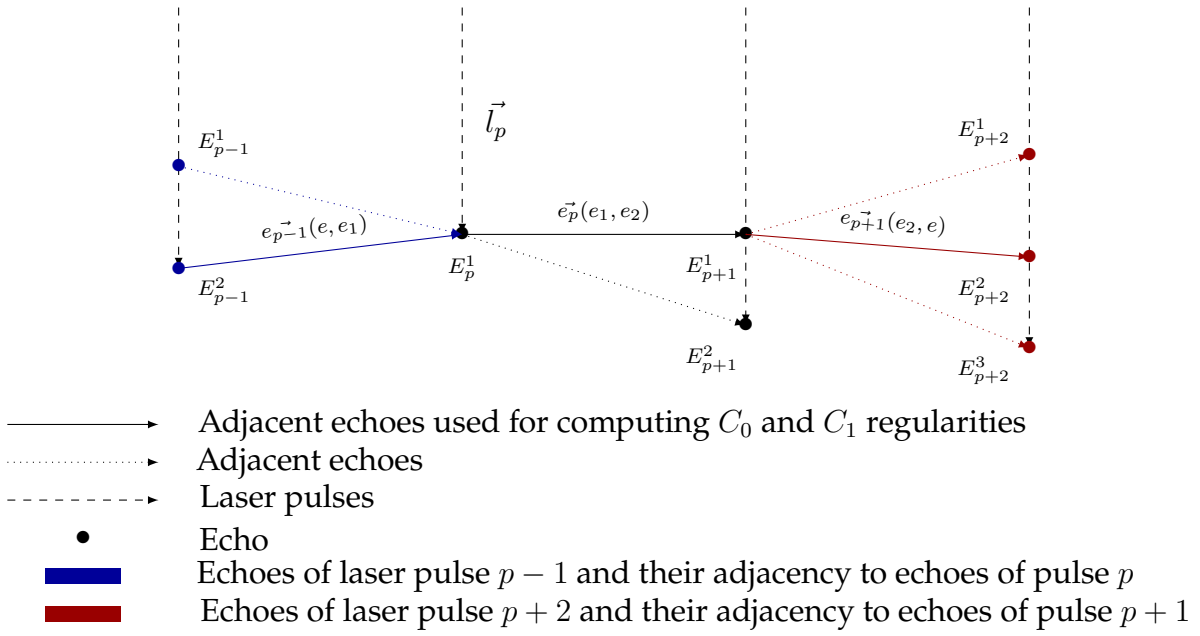


Figure 3.13 – Illustration of the computed regularities C_0 and C_1. From the blue and red echoes, we selected the closest echoes to the line based on $\vec{e_p}(e_1, e_2)$. The selected echoes are used to compute the C_0 and C_1 regularities.

3.2.2.2 Multi Echo Case

In this case, we consider that each LiDAR pulse can have 0 or several returned echoes. This means we use the neighborhood presented on Figure 3.8b.

We generalize the approach presented in the previous section in the multiple echo case. This means that the regularities are expressed in an angular manner, and that each direction in the sensor topology in process independently as well. For the reminder of this thesis, and because a single pulse can have multiple echoes, we will express, for a pulse p, its echoes as: E^e_p where $e \in 1 \dots N_p$, with N_p the number of echoes of p. We then express the regularities as:

— C_0 regularity: for an edge $(E^{e_1}_p, E^{e_2}_{p+1})$ between two echoes of two neighboring pulses:

$$C_0(p, e_1, e_2) = 1 - \vec{e_p}(e_1, e_2) \cdot \vec{l_p}\,, \tag{3.5}$$

where $\vec{e_p}(e_1, e_2) = \frac{\overrightarrow{E^{e_1}_p E^{e_2}_{p+1}}}{||\overrightarrow{E^{e_1}_p E^{e_2}_{p+1}}||}$ and $\vec{l_p}$ is the direction of the laser beam of pulse p (cf Figure 3.13). C_0 is close to 0 for surfaces orthogonal to the LiDAR ray and close to 1 for grazing surfaces, almost parallel to the ray.

— C_1 regularity, for an edge $(E^{e_1}_p, E^{e_2}_{p+1})$ between two echoes of two neighboring pulses:

$$C_1(p, e_1, e_2) = min^{N_{p-1}}_{e=1} |1 - \vec{e}_{p-1}(e, e_1) \cdot \vec{e}_p(e_1, e_2)| \cdot min^{N_{p+2}}_{e=1} |1 - \vec{e}_p(e_1, e_2) \cdot \vec{e}_{p+1}(e_2, e)|\,, \tag{3.6}$$

where the minima are given a value of 1 if the pulse is empty. C_1 is close to 0 is the edge is aligned with at least one of its neighboring edges, and close to 1 if it is orthogonal to all neighboring edges.

From these regularities, we propose a simple filtering based on the computed angles. Figure 3.13 illustrates the C_0 and C_1 regularities. Considering two adjacent echoes $E_p^{e_1}$ and $E_{p+1}^{e_2}$, the C_0 regularity can be interpreted as the cosine of the angle between the laser beam direction in p and $\vec{e_p}(e_1, e_2)$. We want to favor low values of the C_0 as it corresponds to echoes with a low depth difference. On the other hand, to compute the C_1 regularity, we have to process all the echoes of the preceding and following pulses along the 3 directions of our structure. For the preceding and following pulses, we select the echo which minimizes $\mid 1 - \vec{e_{p-1}}(e, e_1) \cdot \vec{e_p}(e_1, e_2) \mid$ (respectively $\mid 1 - \vec{e_{p+1}}(e_2, e) \cdot \vec{e_p}(e_1, e_2) \mid$). This gives us an information about the tendency of the considered edge to be collinear with at least one of its adjacent edges. We will favor the most collinear cases.

We decide to add an edge in the reconstruction if and only if it satisfies the criterion of Equation 3.3. The pseudo-code for the *edge reconstruction* is presented in Algorithm 1.

Algorithm 1 Edge reconstruction

$\mathcal{SC} \leftarrow \mathcal{V}$
for $E_p^i \in \mathcal{V}$ **do**
  for $E_q^j \in$ Neighborhood$(\mathcal{V}, E_p^i)$ **do**
    if $C_0(p, E_p^i, E_q^j) < \alpha_m$ **then**
      $\mathcal{SC} \leftarrow$ Edge(E_p^i, E_q^j)
    else if $C_1(p, E_p^i, E_q^j) < \frac{\lambda \cdot \alpha_m \cdot C_0(p, E_p^i, E_q^j)}{\alpha_m - C_0(p, E_p^i, E_q^j)}$ **then**
      $\mathcal{SC} \leftarrow$ Edge(E_p^i, E_q^j)
    end if
  end for
end for
return $\mathcal{SC}$

We also filter out unconnected edges, or edges not aligned with any other simplex, in the same way as in the *single echo case*.

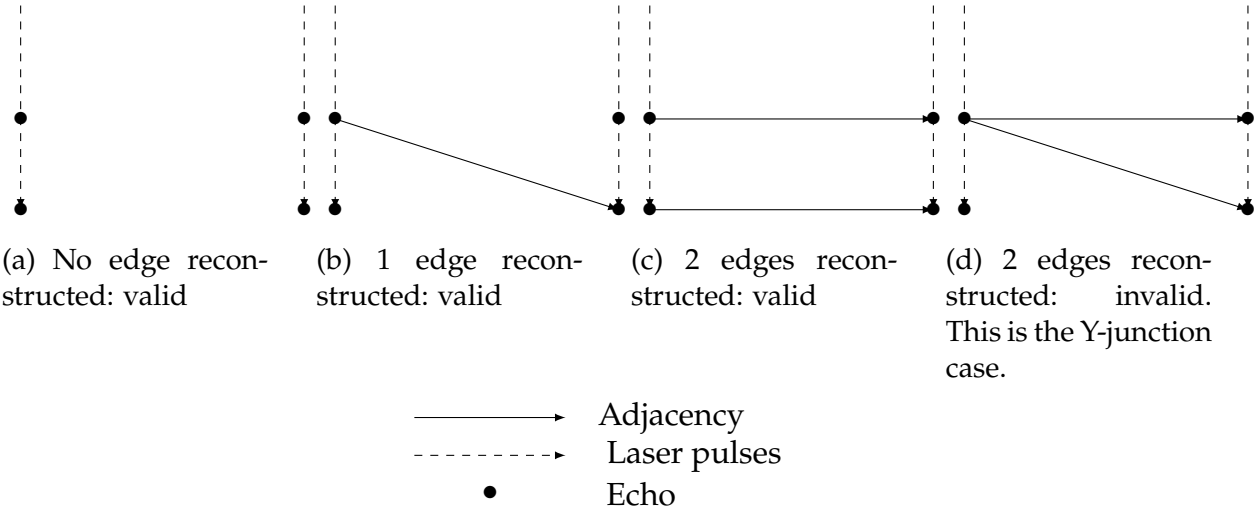
3.2.2.3 Y-junctions

When processing LiDAR point clouds with *multiple echoes* per pulse and using the sensor topology, there can be some edges added between two different echoes of a same pulse and the same echo of the next pulse. This is what we call a Y-junction. We do not want to keep such junctions in our *simplicial complex* as it is highly improbable that an object, when acquired by a laser that has the shape of a Y-junction. The different junctions are presented on Figure 3.14.

Y-junctions can be penalized by adding a cost for edges sharing an echo and whose other echo belong to an identical pulse. However, we did not observe such configurations in our experiments so we did not implement a penality for Y-junctions.

3.2.3 Local Hole Filling

Once we obtained a set of edges in our point cloud, a simple approach to filter triangles is to only keep the triangles for which three edges have passed the edge filtering described above. This means that every triplet of self-connected edges will create a triangle. Even if this method is an easy way to retrieve most triangles of the scene, it prevents recovering triangles in areas where edges are close to the threshold presented



(a) No edge reconstructed: valid

(b) 1 edge reconstructed: valid

(c) 2 edges reconstructed: valid

(d) 2 edges reconstructed: invalid. This is the Y-junction case.

Figure 3.14 – Illustration of the different ways of adding edges between neighborhing echoes in the multiple echo case. Figure d shows the Y-junction which is invalid.

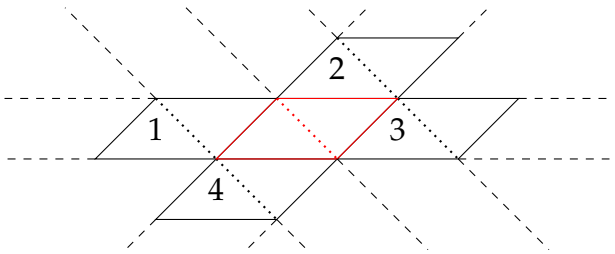


Figure 3.15 – Representation of a wedge (red) and its adjacent wedges. The limit between the triangles of each wedge is represented with a dotted line. The dashed lines stand for the directions of our structure. Wedges adjacent to the red one are numbered from 1 to 4.

in previous section, in which case triangles will often have some edges just below and some just above the threshold so most triangles would be filtered out.

In order to regularize the computed triangulation, we want to favor triangles that are coplanar to other adjacent triangles, in the same way we favored edges aligned with at least one neighboring edge. This is motivated by the fact that we want to ensure spatial regularity in our scene. In fact, if a single triangle is created, this usually means that a larger, locally planar, object is present in the scene at this place, and other triangles close to the first one should be reconstructed as well. If this is not the case and we can't find any other coplanar structure close to the reconstructed triangle, this usually means that the triangle was created by error as we find very unlikely the cases where a single triangle is left alone in the reconstruction. Moreover, we want to remove all the triangles that may be formed by edges in noisy parts of the cloud as we cannot ensure the existence of such simplex in the original scene. However we decide to keep the edges as they passed the *edge reconstruction* step and they provide useful information (pairs of points are connected).

As triangles are 2D objects, we want to define a 2D C_1 regularity by separating between C_1 regularity along two directions. Unfortunately, a triangle has 3 neighbors. We solve this problem by filtering pairs of adjacent triangles (that we will call wedges) which have four adjacent wedges in 2 separate directions as illustrated on figure 3.15.

The filtering we propose is to keep wedges that are C_1 regular with neighboring

wedges in the two directions different, where C_1 regularity is defined as the criterion:

$$1 - |W^N \cdot W_i^N| < \omega \,, \tag{3.7}$$

where W is a wedge whose normal is W^N and $\{W_1, \ldots, W_n\}$ are its adjacent wedges whose normals are $\{W_1^N, \ldots, W_n^N\}$ respectively. This means, on figure 3.15, that if the red wedge is only C_1 regular with the wedges 1 and 3, it will be discarded, while it will be kept if it is regular with only 1 and 2. ω is the tolerance on the co-planarity of the two wedges to define regularity. The rationale behind this choice is the same as for the edges: being irregular with both neighbors in one direction means that we are on a depth discontinuity in that direction that cannot be distinguished from a grazing surface, while regularity with at least one neighbor in both directions means that the wedge is part of a (potentially grazing) planar surface.

The pseudo-code for the full reconstruction is presented in Algorithm 2.

Algorithm 2 Full reconstruction

```
SC ← EdgeReconstruction(V)
for W ∈ Wedges(SC) do
  for W_i ∈ NeighborhingWegdes(W, SC) do
    if 1 − |W^N · W_i^N| < ω then
      SC ← TriangleFromWedge(W)   % Add the two triangles forming the wedge
    end if
  end for
end for
return SC
```

3.3 Experiments

We now present some results using our method, to reconstruct MLS data as a simplicial complex. In this part, every simplicial complex will be represented as follow:

— triangles in red,

— edges in green,

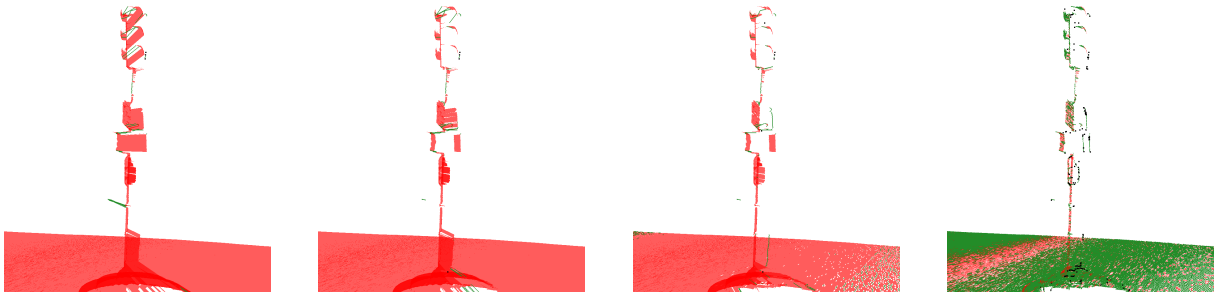
— points in black.

Note that following its mathematical definition, the endpoints of an edge of a simplicial complex also belong to the complex, and similarly for the edges of a triangle, but we do not display them for clarity.

We implemented the pipeline presented before, first with only the edge filtering and the simple triangle reconstruction from edge effectively forming a triangle. Then we added the triangle filtering step. We compared our results with a naive filtering on edge length where triangles in the simplicial complex correspond to all triplets of edges forming a triangle.

3.3.1 Dataset

For all the following tests, we used data from the Stereopolis vehicle (Paparoditis et al., 2012). The scenes have been acquired in an urban environment (Paris) and are mostly composed of roads, facades of Hausmannian buildings, trees, and urban planning. In order to facilitate the process, the entire acquisition has been cut in blocks,



(a) α_m= 0.01 (b) α_m= 0.05 (c) α_m= 0.1 (d) α_m= 0.5

Figure 3.16 – Influence of α_m. λ is fixed to 10^{-4}. The scene represents a traffic light on the road. The best result here is the one shown on Figure 3.16b. Lower values of α_m connect points that are not part of a same object and highest values of α_m prevent the creation of most edges and triangles.

corresponding to 1 second of scan. Each block is processed independently. We do not look for edges nor triangles across adjacent blocks. This causes the apparition of small gaps on some results, they correspond to the limit of each block.

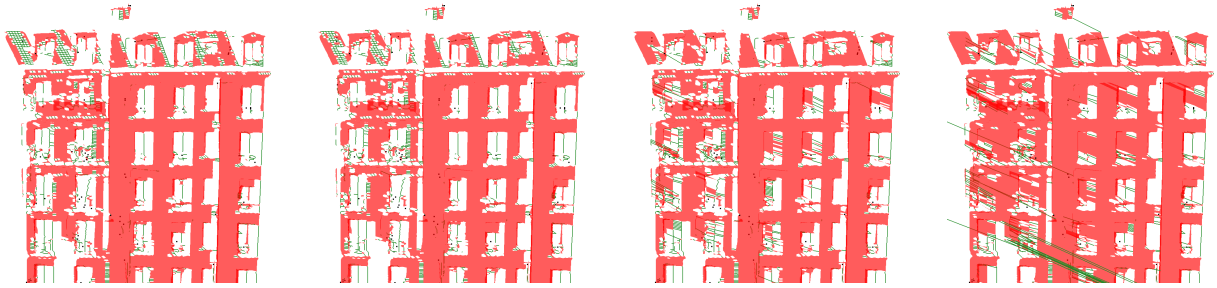
3.3.2 Parameterization

The parameterization step was conducted in two experiments. In the first set of experiments, the impact of parameters α_m and λ were studied. α_m and λ parameters condition the edge filtering step. For the remaining parameters ω and ϵ, their influence was tested in a second set of experiments. The last two parameters correspond to the homogenization part. Because all our criteria depend on trigonometric functions (the dot products of normalized vectors is the cosine of their angle), all these parameters are chosen in $[0, 1]$. Lastly, we compare both methods with the naive filtering on edge length.

3.3.2.1 Parameterization of α_m and λ

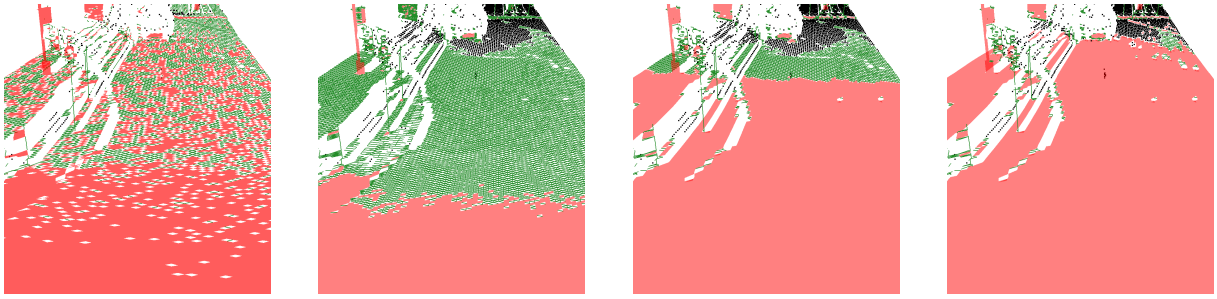
We first study the influence of α_m. A high value will discard many edges and prevent the formation of triangles, whereas a low value will preserve too many edges on actual discontinuities. The results are presented in figure 3.16. We see on the left example that on the one hand, low values of α_m allow the formation of edges between the bottom of the traffic sign and the road. On the other hand, high values of α_m show that fewer triangle are present in the reconstruction, even on the road as shown on Figure 3.16d.

The second parameter of this method, λ is a trade off between the creation of elongated edges (which are part of the scene in the grazing surface case) and the removal of edges connecting points far from one another. With this paramter, we favor sets of collinear edges (which can be found in the grazing surface case). Figure 3.17 illustrates the tuning of this λ parameter. On the one hand, for high values of λ (right), edges between window bars and walls or insides of buildings are created. On the other hand, when λ is too low (left), most of the edges are not retrieved. For these cases, the number of remaining edges is low (hundreds of edges for millions of points), and lowering λ removes edges that may be useful for human interpretation of the reconstruction.



(a) $\lambda = 10^{-6}$ (b) $\lambda = 10^{-4}$ (c) $\lambda = 10^{-2}$ (d) $\lambda = 0.1$

Figure 3.17 – Influence of λ. α_m is fixed to 0,05. The scene represents a facade in the grazing surface case.



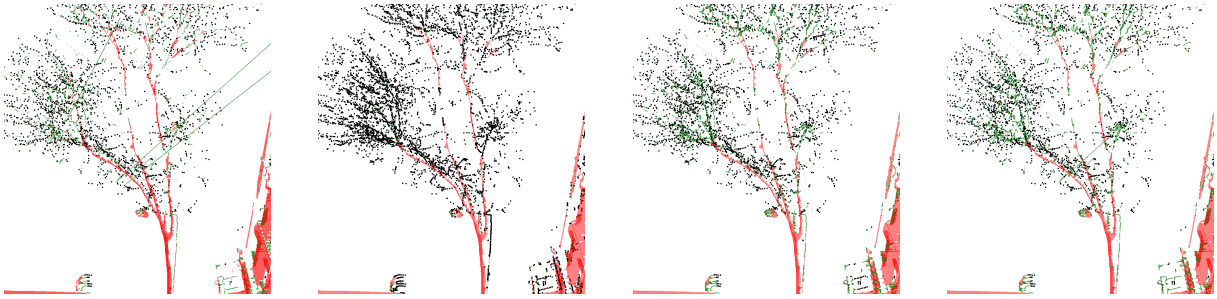
(a) No triangle filtering (b) $\omega = 10^{-5}$ (c) $\omega = 10^{-4}$ (d) $\omega = 10^{-3}$

Figure 3.18 – Parametrization of ω. ϵ is fixed to $5 \cdot 10^{-3}$. The scene represents a road in the grazing surface case. Increasing ω higher than $10 \cdot 10^{-3}$ do not significantly change the results.

3.3.2.2 Parameterization of ω and ϵ

For this set of experiments, α_m and λ were fixed respectively to 0.05 and 10^{-4}. The influence of the last two parameters is especially visible on noisy areas and grazing surfaces, where the level of detail of the scene is close to the acquisition density. We first studied the effect of parameter ω on triangles. For high values of ω, we expect that many triangles will be retrieved, especially in the grazing surface case, where sometimes our algorithm can struggle to retrieve edges in the 3 directions of the neighborhood presented on Figure 3.8b (but performs well on two directions). The results are shown on figure 3.18 and illustrate our problem in the grazing surface case. Figure 3.18a is the baseline computed previously. As expected, the number of triangles increases for high values of ω and the reconstruction of the road is less holed and noisy than without the triangle filtering part. The main drawback of the homogenization based on wedges is its propensity to let a few triangles in noisy areas like tree's foliage.

The second parameter ϵ is a regularization term on the edges. Low values of ϵ will decrease the number of edges, thus leaving many points not linked to others. Results are presented in figure 3.19. As previously, the figure on the left shows the output of the first filtering step. Figures corresponding to lowest values of ϵ respect our previsions: the number of edges keeps decreasing whereas the number of points increase. A side effect of using an edge regularization based as defined in Equation 3.4 can be seen on figure 3.19b, as there is nearly no edge in the tree's foliage.



(a) No triangle filtering (b) $\epsilon = 10^{-5}$ (c) ϵ=$5 \cdot 10^{-3}$ (d) $\epsilon = 10^{-2}$

Figure 3.19 – Parametrization of ϵ. ω is fixed to 10^{-3}. The scene represents a tree with its foliage. In the bottom right corner we can see a facade

3.3.3 Results

In this section we present some results computed on the dataset presented earlier. We compare a simplicial complex reconstruction without a geometric homogenization (which we denote *edge filtering*) and a reconstruction with a geometric homogenization (which will be refered to as *triangle filtering*). We compare the edge filtering and the triangle filtering to a naive baseline introduced in the following paragraph.

3.3.3.1 Baseline

We implemented a naive filtering baseline. This method first filters edges according to their length: we consider that two adjacent points that are too far from one another are not part of the same object. This means that the only criterion for deciding the reconstruction of an edge is its length. Then, we reconstruct a triangle for every triplet of self-connected edges. The final reconstruction is a simplicial complex too. We name this method the *naive filtering*.

3.3.3.2 Results

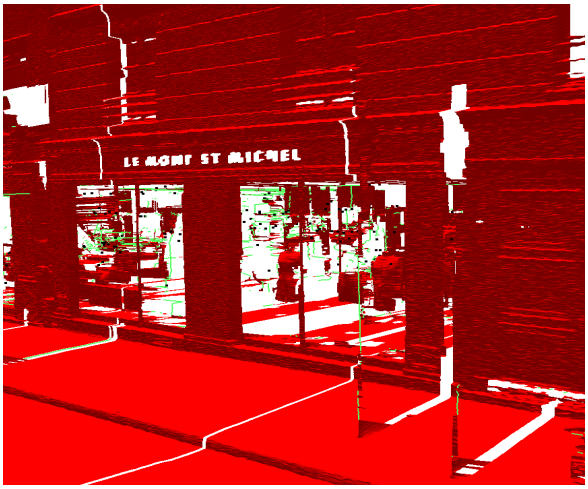
In this part, we compare our two methods (*edge filtering* and *triangle filtering*) to the baseline presented before. The naive filtering is based on a 0.5 meters threshold. For both methods, α_m and λ are respectively fixed to 0.05 and 10^{-4}. Furthermore, for the *triangle filtering*, ω and ϵ are fixed to 10^{-3} and $5 \cdot 10^{-3}$ respectively. A video of the results is available on Youtube (*Sensor-Topology based simplicial complex reconstruction from mobile laser scanning*). To our knowledge, there exist no benchmark for the evaluation of simplicial complexes reconstruction, nor any dedicated metric. As the quality of the reconstruction may be hard to evaluate, we validate our results visually. Regarding the processing times, the baseline and the *edge filtering* method run in approximately 5 seconds for a 1 second acquisition block. The *triangle filtering* method, due to its design, is a bit longer and run in approximately 1.5 minutes for a 1 second acquisition block.

Figure 3.20 presents a reconstruction in a complex urban scene. Unlike the naive filtering, our methods are able to retrieve thin objects such as poles or windows bars without merging them to the closest objects. This is illustrated on figure 3.20. The top right image of this figure is an extract from Google Streetview to help interpretation.

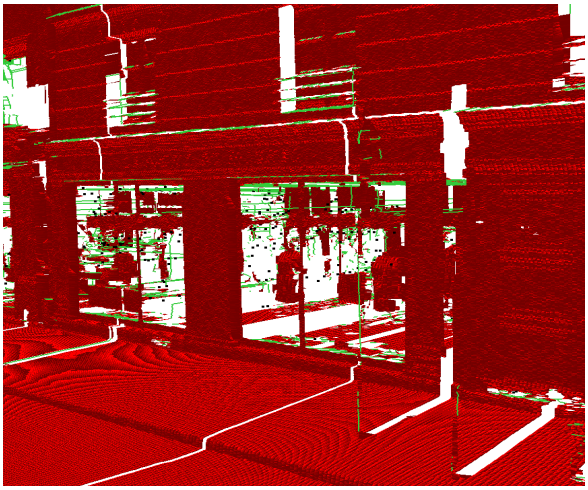
Figure 3.21 focus more on specific areas of the scan. The first row shows the naive filtering method, whereas the second and third rows present respectively the edge filtering method and its extension: the triangle filtering. We remark that the naive fil-

tering method fails to retrieve limits between objects (e.g. between poles and road, or people and buildings), whereas both of our methods are able to separate these objects. The main advantage of the triangle filtering over the edge filtering that can be seen here, is that the triangle filtering method helps to reduce the noise that occurs in complex areas such as tree foliage or grazing surfaces. We assume that in complex areas we cannot ensure the existence of connections between some points and will favor a reconstruction that remains careful on such areas. This is why the second method, which produces a simplicial complex less noisy than the reconstruction of the edge filtering method, is considered a more appropriate baseline for further developments, even if the triangle filtering discards some edges or triangles that had been well retrieved by the edge filtering method on grazing surfaces.

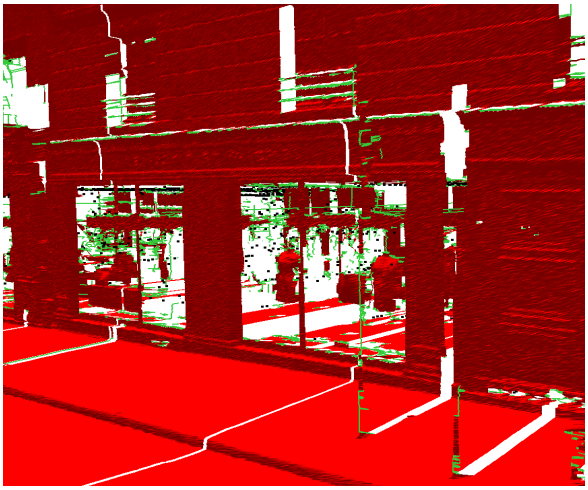
(a) Image of the scene from Google (*View of the Mabillon street next to the corner of Lobineau street (Paris)*)



(b) Naive method

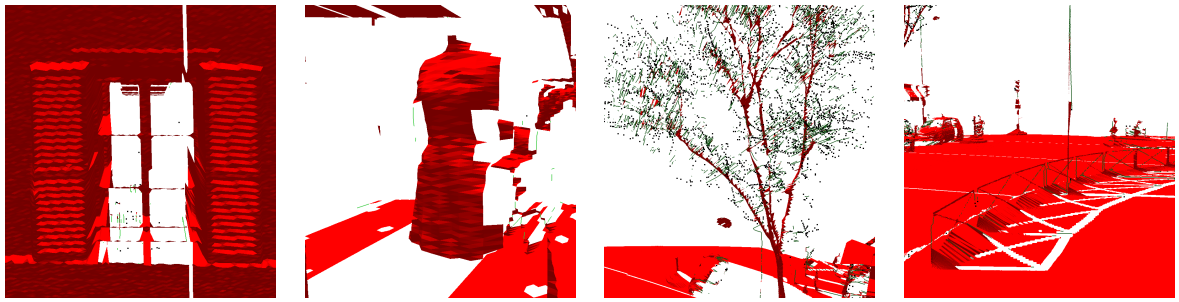


(c) Edge filtering method. The visual artifacts seen on the road and the pavement are due to rendering issues that did not appear for simplicial complexes produced by other methods.

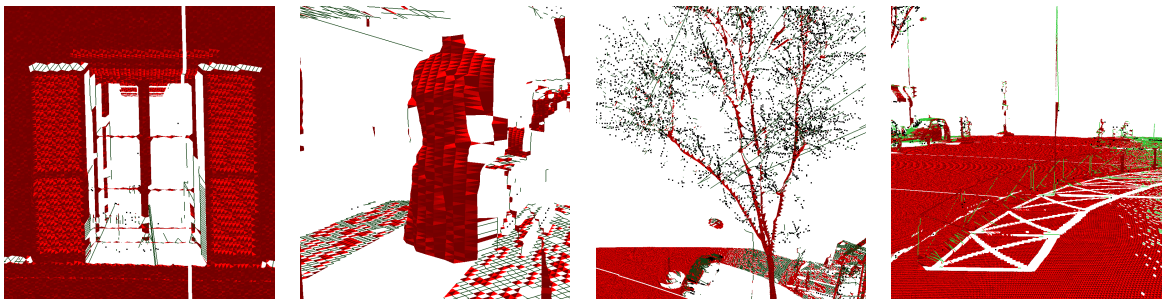


(d) Triangle filtering method.

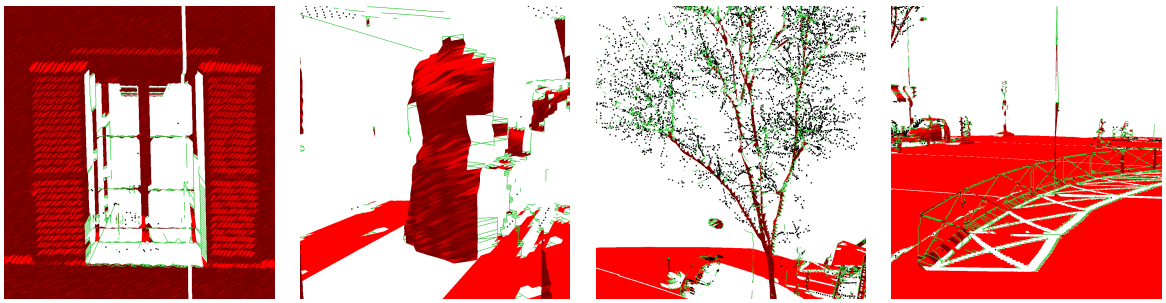
Figure 3.20 – Results on a complete urban scene, with road, facades, poles and pedestrians.



(a) Naive filtering



(b) Edge filtering



(c) Triangle Filtering

Figure 3.21 – Comparison of the three methods: the naive filtering, the edge filtering and its extension with the triangle filtering. From left to right, the scenes represent: a window, a model in a showcase, a tree and barriers on a pavement.

3.4 Weighted Simplicial Complexes Reconstruction

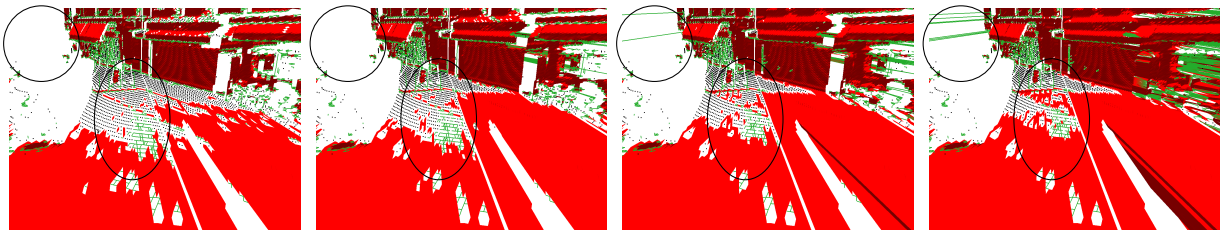
We noticed that our reconstruction method was less accurate in part of the data that are far from the sensor. In remote areas, acquired points tend to be further from one another. Thus, the regularities as defined in Section 3.2 discard edges more frequently. Hence, we decided to take into account the distance of the points to the sensor. To achieve this, we modify the C_0 regularity introduced in Section 3.2.

3.4.1 Weighted Reconstruction

From the C_0 regularity, we derive the following weighted C_0^w regularity:

$$C_0^w(p, e_1, e_2) = C_0(p, e_1, e_2) + \kappa \cdot \frac{l_p}{l_{max}}, \tag{3.8}$$

where l_p is the distance from the sensor to the point, l_{max} is the maximum distance between a position of a laser and one of its recorded echoes and κ is the parameter con-



(a) κ= 0 (b) κ= 0.1 (c) κ= 0.2 (d) κ= 0.3

Figure 3.22 – Influence of κ. The scene represents a road far from the laser, with a facade in the background. The main differences can be seen on the road, which is filled as κ increases; and on the edges between facades and small objects. The black circles show areas with the most differences.

ditioning the influence of the weighting term. C_0^w is no longer independent from the sampling and varies between 0 and $1 + \kappa$. This formulation allows for our reconstruction to stay consistent in areas close to the sensor, while favoring the reconstruction of edges in more distant areas.

The influence of this parameter can also be improved by adding a threshold on edge length. This way, we prevent the apparition of long edges between objects far away from one another (e.g., with a depth difference of several meters). We propose this approach since it is very unlikely that an edge of several meters long links two adjacent echoes. Such a configuration would mean that there is an object in the scene big enough to have a side of several meters, and that this object is close to parallel to the laser beam. Intuitively, this description could correspond to a building. However in our data, there are very few points behind the main facade of a building and they are too sparse to let the algorithm find a possible shape.

3.4.2 Parameterization of κ

In this section, we investigate the influence of the weighting parameter κ on the reconstruction. We conducted two different experiments: the first one shows the influence of κ if we do not threshold edge length. The second one corresponds to the case where we limit the maximum length of edges, thus enabling the reconstruction of many more triangles and edges.

For this set of experiments, the values of α_m, λ, ω and ϵ where respectively fixed to: $5 \cdot 10^{-2}$, 10^{-3}, 10^{-3} and $5 \cdot 10^{-3}$. We expect the C_0^w regularity to improve the reconstructions using only the C_0 regularity. Indeed, this would allow the creation of more edges in the simplicial complex, in remote areas, where they are more easily discarded. These edges should also encourage the creation of triangles on places that contained holes, and in the farthest places of the scene. The results are shown on figure 3.22. The top left figure is the output produced using the *triangle filtering* method introduced in Section 3.2. We see that weighting the C_0 regularity helps our algorithm to retrieve edges and triangles that were lost before. Low values of κ fill some holes in the road, whereas high values of κ tends to create edges between distant objects. The number of points, edges and triangles for each simplicial complex is shown in table 3.1. There is a significant decrease of the number of points and egdes when κ rises, whereas the number of triangles increases a bit.

	Triangles	Edges	Points
$\kappa = 0$	737,596	364,730	12,090
$\kappa = 0.1$	744,702	353,968	10,944
$\kappa = 0.2$	752,726	333,926	10,331
$\kappa = 0.3$	758,714	348,742	9076

Table 3.1 – Number of triangles, edges and points per simplicial complex.

(a) κ = 0 and no edge length filtering

(b) κ = 0.03 and no edge length filtering

(c) κ = 0.2 and edge length filtering of 10 meters

(d) κ = 0.5 and edge length filtering of 10 meters

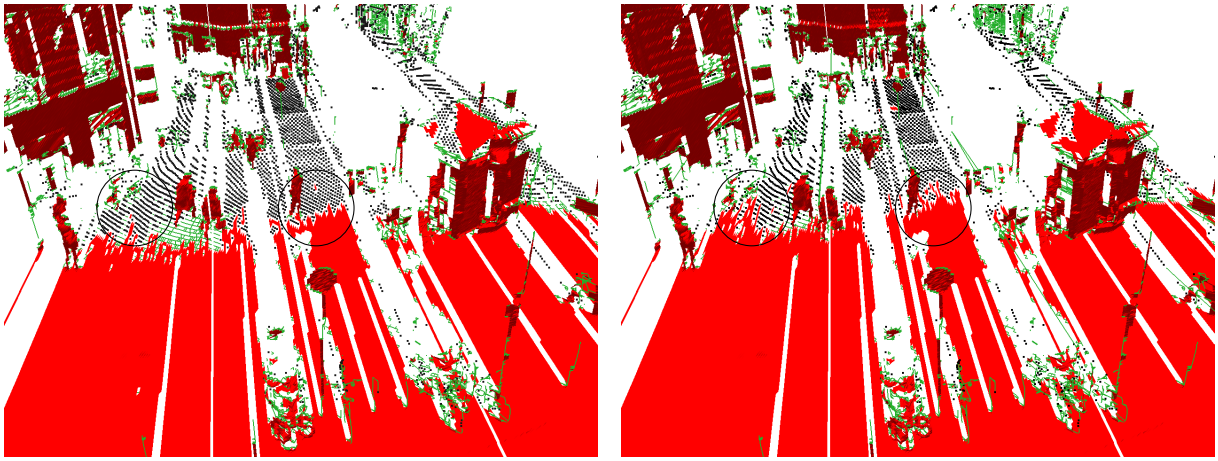
Figure 3.23 – Influence of κ. The scene represents a facade far from the laser. The influence of κ is mostly visible on the top left corner of the facade.

The results of the second set of experiments are shown on figure 3.23. Here, we wanted to study the influence of κ when we prevent the reconstruction of too long edges. We fixed a maximum length for all edges to 10 meters. The figure on the left corresponds to a reconstruction without any weighting. Next, the weighting without any thresholding is done for $\kappa = 0.03$. Last, we show two examples in which we set a threshold on the edge length. This threshold is set to 10 meters. κ is fixed respectively to 0.2 and 0.5. Using this threshold while increasing the value of κ allow the creation of a reconstruction containing less holes while preventing the creation elongated edges. The influence of κ and the threshold on edge length is especially visible in figure 3.23c. However, a too high value of κ creates edges between objects which we do not want to connect.

3.4.3 Results

In this part, we compare the weighted reconstruction to the *triangle filtering* method presented in Section 3.2. For these methods, α_m , λ, ω, ϵ and κ are respectively fixed to 0.05, 10^{-4} , 0.1, 5×10^{-3} and 0.4. As for the previous set of experiments, we visually validated the results. Figure 3.24 presents a reconstruction in a complex urban scene with facades, roads, trees . . .. The figure on the left shows the results of the unweighted method (see Section 3.2) and the right one shows the weighted reconsturction. There is nearly no difference on small objects like poles or pedestrians. However, we can see that the weighted method is able to retrieve more triangles in the limits of laser's scope. This is showned by filled facades in the top of the image, and also by a smoother reconstruction of the road, even if reconstructing the whole road would recquire a higher value of κ that would perform very poorly on the remaining parts of the reconstruction.

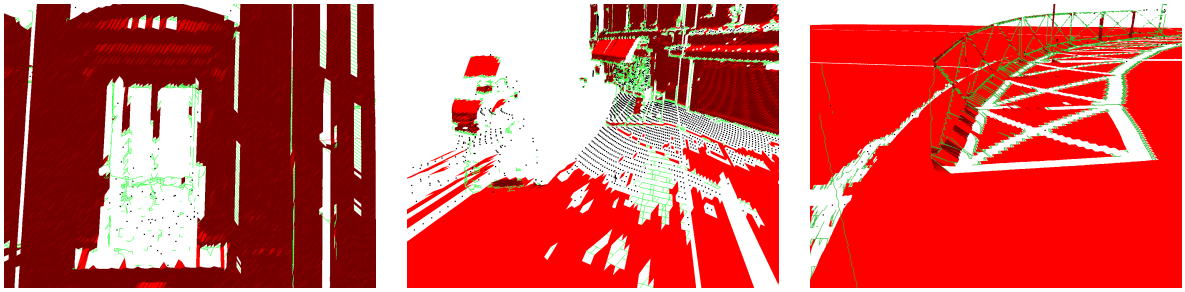
A zoom on specific areas of the scene (windows, fences . . .) is visible on Figure 3.25.



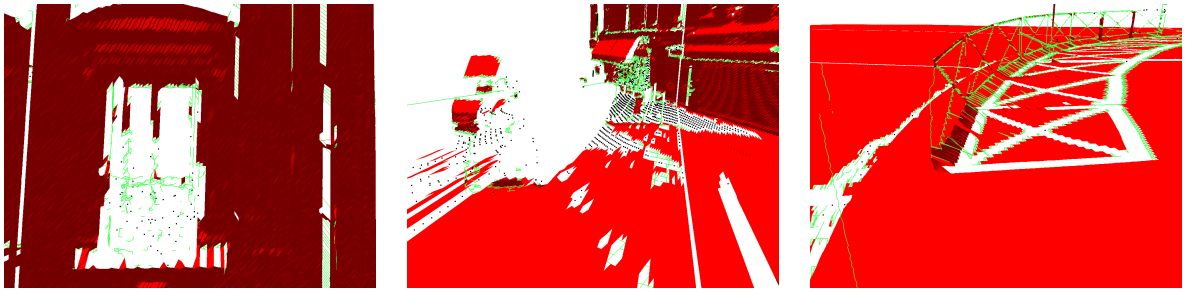
(a) Unweighted reconstruction (Guinard and Vallet, 2018a)

(b) Weighted reconstruction (Guinard and Vallet, 2018b)

Figure 3.24 – Results on a complete urban scene, with road, facades, poles and pedestrians. The black circles show areas with the most difference.



(a) Unweighted reconstruction (Guinard and Vallet, 2018a)



(b) Weighted reconstruction (Guinard and Vallet, 2018b)

Figure 3.25 – Comparison of the two methods: the unweighted and the weighted methods. From left to right, the scenes represent: a window, a road in the grazing surface case and fences.

We note that the weighting of the edge reconstruction process does not harm the final reconstruction in areas where the unweighted reconstruction performed well. Most differences between both reconstructions happen on grazing surfaces, where the weighted method is more efficient. The number of triangles, edges and points are provided in Table 3.2. Again, the number of points and edges dicreases and the number of triangles rises a bit. This is mainly due to the fact that by authorizing more edges in the first step of the reconstruction, the weighted method can produce more triangles later, thus dicreasing the number of remaining edges and points.

	Triangles	Edges	Points
Unweighted	1,143,482	755,582	13,757
Weighted	1,153,932	713,064	11,942

Table 3.2 – Number of triangles, edges and points per simplicial complex in Figure 3.24.

In conclusion, the add of the weighting parameter let us improve the results in remote parts of the scene, but it forces the user to have some knowledge on the scene organization to parameterize this method. Moreover, the reconstruction is still not perfect (portions of road are not reconstructed as triangles), hence we will use the results from the unweighted method for the remaining of this thesis.

3.5 Conclusion

3D-simplicial complexes are composed of points, edges and triangles. This allow us to build a representation of a scene that is adaptive to the local geometry. Unlike for meshes, we choose here to reconstruct every simplex for every dimension, and this prevents all the problems that can appear when reconstruting meshes on object whose level of detail is too close to the geometric resolution of the acquisition, like over-simplification. This occurs, for exemple, when acquiring the foliage part of a tree. In our case, we can just add edges, or let points unconnected. For a tree, the trunk and the main branches will be reconstructed as a set of triangles, whereas the twigs will be represented with edges and the foliage as a set of points.

Moreover, we used the sensor topology to create an adjacency relationship between points. This adjancency is independent from the sampling of the scene and help us take into account all the small details of the scene. We also want to emphasize on the fact that this method does not recquire any assumption on the scene geometry; it only needs points' coordinates and the sensor position and acquisition time for each point.

However, this reconstruction suffers from its high locality: it is still noisy and contains thousands of simplices for regions that could be represented with just a few ones (e.g. the road). Also, this reconstruction can be harmed by occlusions: the part of a building behind a trafic sign will not be reconstructed. Furthermore, the reconstruction contains mainy holes, especially in areas far from the sensor. We showed that the use of wedges and the weighting of the reconstruction could overcome part of this drawback, but it is not sufficient to recover from big occlusions. A more global hole filling process (Chauve, Labatut, and Pons, 2010; Harary, Tal, and Grinspun, 2014; Van Sinh, Ha, and Thanh, 2016) should improve the reconstruction on such parts of the scene.

Last, this method gives strong hints on the local geometry of each part of the scene. Thus, it can serve as a baseline for a generalization method (Hoppe, 1996; Popović and Hoppe, 1997; Nan and Wonka, 2017), or a semantic segmentation method (Rusu et al., 2009; Jeong, Yoon, and Park, 2018). In the following chapter, we will use this reconstruction as a baseline to detect planar parts of the scenes.

4 Generalization of 3D Point Clouds

Contents

This chapter focuses on the generalization of 3D-simplicial complexes. We call *generalization of 3D-simplicial complexes* the process of aggregating simplices of identical dimensions in order to obtain a more compact, yet geometrically accurate representation of the data. We expect that this representation can be used for guiding 3D reconstruction algorithms. In particular, we study the generalization of the 2D simplices (i.e. the triangles) of our simplicial complexes. We first investigate different ways of representing the geometry of urban areas. Then, we present our work on the piecewise-planar approximation of 3D point clouds, including the *ℓ_0-plane pursuit* algorithm. The motivation behind this algorithm is that each triangle of our *simplicial complex-based* reconstruction is part of a larger planar structure. Thus, we simplify the reconstructed meshes of our reconstructions as a set of planar regions. We evaluate this approach on MLS, TLS and ALS data.

4.1 Geometric Representation of Urban Areas

This section investigates different techniques used for the geometric representation of urban areas. Urban scenes have specificities, as they combine large and simple objects, such as roads or facades, with small and intricate objects like traffic signs. This means that, in order to approximate an urban scene with a high geometric fidelity, one should be able to adapt the degree of simplification to the geometric regularity of the objects considered.

We distinguish three different types of approach: procedural approaches (Haala and Kada, 2010), plane-fitting approaches, and segmentation-based approaches. For a more extensive review of current approaches for representing the geometry of urban scenes, we refer the reader to Dore and Murphy (2017).

4.1.1 Procedural Approaches

A first approach to the reconstruction of urban areas is to set rules to charactetize dominant or repetitive features in the data (Milde et al., 2008; Becker and Haala, 2009; Lafarge et al., 2008; Toshev, Mordohai, and Taskar, 2010). These rules are especially useful in the context of building reconstruction: a building can be seen as a main facade with a set of doors and windows. The doors must touch the floor and cannot exceed a certain size. The windows are usually regularly spaced out and have similar sizes, especially when they lay on the same floor. Using these simple rules, one can easily represent a geometrically simple building. Vanegas, Aliaga, and Beneš (2010) used Manhattan rules to reconstruct buildings: the authors state that the majority of objects in urban scenes are structured along three main orthogonal directions. Tutzauer and Haala (2015) combined the geometry and radiometry of a scene to reconstruct facades. Ponciano, Trémeau, and Boochs (2019) showed that reinforcement learning techniques could be used to improve a procedural classification approach. Probabilistic grammars can also be used for texturing buildings (Li et al., 2018).

4.1.1.1 Grammar-Based Approaches

The split grammars (Wonka et al., 2003) are a first simple grammar. It consists of a set of rules splitting the data along a single feature dimension at a time. This approach has been used for building reconstruction based on footprints (Larive and Gaildrat, 2006) or LiDAR scans (Wan and Sharf, 2012). Martinovic and Van Gool (2013) convert a set of input images to lattices in order to decrease the complexity of the

problem. They then define and learn an optimal 2D split grammar and demonstrate that the generated grammar is as good as human-generated grammars. Some work even proposed interactive tools to edit the proposed reconstruction (Demir, Aliaga, and Benes, 2016). Teboul et al. (2011) extended this concept to binary split grammars, where a rule used to split walls and floors is represented as a set of smaller, simpler rules that split the data in two parts. This allow the grammar to be more adaptive.

4.1.1.2 Sampling-Based Approaches

Markov Chain Monte Carlo (MCMC) have been used for shape analysis (Talton et al., 2011), leading to new reconstruction algorithms (Dick, Torr, and Cipolla, 2002; Huang, Brenner, and Sester, 2013). Tran and Khoshelham (2019) uses reversible jump Markov Chain Monte Carlo (rjMCMC) to reconstruct interiors of buildings as a way to find the configuration of 3D scenes using planar primitives extracted by a RANSAC. Their algorithm rely on probabilities that each cell defined by primitive extraction is part of the building or not. rjMCMC has also been useful for building facade reconstruction (Brenner and Ripperda, 2006; Ripperda and Brenner, 2009).

4.1.1.3 Classification-Based Approaches

Support Vector Machine (SVM) have been used to generate a grammar (Tsochantaridis et al., 2004). Dehbi et al. (2017) used a SVM to generate a weighted context-free grammar which is then used with a *Markov Logic Network* to recreate buildings from combined images and point clouds.

4.1.1.4 Neural Networks

Recently, some work started using NN to learn grammars and reconstruct buildings (Zeng, Wu, and Furukawa, 2018) or smaller objects (Sharma et al., 2018). Nishida et al. (2016) proposed a method for generating 3D buildings from sketches. Kelly et al. (2017) used a CNN to extract windows, doors, and facades corners and use a global data fusion step to re-create a full city.

In our case, we consider large sets of 3D points, without semantic information. We also want our method to be completely agnostic to external informations or user-based knowledge, as we believe it is better for scalability and adaptivity to different / unkown data. For these reasons, we operate directly on the points' coordinates and do not use grammar-based methods in this thesis.

4.1.2 Primitive-Based Approaches

4.1.2.1 Least Square-Fitting

The least squares method is the standard approach for fitting parametric models on 3D data. It consists in selecting the models for which the sum of square distance between the points and the model is minimal. This method has been used extensively to find relevant primitives to approximate 3D point clouds as set of planes (Xie et al., 2003; Dzitsiuk et al., 2017). Tatavarti, Papadakis, and Willis (2017) partition their data into cubical regions and use a least square algorithm to fit a plane in each cell. This approach has also been used to reconstruct facades from 3D point clouds (Martinovic et al., 2015).

4.1.2.2 RANSAC

The RANSAC algorithm (Fischler and Bolles, 1981; Schnabel, Wahl, and Klein, 2007) is one of the most popular method for extracting planes in 3D point clouds. Given a dataset composed of points, this method works by iteratively fitting a model to a random subset of the data. Points that are well-approximated by the model are called *inliers*, while those who are not are labeled as *outliers*. After each iteration, the *inliers* are removed from the data and the algorithm continues to process the *outliers*. In the case of piecewise-planar approximation of 3D data, the model is simply a plane equation. This approach has been thoroughly investigated in the literature (Asvadi et al., 2016; Dzitsiuk et al., 2017). Holzmann et al., 2017 used a RANSAC-based approach for plane extraction in point clouds, and a reconstruction algorithm based on extracted planes to build a Manhattan-like structure. Moreover, Xu et al., 2015 introduced a weighted RANSAC approach with a soft threshold voting function. It allows them to lower the number of spurious planes and improve segmentation quality.

4.1.2.3 Local Geometry-Based

Another way of finding planes in 3D data is to use the planarity feature as detailed in Demantke et al. (2011). The planarity of a point is determined with the eigenvalues of the co-variance matrix of the point's neighbors coordinates. Chauve, Labatut, and Pons (2010) use a planarity feature to detect planes. Boulch, La Gorce, and Marlet, 2014 order points according to their local planarity to seed a region growing algorithm. In Ma et al. (2013), the authors find planes' normals by selecting points with the lowest curvature, and iteratively growing regions from these points.

4.1.2.4 Line-Based Approaches

Planes can be interpreted as sets of non-parallel lines: two lines that are not parallel form the basis of a plane. Holzmann et al. (2018) propose to find planes using sets of orthogonal lines that are close enough to be considered part of a same object. Their method allows them to detect less spurious planes in urban areas.

4.1.2.5 Neural Networks

Finally, recent work proposed to train a neural network to fit primitives, either from an unsupervized (Sharma et al., 2018; Tulsiani et al., 2017) or a supervized method (Zou et al., 2017). Li et al. (2019) introduced SPFN, a neural network able to detect various primitives in point clouds. Their network first find pointwise properties and use a differential model estimator to generate primitives.

4.1.3 Segmentation Approaches

One of the main challenges after extracting planes in a 3D point cloud, is to find the limits of the region in the data supported by the considered plane. For some approaches, like the RANSAC algorithm, the 3D shape of each region can be easily retrieved thanks to the set of *inliers* and *outliers*. However, this is not the case for all plane extraction methods. Some methods propose to compute large planar arrangements to find the the limits of each planar regions (Nan and Wonka, 2017). However, such planar arrangements are usually computationally heavy if the number of primitives exceeds a few dozens. In order to limit the computational complexity of the problem, it can be interesting to combine plane extraction with point cloud segmentation.

We now focus on segmentation approaches for 3D point clouds by presenting different methods either from a local point of view, such as region growing, or from a more global point of view.

4.1.3.1 Region Growing

Region growing is one of the most prevalent segmentation approaches for the segmentation of 3D data (Cohen-Steiner, Alliez, and Desbrun, 2004; Whelan et al., 2015; Fang, Lafarge, and Desbrun, 2018). Vo et al. (2015) use an adaptive octree on which they grow regions, based on each voxel's geometric features. They argue that their approach is more robust than standard region-growing-based segmentations, and that it improves segmentation results. The RAPter algorithm (Monszpart et al., 2015), a reconstruction algorithm based on regular arrangements of planes, is initialized using a region growing-based over-segmentation of the input point cloud. This algorithm takes into account plane primitives and inter-plane relations. Whelan et al., 2015 argue that a local curvature-based region growing algorithm can outperform RANSAC-based approaches for segmenting point clouds. Region-growing has also been coupled with a regularization step based on simple relationships between extracted primitives (coplanarity, parallelism and orthogonality) in Oesau, Lafarge, and Alliez (2016).

4.1.3.2 Mean-Shift

One of the main other approaches for segmenting 3D data is the Mean-shift algorithm (Fukunaga and Hostetler, 1975; Cheng, 1995). It seeks maxima of a density function, given discrete data samples. Thus it is appropriate for LiDAR data processing. This method has the advantage of being non-parametric. Dai et al. (2018) and Ferraz et al. (2012) use a mean-shift method for individual tree segmentation from airborne LiDAR point clouds.

4.1.3.3 Voxel-Based Approaches

LiDAR data are usually large and hard to process: a single building can contain more than 100,000 points. Hence, it is relevant to propose to arrange points along a 3D grid structure (Douillard et al., 2011; Maturana and Scherer, 2015a), where each cell is called a *voxel*. We can then process each voxel separately. The main advantage of using voxels is to decrease computation time. This allows to compute computationally expensive features at a more global scale, directly on voxels (Boerner, Hoegner, and Stilla, 2017; Plaza-Leiva et al., 2017). These features can then be used for solving segmentation (Xu et al., 2017) or semantic segmentation (Maturana and Scherer, 2015b; Tchapmi et al., 2017) problems. Zhou and Tuzel (2018) learned the features and object bouding boxes to improve semantic segmentation results. Poux and Billen (2019) use a voxel-based approach for the semantic segmentation of 3D point clouds, in which per-voxel descriptors are computed to identify geometric continuity between adjacent voxels.

The main drawback when using voxels is that all the geometric details smaller than 1 voxel are lost. This means that there is a trade-off between computationnal efficiency and geometric fidelity that is usually dependent on the scene.

4.1.3.4 Graph-Based Approaches

Point clouds can be structured by the adjacency relationship between points, such as the relationship defined by nearest neighbors, a triangulated mesh, or a 3D Delau-

nay triangulation (Boissonnat, 1984). This allows us to represent the cloud as a graph, of which the vertices are the 3D points. As a benefit, efficient graph-structured clustering methods can be used, typically relying on graph-cuts (Klasing, Wollherr, and Buss, 2008; Strom, Richardson, and Olson, 2010). Landrieu and Obozinski (2017) introduced the ℓ_0-cut pursuit algorithm, a greedy working-set method for computing piecewise constant approximation of signals on such graphs, and proved its efficiency on point clouds in Landrieu et al. (2017). Dutta, Engels, and Hahn, 2018 adapted the Normalized Cut algorithm (Shi and Malik, 2000), an algorithm originally designed for perceptual grouping in raster images, to segment 3D LiDAR point clouds.

One of the main dangers when clustering point clouds is to either lose information on areas with high geometric complexity, or to over-segment the cloud. Thus, some work focus on over-segmentation techniques to then merge the smaller segments and generate different models at different LoDs (Attene and Patanè, 2010; Lejemble et al., 2018). Nan and Wonka (2017) construct an arrangement of planes from which a manifold polyhedral surface model without boundary is extracted, which can be seen as a piecewise planar approximation.

For a more complete state-of-the-art on primitive extraction and segmentation, especially in a 3D reconstruction context, we refer the reader to Berger et al., 2017.

4.1.4 Evaluation of 3D Models of Urban Scenes

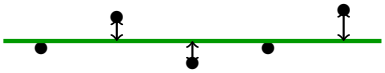
Last, evaluating the performances of such method is not a straightforward task. Historically, the evaluation has often been visual (Durupt and Taillandier, 2006). Some work focused on a voxel-based evaluation (McKeown et al., 2000; Schuster and Weidner, 2003). By contrast, Yu, Helmholz, and Belton (2016) evaluated grammar-based methods directly in 3D, by comparing it to a manual reconstruction of the processed scene. Recently, Ennafii et al. (2019) proprosed to train a classifier to detect modelling errors. To this end they compute various geometric and optical features. In this chapter, we focus only on the segmentation process, and the final reconstruction will be evaluated in Chapter 5.

4.2 Piecewise-Planar Approximations

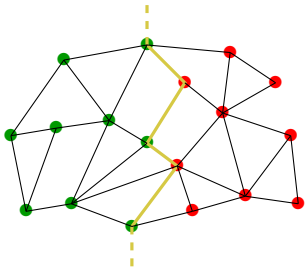
In this section, we present the ℓ_0-*plane pursuit algorithm* developed during this thesis. This algorithm is used for the piecewise-planar approximation of 3D point clouds, and is an adaptation of the ℓ_0-cut pursuit algorithm (Landrieu and Obozinski, 2017).

4.2.1 Graph Structure

We denote V the set of 3D points in the input data, characterized by their position $p_v \in \mathbb{R}^3$. We consider that a good planar approximation of our data should contain as few planes as possible, while remaining geometrically faithful to the original data. As we are working with urban areas containing many man-made objects—which often have simple shapes—and that we are seeking compact representations, we favor simple interfaces between planes as well.



(a) Illustration of the data term: minimization of the distance between each point (black circles) and its associated plane (shown as a green line).



(b) Illustration of the regularization term: creating regions with simple shapes. Green points and red points belong to two different regions. The yellow line shows the boundary between these regions.

Figure 4.1 – Representation of our model: we aim at minimizing the distance between points and their associated plane, while favoring simple interfaces between regions associated with the same planes..

4.2.1.1 Weighted Adjacency Graph

We use a weighted oriented graph structure $G = (V, E, w)$ to represent our data, where $E \subset V \times V$ characterizes the adjacency between each point and $w \in \mathbb{R}_+^E$ stands for the edge weight, encoding the proximity between points.

If the input 3D data already has a graph structure (for instance if it is a mesh), we can directly use this structure as adjacency graph. If the data is an unstructured point cloud, there are several possibilities for building an adjacency graph structure i.e. defining which points are neighbors: k-nearest neighbors, triangulated meshing, 3D Delaunay triangulation, among others. We propose to use the same graph adjacency as the one introduced in Chapter 3, which exploits sensor topology to connect scanned points with edges and triangles and leaves isolated points unconnected. We create an edge in G if and only if the two vertices belong to at least one shared triangle. Indeed, isolated points and edges of the simplicial complex correspond to parts of the scene where the geometry is not even locally surfacic. In such areas, the geometry is too complex relative to scanning resolution to define a local tangent plane. These areas include tree foliage or linear structures for which planar approximation would not be appropriate. This acts as a prefiltering step in combination to providing the adjacency structure.

We choose edge weights which decrease with the distance between points:

$$w_{u,v} = \frac{1}{\alpha + \frac{d(u,v)}{d_0}} \ , \tag{4.1}$$

where d_0 is the mean length of edges in the adjacency graph, and α a non-negative constant taken here as 2.

4.2.2 Problem Formulation

We denote the set of all planes of $\mathbb{R}^3$ as $\mathcal{P}$. We denote by $d(v, \pi)$ the euclidian distance between a vertex v and a plane $\pi \in \mathcal{P}$. We associate to each vertex v a plane π_v and denote by $\Pi \in \mathcal{P}^V$ the set of planes thus defined. Such plane set defines an approximation of V characterized by the projection of each vertex to its associated plane.

We propose an energy $E : \mathcal{P}^V \to \mathbb{R}$ whose minimization gives a precise yet simple piecewise-planar approximation of V. For $\Pi \in \mathcal{P}^V$, the energy writes as follows:

$$E(\Pi) = \underbrace{\sum_{v \in V} d\,(\,v, \pi_v\,)^{\,2}}_{Data\ term} + \mu \underbrace{\sum_{(u,v)\in E} w_{u,v}\,[\,\pi_u \neq \pi_v\,]}_{Regularizer}\,, \tag{4.2}$$

where $[\pi \neq \pi']$ is the function of $\mathcal{P}^2 \mapsto \{0, 1\}$ equal to 0 when π and π' are identical planes, and 1 otherwise. The parameter $\mu \in \mathbb{R}_+$ is the regularization strength defining the trade off between data fidelity and regularization. The first term of Equation 4.2 corresponds to the fidelity term. This term ensures that each point is well approximated by its corresponding plane. The second part of Equation 4.2 encourages adjacent vertices to share the same planes. It also forces the interface between adjacent planes to have simple shapes by penalizing cuts, i.e. edges between adjacent vertices with different planes. Therefore, a set of planes with low energy should achieve a tradeoff between shape complexity and precision of their approximation of the input point cloud. Note that our choice of w (see Equation (4.1)) favors cuts between distant points, insuring that vertices associated to the same plane are generally close from one another. The data term and regularization term are illustrated on figure 4.1.

We define the set of approximating planes as the result of the following optimization problem:

$$\Pi^\star = \operatorname*{arg\,min}_{\Pi \in \mathcal{P}^V} E(\Pi)\,. \tag{4.3}$$

The energy E is non-convex, non-continuous, and non-differentiable, and hence is hard to minimize. However, it has a similar structure than the energy defined by the *generalized minimal partition problem* introduced by Landrieu and Obozinski (2017) to compute piecewise constant approximation of multidimensional signals on graphs. This working set algorithm iteratively splits a partition $\mathcal{V}$ of the graph G into disjoint constant connected components. It also keeps track of the adjacency structure of the components.

We first present the *generalized minimal partition problem* before explaining the principle of the ℓ_0-cut pursuit algorithm.

4.2.3 Generalized Minimal Partition Problem

Let $G = (V, E, w)$ be a graph structure, similarly to the one defined in the previous section. Now, instead of planes, we associate a multidimensional signal $f_v \in \mathbb{R}^d$ to each node V. A signal-homogeneous partition of V is defined as the constant connected components of the solution of the following optimization problem: [24]

$$g^\star = \operatorname*{arg\,min}_{g \in \mathbb{R}^{d\times V}} \sum_{v\in V} \|g_v - f_v\|^2 + \mu \sum_{(u,v)\in E} w_{u,v}[g_u - g_v] \neq 0\,, \tag{4.4}$$

where $[x-y]$ is the function of $\mathbb{R}^{d\times 2} \mapsto \{0, 1\}$ equal to 0 when $x = y$, and 1 otherwise. The problem defined here is called the *generalized minimal partition problem*.

This problem is similar to the one presented in Equation 4.2. Regarding the data-fitting term, Equation 4.2 aims at minimizing the squared distances between each point

24. The Generalized Minimal Partition Problem is actually defined with an arbitrary fidelity function as long as it is separable with respect to V. For simplicity's sake, and to reinforce the analogy with our plane-approxiamtion problem, we write the Generalized Minimal Partition Problem with the square difference.

and their associated plane, whereas in Equation 4.4, the goal is to mimimize the sum of squared differences between an approximated signal g and an original signal f. In the case where f is the positions of 3D points, then the data-fitting term of the *generalized minimal partition problem* can be viewed as a geometric fidelity between points. For the regularizing term, both problems aim at enforcing small and simple interfaces between adjacent regions. This is done by summing edges weights at the interface between adjacent regions. In Equation 4.4, this is done by comparing adjacent point's approximated vector of descriptors: this vector is identical for points in a same region. In Equation 4.2, this is done by comparing the supporting planes of the neighborhing points: if the two points have identical supporting planes, then they belong to the same region.

The functional considered in Equation 4.4 is non-convex and non-continuous, hence hard to minimize for large 3D point clouds. However, Landrieu et al. (2017) argue that the ℓ_0-cut pursuit algorithm is able to find an approximate solution with only a few graph-cuts. Due to the similarity between the *generalized minimal partition problem* and ours, we investigated how to adapt the ℓ_0-cut pursuit algorithm to our setting.

4.2.4 ℓ_0-cut pursuit Algorithm

The ℓ_0-cut pursuit algorithm (Landrieu and Obozinski, 2017, Section 3.1.4) proceeds in a top-down manner, iteratively splitting a current partition $\mathfrak{P} = \{A_1, \dots, A_k\}$ in finer components. This partition is initialized with the trivial partition $\{V\}$ comprised for which all points are in the same region. At each iteration, a refining step splits each component into smaller parts through a graph-cut formulation. Then, a backward step computes a new approximation of f with respect to the updated component, and proceed to merge pairs of adjacent components for which this fusion decreases the objective energy, in the manner of Soussen et al. (2011). The algorithm stops when no components can be further refined nor merged. While this algorithm does not guarantee to find a global optimum of the generalized minimal partition problem, it is in practice able to find good approximate solutions in a few iterations only.

4.2.4.1 Split Step

In this step, each region $A_i \in \mathfrak{P}$ is divided in two sub-regions $B_i \subset A_i$ and $A_i \setminus B_i$. For each region A_i, Landrieu and Obozinski (2017) propose the following formulation:

$$\min_{B_i \subset A_i} \min_{(h_j, h'_j)} \sum_{v \in B_i} \|h_j - f_v\|^2 + \sum_{v \in A_i \setminus B_i} \|h'_j - f_v\|^2 + \lambda \sum_{(i,j) \in E_{B_i}} w_{i,j} \ , \tag{4.5}$$

with $E_{B_i} = (B_i \times A_i \setminus B_i \cup A_i \setminus B_i \times B_i) \cap E$ the set of edges between two regions B_i and $A_i \setminus B_i$ of V.

The first part of Equation 4.5 corresponds to a data fidelity term. The second part of Equation 4.5 is a regularization term. It corresponds to the value of the cut between B_i and $A_i \setminus B_i$. The authors show that this formulation can be generalized to V thanks to the separability of the problem with respect to $\mathfrak{P}$.

This optimization problem, defined with respect to both continuous and set variables, is hard to solve exactly. Hence, Landrieu and Obozinski (2017, Section E.1.1) propose an alternated minimization scheme alternating between (h, h') and B_i. They first initialize the values of h and h' with the 2-means algorithm set to approximately solve the following problem:

$$\underset{B \subset A_i, h, h'}{\arg\min} \sum_{v \in B_i} f_v(h) + \sum_{v \in A_i \setminus B_i} f_v(h') \ . \tag{4.6}$$

With these values set, the authors now solve Equation 4.5 with respect to B_i:

$$\underset{B \subset A_i}{\arg\min} \sum_{v \in B_i} \|h_j - f_v\|^2 + \sum_{v \in A_i \setminus B_i} \|h'_j - f_v\|^2 + \lambda \sum_{(i,j) \in E_{B_i}} w_{i,j} \ . \tag{4.7}$$

This problem can be solved using an efficient MinCut / MaxFlow solver, such as the one of Boykov, Veksler, and Zabih (2001). Figure 4.2a shows an illustration of this step.

After minimizing Equation 4.7, the values of h and h' can be re-estimated:

$$\begin{cases} h \leftarrow \arg\min_x \sum_{v \in B_i} \|x - f_v\|^2 \ , \\ h' \leftarrow \arg\min_x \sum_{v \in A_i \setminus B_i} \|x - f_v\|^2 \ , \end{cases} \tag{4.8}$$

The authors argue that this alternating minimization scheme converges to a local minima in a finite number of iterations (see Landrieu and Obozinski (2017, Section E.1.2) for a complete proof).

In some cases, this alternating minimization scheme can lead to some trivial cut, where $B_i = A_i$ or $B_i = \varnothing$. In this case, splitting A_i results in no modification of the partition $\mathfrak{P}$. Hence, this component is not updated.

4.2.4.2 Backward Step

In their paper, Landrieu and Obozinski argue that their method can be improved by allowing the mergeing of adjacent components when it decreases the functional defined in Equation 4.4. In fact, when iteratively splitting the original data, one may obtain a final partition composed of numerous small regions, with similar values of f. In this case, it may be benefical to merge some of these regions. A simple merge step consist in merging two adjacent components, while associating to each the optimal constant approximation of f of the merged region:

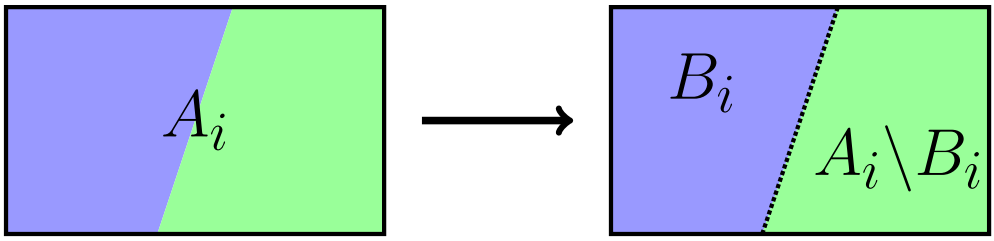
$$g^m \leftarrow \underset{h}{\arg\min} \sum_{v \in \cup(B_i, B_j)} f_v \ . \tag{4.9}$$

Landrieu and Obozinski (2017) consider that two adjacent subregions should be merged if and only if the increase of the fidelity of term associated to mergeing all the vertices of the considered is smaller than the cut between them:

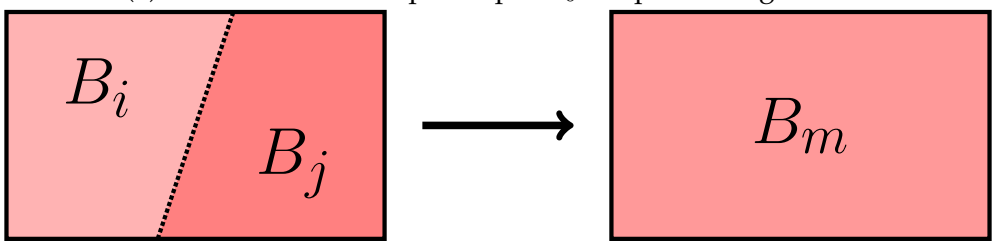
$$\sum_{v \in \cup(B_i, B_j)} \|h_m - f_v\|^2 < \sum_{v \in B_i} \|h_i - f_v\|^2 + \sum_{v \in B_j} \|h_j - f_v\|^2 + \lambda \sum_{(i,j) \in E_{B_{i,j}}} w_{i,j} \ , \tag{4.10}$$

with $E_{B_{i,j}} = (B_i \times B_j \cup B_j \times B_i) \cap E$ the set of edges between B_i and B_j. The merge step is showned on Figure 4.2b.

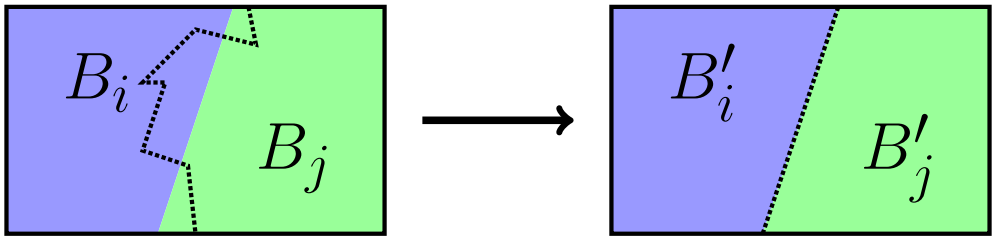
This strategy can be further improved by using a *merge-resplit* strategy. This consists in re-estimating the boundary between two adjacent subregions. It is dones by merging the two subregions and splitting them again as defined in Section 4.2.4.1. This step is represented on Figure 4.2c.



(a) Illustration of the split step of ℓ_0-cut pursuit algorithm.



(b) Illustration of the merge step of ℓ_0-cut pursuit algorithm.



(c) Illustration of the merge-resplit step of ℓ_0-cut pursuit algorithm.

Figure 4.2 – Illustration of the different steps of the ℓ_0-cut pursuit algorithm. The color corresponds to the value of the signal f to approximate. The signal is represented as an image, and the graph structure is the 4-adjacency between pixels (not represented for clarity). The area in green, blue, and red are different values of the signal on this graph. Dotted lines show the boundaries between adjacent regions.

4.2.4.3 Applications

ℓ_0-cut pursuit has been used for segmenting point clouds, as a way to regularize a pointwise classification (Landrieu et al., 2017). This is described in Chapter 6. This algorithm has also been used for point cloud sparsification (Tenbrinck, Gaede, and Burger, 2019) or piecewise horizontal roof segmentation (Namouchi et al., 2019). This algorithm is also used for the geometric partition of SuperPoint Graph (Landrieu and Simonovsky, 2018; Landrieu and Boussaha, 2019).

4.2.5 ℓ_0-plane pursuit Algorithm

Our problem differs from the generalized minimal partition problem since the variables associated with each vertex are planes instead of multidimensional signals. However, we can follow a very similar scheme to the ℓ_0-cut pursuit algorithm to minimize E. We added an adapted initialisation step and modified the split and merge steps of the ℓ_0-cut pursuit algorithm, but retained to a similar approach. These contributions are listed hereafter:

— **Initialization:** to minimize the effect of a bad initialization, we start by extracting planes using the RANSAC algorithm. Let $E_R(k)$ be the error associated to the RANSAC-based segmentation after drawing Π_k, a set of k planes such that $|\Pi_k| =$

k. We have:

$$E_R(k) = \sum_{v \in \mathcal{V}} \min_{\pi \in \Pi_k} d(v, \pi)^2 \, . \tag{4.11}$$

We stop the RANSAC algorithm after k_0 iteration, when the decrease in error provided by adding an extra plane, relative to the error after drawing a single plane, is under a given threshold τ_R (chosen here to 0.005):

$$\frac{E_R(k-1) - E_R(k)}{E_R(1)} < \tau_R \, . \tag{4.12}$$

— **Refining:** in the ℓ_0-cut pursuit algorithm, components are split according to the optimal binary partition criterion (see Equation 4.7). Adapted to our setting, this amounts to finding the binary partition $(Bi, A_i \setminus Bi)$ of a component $A_i \in \mathfrak{P}$ defined as follow:

$$\underset{B_i \subset A_i, (\pi, \pi') \in \mathcal{P}^2}{\arg\min} \sum_{v \in B_i} d(v, \pi) + \sum_{v \in A_i \setminus B_i} d(v, \pi') + \mu \sum_{(i,j) \in E_{B_i}} w_{i,j} \, , \tag{4.13}$$

with $E_{B_i} = E \cap (B_i \times A_i \setminus B_i \cup A_i \setminus B_i \times B_i)$ the set of edges linking B_i and $A_i \setminus B_i$. This step can be approximately solved by an alternated minimization scheme adapted from the original ℓ_0-cut pursuit algorithm. We replace the initialization step from 2-means to a 2-plane extraction step using the RANSAC algorithm. The optimization with respect to B_i can be performed efficiently through a graph-cut formulation, while the optimization with respect to π and π' is done through least square minimization. The scheme of this algorithm is detailed in Algorithm 3. We acknowledge that this optimization scheme may need many iterations to converge to a local minimum. In practice, we observe that `ite_split` $= 3$ iterations of this scheme are sufficient.

— **Backward step**: as explained in Section 4.2.4.2, the ℓ_0-cut pursuit algorithm benefits from allowing the mergeing of adjacent components as long as it decreases the global energy. This can be easily adapted to our setting: to estimate if two adjacent components should be merged, a common plane is computed through a least square minimization, and the resulting increase in fidelity error is compared to the decrease in penalty.

To summarize our algorithmic scheme, we introduce the following subroutines:

— $(\pi_1, \cdots, \pi_k) \leftarrow$ `RANSAC`(U, k): takes $U \subset V$ a set of vertices and k a number of planes as input, and returns a set of k planes extracted by the RANSAC algorithm.

— $(\pi_1, \cdots, \pi_k) \leftarrow$ `fit`$(\mathcal{U})$: takes a set of k disjoint components $\mathcal{U} = \{U_1, \cdots, U_k\}$ as input and returns $(\pi_1, \cdots, \pi_k)$ a set of k planes fitting each vertex set U_i according to the least square criterion.

— $\mathcal{V} \leftarrow$ `associate`$(\pi_1, \cdots, \pi_k)$: takes a set of k planes as input and return a partition $\mathcal{V} = \{U_1, \cdots, U_k\}$ of V such that each component U_i contains all the points for which the plane π_i is the closest according to the euclidian distance.

— $\mathcal{U}' \leftarrow$ `connected_components`$(\mathcal{U})$: takes $\mathcal{U}$ a set of disjoint components as input and and returns the union of their connected components with respect to graph G.

Algorithm 3 $\mathcal{U} \leftarrow \texttt{split}(U)$

$(\pi, \pi') \leftarrow \texttt{RANSAC}(U, 2)$
for `ite_split` iterations **do**
  $B \leftarrow \arg\min_{B \subset U} \sum_{v \in B} d(v, \pi) + \sum_{v \in U \setminus B} d(v, \pi') + \mu \sum_{(i,j) \in E_B} w_{i,j}$
  $(\pi, \pi') \leftarrow \texttt{fit}(\{B, U \setminus B\})$
end for
$\mathcal{U} \leftarrow \texttt{connected\_components}(\{B, U \setminus B))$
return $\mathcal{U}$

— $\mathcal{U} \leftarrow \texttt{merge}(U, U')$: takes two components U and U' adjacent in G and returns $U \cup U'$ if mergeing them is profitable with respect to the energy E, or $\{U, U'\}$ otherwise.

We summarize our method in Algorithm 4. The first for-loop corresponds to the split step and the second one to the merge step. We draw attention on the fact that the merge step is optional, but helps reducing the number of regions in the partition as well as decreasing the global energy.

Note that our implementation of ℓ_0-plane pursuit do not use the *merge-resplit* strategy. If we want to add this step, we can replace the *merge* step in Algorithm 4 by the *merge-resplit* function.

Algorithm 4 ℓ_0-plane pursuit

$(\pi_1, \cdots, \pi_{k_0}) \leftarrow \texttt{RANSAC}(V, k_0)$ % Initialization
$\mathcal{V} \leftarrow \texttt{associate}(\pi_1, \cdots, \pi_{k_0})$
$\mathcal{V} \leftarrow \texttt{connected\_components}(\mathcal{V})$
while not_converged **do**
  for $U \in \mathcal{V}$ **do**
    $\mathcal{U} \leftarrow \texttt{split}(U)$ % Component splitting
    $\mathcal{V} \leftarrow \mathcal{V} \setminus U \cup \mathcal{U}$
  end for
  for U, U' adjacent in G **do**
    $\mathcal{U} \leftarrow \texttt{merge}(U, U')$ % Component merging
    $\mathcal{V} \leftarrow \mathcal{V} \setminus \{U, U'\} \cup \mathcal{U}$
  end for
end while
return $\mathcal{V}$, $\texttt{fit}(\mathcal{V})$

The ℓ_0-plane pursuit algorithm presented in Algorithm 4 returns a partition $\mathcal{V} = (U_1, \cdots, U_k)$ of V as well as a set of planes $(\pi_1, \cdots, \pi_k)$. The vertices in U_i are approximated by their projection in the plane π_i. The plane set Π, such that $\pi_v = \pi_i$ if and only if v is in U_i, is an approximate minimizer of the energy E.

We want to draw attention on the fact that our formulation doesn't have any assumption on the planes' orientation and number, nor on the number of vertices per region. This allows us to produce a segmentation that is adaptive to the local geometry of the cloud, by creating more regions in complex parts of the scene while keeping a simple approximation with a small number of planes for large planar parts, such as roads or facades.

4.3 Experiments

In this section, we test the ℓ_0-plane pursuit algorithm on data from MLS, TLS and ALS. We compare this method to a classic *region growing*-based baseline as in Cohen-Steiner, Alliez, and Desbrun (2004).

For our experiments, we used two different graph structures. When the sensor topology is available, we used the weighted adjacency graph as defined in Chapter 3. This is motivated by the fact that we can filter out all the 0- and 1-simplices of the *simplicial complex*-based reconstructions, and only keep the triangles from them. For the other cases, we decided to use a 3D Delaunay Triangulation, and more specifically, the implementation proposed in the CGAL library (Devillers, Hornus, and Jamin, 2019), as it gives a representation independent from the sampling.

4.3.1 Datasets

We used 3 different datasets. The Paris dataset is from MLS, the chapel dataset has been acquired with TLS and the Barcelona one has been acquired with ALS. Some technical informations about each sensor is provided in table 4.1.

4.3.1.1 Paris Dataset

This dataset has been acquired with the Stereopolis vehicle (Paparoditis et al., 2012) in the streets of Paris. It is the same dataset than the one used in Chapter 3. We limited the dataset to a small portion corresponding to a 1 second acquisition and just kept the triangles created by our simplicial complexes. This limitation is due to the fact that the data was cut in small pieces for our *simplicial complexes* reconstruction, for computing simplicity. However, this implies that each zone has a different reconstruction that is not necessarily consistent with the others. Thus, we decided to restrain our tests on a single block of data. It is composed of $218,546$ points and $418,254$ triangles. The scene is mostly composed of a street and large facades of Hausmannian buildings.

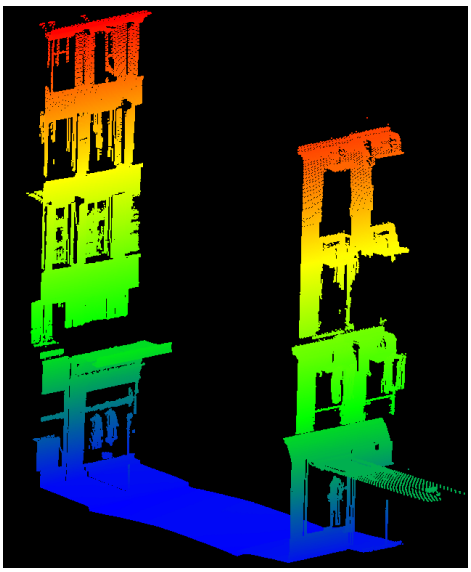


Figure 4.3 – View of Paris dataset. For vizualisation purposes, we displayed the height for each point.

4.3.1.2 Chapel Dataset

This dataset has been acquired by the Centre de formation ENSG-Forcalquier with a TLS inside a chapel. We just kept the triangles created by our simplicial complexes. It is composed of $1,263,321$ points and $7,313,760$ triangles. The scene is the inside of chapel. It is especially interesting as the chapel contains vault portions and we wanted to test our algorithm on vaults to see how it performs. The scene has also a higher density than the previous one.

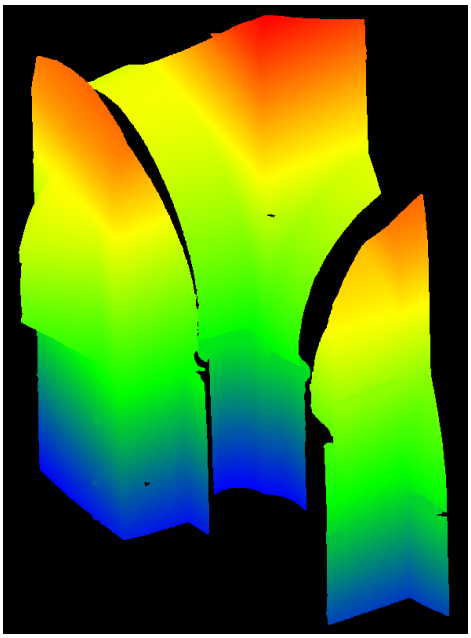


Figure 4.4 – View of Chapel dataset. For vizualisation purposes, we displayed the height for each point.

4.3.1.3 Barcelona Dataset

This dataset has been acquired by the *Institut Cartogràfic i Geològic de Catalunya*[25] with an ALS above Barcelona. The data is the concatenation of several flights above the city, with different directions and densities. Thus we could not apply our *simplicial complex*-based reconstruciotn. That's why we decided to structure it with a 3D Delaunay triangulation. More preciselly, we used the CGAL implementation (Devillers, Hornus, and Jamin, 2019). Due to the unability to filter out linear and volumetric regions, we kept a very noisy point cloud. It contains $500,000$ points and $872,372$ triangles. The scene has been acquired above Barcelona and is composed of streets, roofs and a bit of vegetation. The experiments with this dataset are part of a **Volta** secondment[26] at *Institut Cartogràfic i Geològic de Catalunya*. The results are visible on Figure 4.12.

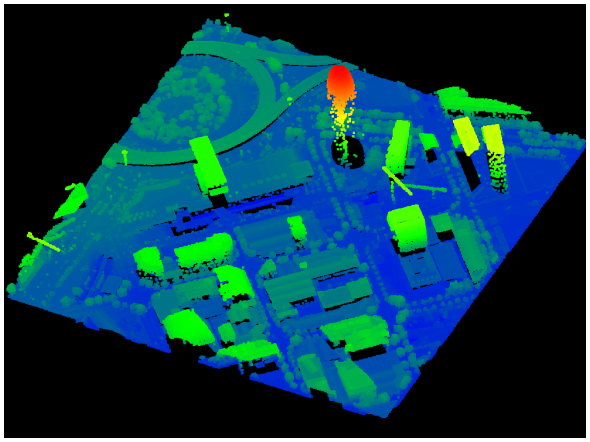


Figure 4.5 – View of Barcelona dataset. For vizualisation purposes, we displayed the height for each point.

26. Catalonia's mapping agency (`https://www.icgc.cat/en/`)
26. `https://volta.fbk.eu/index.php/about/`

Dataset	Paris	Chapel	Barcelona
Mean of acquisition	MLS	TLS	ALS
Sensor	RIEGL VQ-250	Leica ScanStation P40	Leica ALS50-II
Rotation speed (Hz)	100	50	35
Multi-echo	up to 8	No	up to 4
Density (pts/m^2)	high: > 100	very high: > 1000	low: 4
Number of points	218,546	1,263,321	500,000
Number of triangles	418,254	7,313,760	872,372
Structure	Triangles of (Guinard and Vallet, 2018a)	Triangles of (Guinard and Vallet, 2018a)	Delaunay Triangulation

Table 4.1 – Characteristics of each processed scene.

4.3.2 Baseline

We compare our results to our own implementation of the region-growing-based method of Cohen-Steiner, Alliez, and Desbrun (2004), which will serve as baseline. This method takes an input mesh and tries to segment it in planar regions. [27]

To achieve this, $\mathcal{N}$ triangles are chosen at random. The number of triangles has to be set by the user. Then, the algorithm grows a region for each triangle, noted $r_1, \ldots, r_\mathcal{N}$. This operations starts by sorting all triangles adjacent to the seeds, based their coplanarity with the corresponding adjacent seed. This is used to create a sorted list $\mathcal{L}_{adj}$ of triangles which contains all the triangles of the mesh adjacent to a region. The triangle which is the most coplanar to its neighborhing region r_i is added to this region, and is removed from $\mathcal{L}_{adj}$. Then, its neighbors which have not been yet considered are added to $\mathcal{L}_{adj}$ and the list is sorted again. This continues until a region has been associated to all triangles. Then, the best fitting plane for each region is computed, and a new seed per region is chosen. This seed is the triangle (of the corresponding region) that is the most coplanar to the new estimated plane. Then, all the triangles except the seeds are unassigned and a new growing process starts. This method is illustrated on figure 4.6.

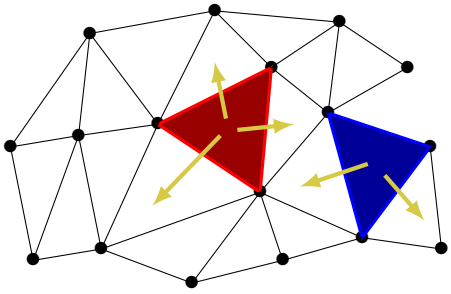


Figure 4.6 – Illustration of the first iteration region growing baseline. We select at random 2 triangles from the input mesh (red and blue triangles) and grow regions around them.

27. At the moment when this experiment was done, the CGAL package (Alliez, Cohen-Steiner, and Zhu, 2019) re-implementing this algorithm had not been released. However, we chose to keep the comparison to *our* implementation as it seems more fair to compare code written by the same person with the same level of optimisation.

4.3.3 Numerical Results

We also compare the baseline approach with our method on the outdoor scene. The baseline algorithm is run twice, for 5 and 15 iterations respectively. More iterations do not significantly improve the results. In order to be able to fairly compare our method to the baseline, we evaluate both methods with the same number of planes. Therefore, we first run our method with a given regularization strength producing a number of clusters. We then then parameterize the region growing baseline such that it produces the same number of clusters. We also decided to measure the influence of the merge step in our method by evaluating the performance of our algorithm with and without this step.

4.3.3.1 Metric

We want to measure the geometric precision of the reconstruction. To this end, we measure the distance between each point $v \in V$ and its supporting plane $\pi_v \in \mathcal{P}$. Therefore we set our error metric as the sum of distances between each point and its plane. We decided to use the L_2 norm over the L_1 norm as we want to heavily penalize points that are very far from their associated plane:

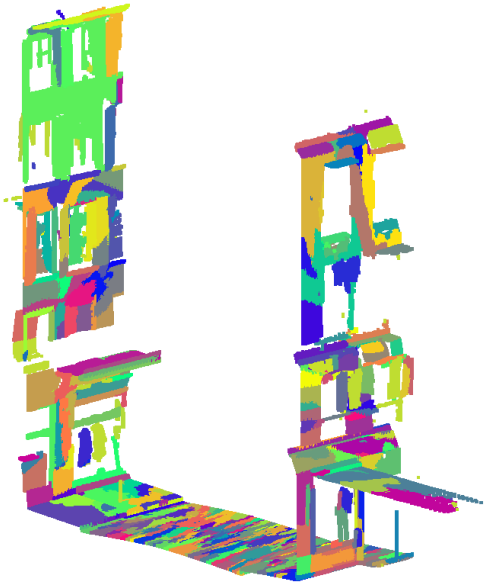
$$E = \sum_{v \in \mathcal{V}} d(v, \pi_v)^2. \tag{4.14}$$

4.3.3.2 Results on Paris Dataset

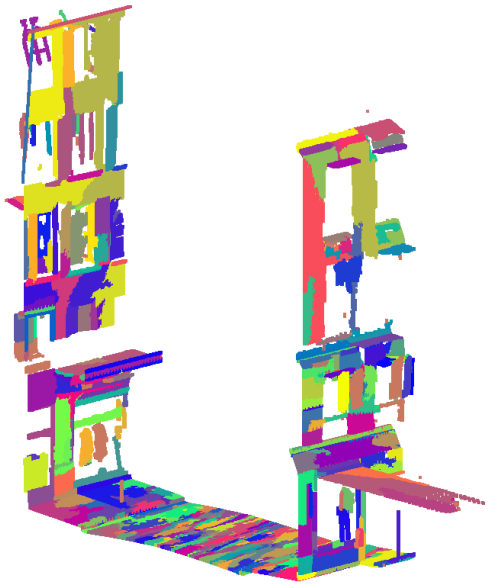
We represent the results on Figure 4.7. We see that the region-growing baseline forms regions of homogeneous size. This implies that simple and regular parts of the cloud, such as the road, will be clustered in several regions while a single plane would have made a good approximation. In contrast, our method adapts the size of the planes according to the geometric complexity of the scene, resulting in a smaller number of planes necessary to reach the same level of precision. This is the main reason for the large difference in terms of performance between the region-growing baseline and our method. This phenomenon can especially be seen on figure 4.8, which shows a geometrically simple part of the scene (a road portion) and a more complex part of the scene (the top of a facade). The detailed results for the error metric and the computation times are plotted in Figure 4.9. We see that our method is 10 times more geometrically accurate than the region growing baseline, but it is also 5 to 10 times faster. This is due to the global formulation of the problem, allowing us to run one single optimization for the whole cloud.

4.3.3.3 Results on Chapel Dataset

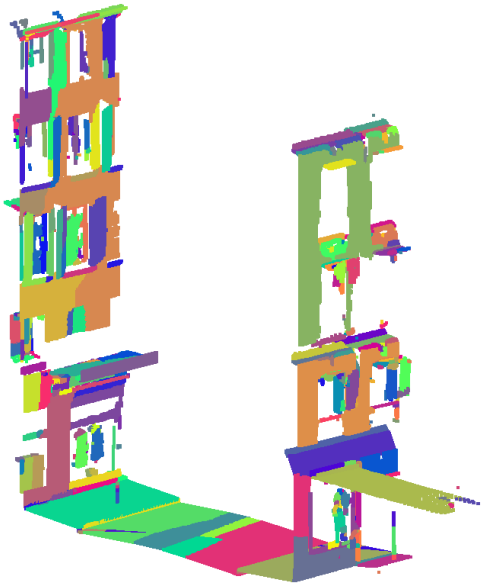
While we tried to compare our method with the baseline on the chapel dataset, we could not achieve convergence with the baseline in reasonable time due to the dataset's size. The results of our method on the chapel dataset are shown on Figure 4.10. Our expectations here, are that the vaults should be divided into small planar pieces elongated in the direction of minimum curvature. We observe that this is indeed what happens on Figure4.10. Our method is able to create large regions for large planar parts of the scene (e.g. walls) and its high adaptability allows it to create smaller regions to fit rounded parts of the scan. The detailed results are displayed on Figure 4.11. We can see that the use of the merge step decreases the number of regions, while improving the global approximation.



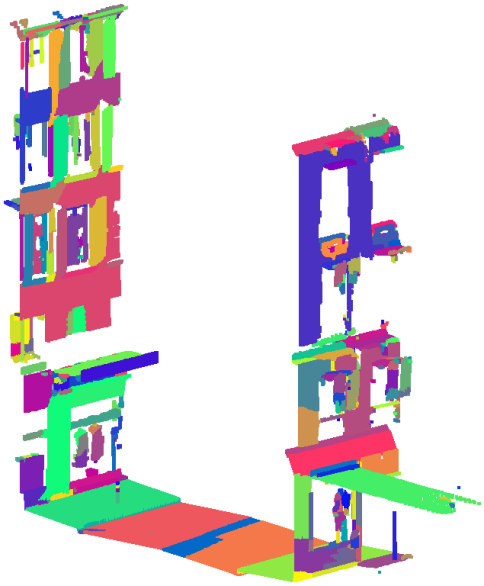
(a) Region Growing
492 components
5 iterations
error: $18.3 \cdot 10^3$



(b) Region Growing
492 components
15 iterations
error: $17.5 \cdot 10^3$

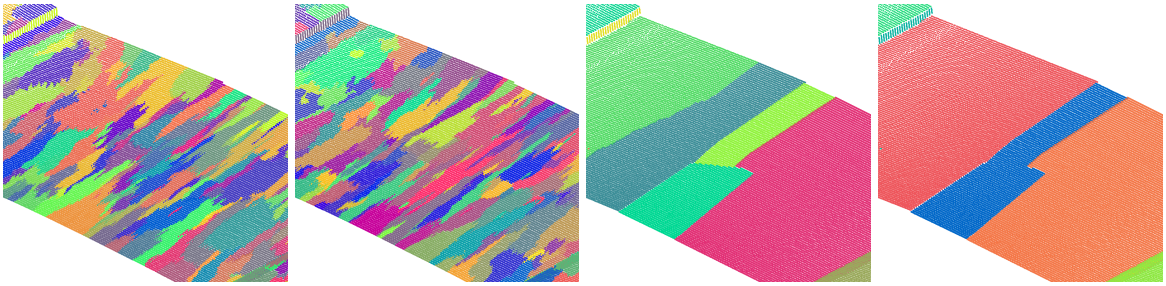


(c) ℓ_0-plane pursuit
no merge step
514 components
error: $1.6 \cdot 10^3$



(d) ℓ_0-plane pursuit
with merge step
492 components
error: $1.6 \cdot 10^3$

Figure 4.7 – Comparison of our method and the baseline. Each color represents a different region. Points are projected on the plane supporting their region. Our method creates large regions on simple parts of the cloud, in order to be more adaptive on complex parts.

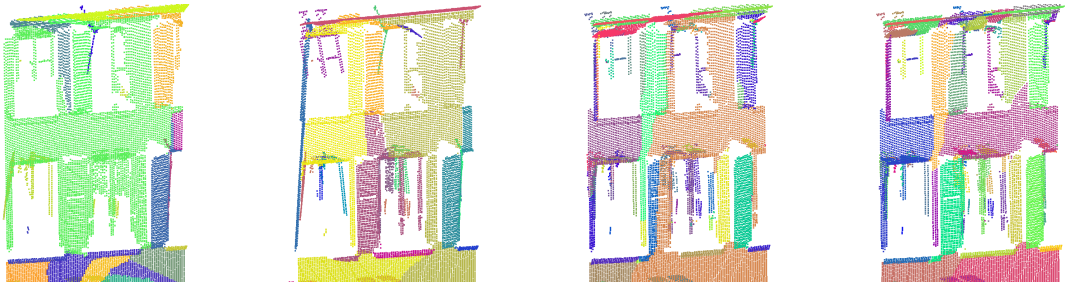


(a) Region Growing
492 components
5 iterations
error: $18.3 \cdot 10^3$

(b) Region Growing
492 components
15 iterations
error: $17.5 \cdot 10^3$

(c) ℓ_0-plane pursuit
no merge step
514 components
error: $1.6 \cdot 10^3$

(d) ℓ_0-plane pursuit
with merge stepd
492 components
error: $1.6 \cdot 10^3$



(e) Region Growing
492 components
5 iterations
error: $18.3 \cdot 10^3$

(f) Region Growing
492 components
15 iterations
error: $17.5 \cdot 10^3$

(g) ℓ_0-plane pursuit
no merge step
514 components
error: $1.6 \cdot 10^3$

(h) ℓ_0-plane pursuit
with merge stepd
492 components
error: $1.6 \cdot 10^3$

Figure 4.8 – Comparison of our method and the baseline. Each color represents a different region. Points are projected on the plane supporting their region. The first row shows some results on a road portion, while the second one shows the top of a building. Our method creates large regions on simple parts of the cloud and more regions on geometrically complex parts. This allows for limiting the number of primitives while preserving the geometric accuracy of the input scan.

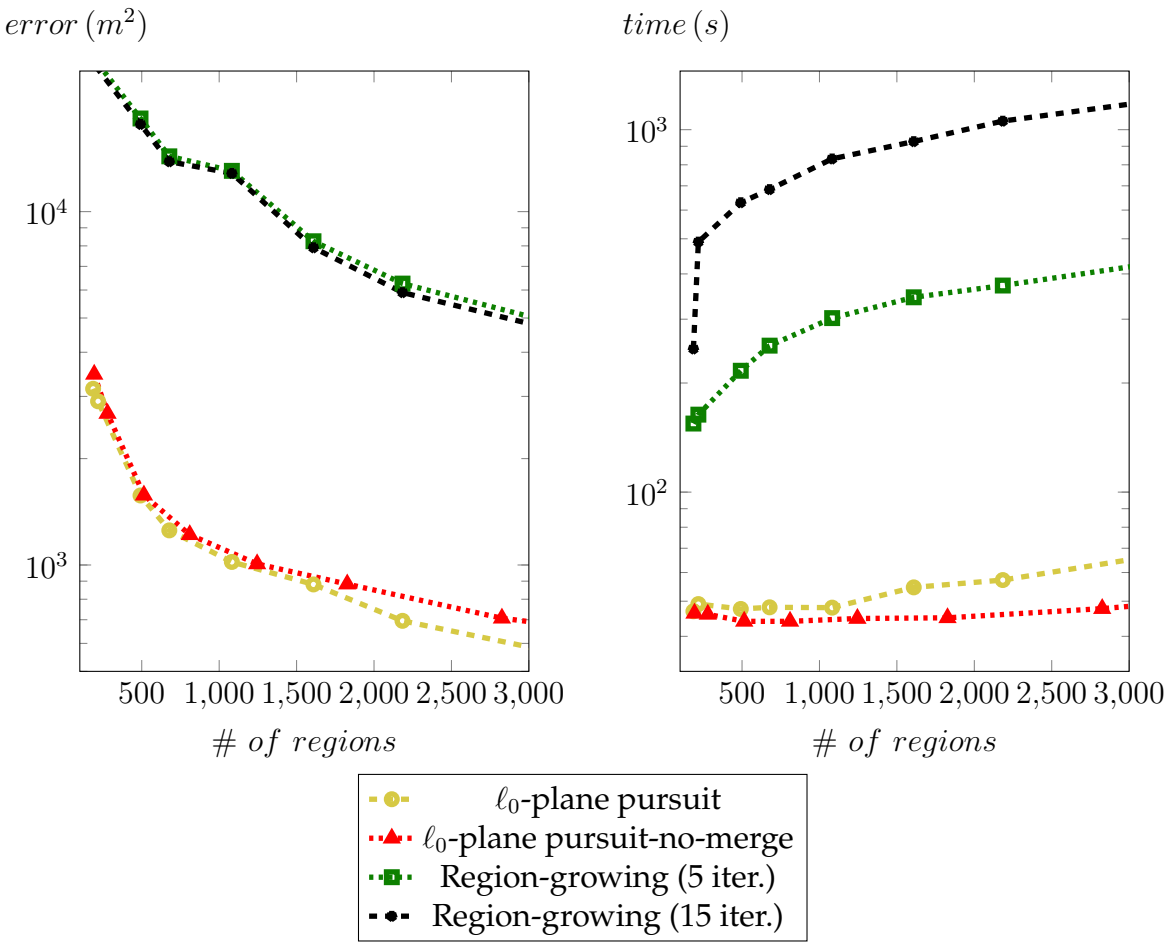
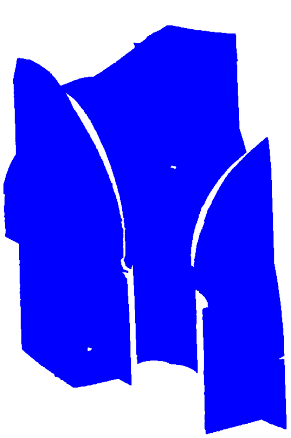
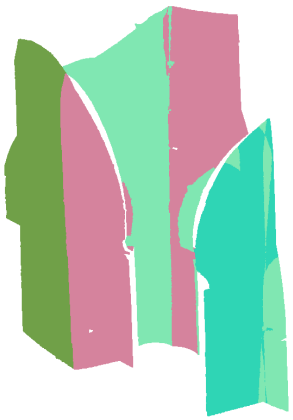


Figure 4.9 – Comparison of the error between our method and the region-growing baseline on the Paris dataset. For a given number of regions, our method has a higher geometric fidelity than the region growing baseline. Also, the graph-cut formulation shows improvments in terms of computation time, compared to the baseline.

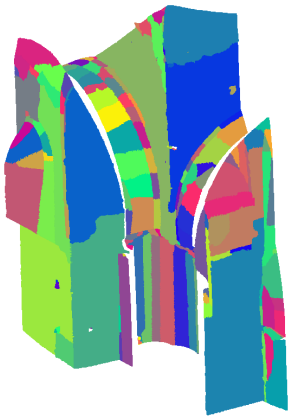


(a) Original point cloud

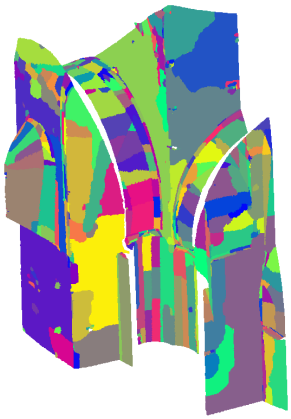


(b) RANSAC-based segmentation

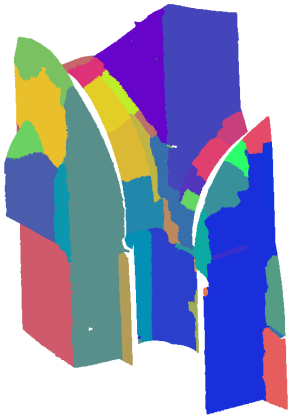
(c) ℓ_0-plane pursuit segmentation without merge step
Regularization strength: 0.1
error: $27 \cdot 10^3$



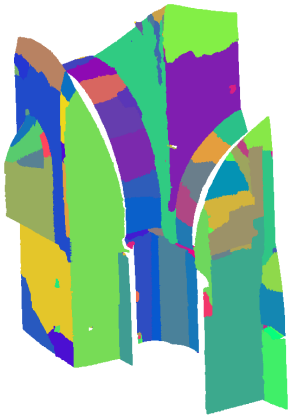
(d) ℓ_0-plane pursuit segmentation without merge step
Regularization strength: 0.01
error: $8 \cdot 10^3$



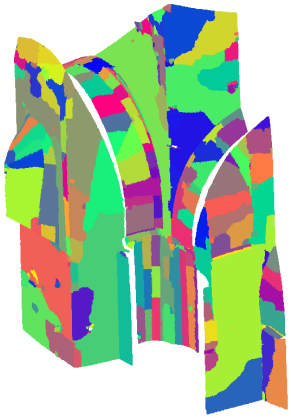
(e) ℓ_0-plane pursuit segmentation without merge step
Regularization strength: 0.001
error: $5 \cdot 10^3$



(f) ℓ_0-plane pursuit segmentation with merge step
Regularization strength: 0.1
error: $20 \cdot 10^3$



(g) ℓ_0-plane pursuit segmentation with merge step
Regularization strength: 0.01
error: $9.8 \cdot 10^3$



(h) ℓ_0-plane pursuit segmentation with merge step
Regularization strength: 0.001
error: $5 \cdot 10^3$

Figure 4.10 – Scan of the inside of the chapel cloud, composed of walls and vaults. Each color corresponds to a different region. The first row shows the raw point cloud and the RANSAC initialization. The second row is the ℓ_0-plane pursuit algorithm without the merge step and the last one is with the merge step. We remark that the use of the merge step decreases the number of regions without affecting the quality of the reconstruction.

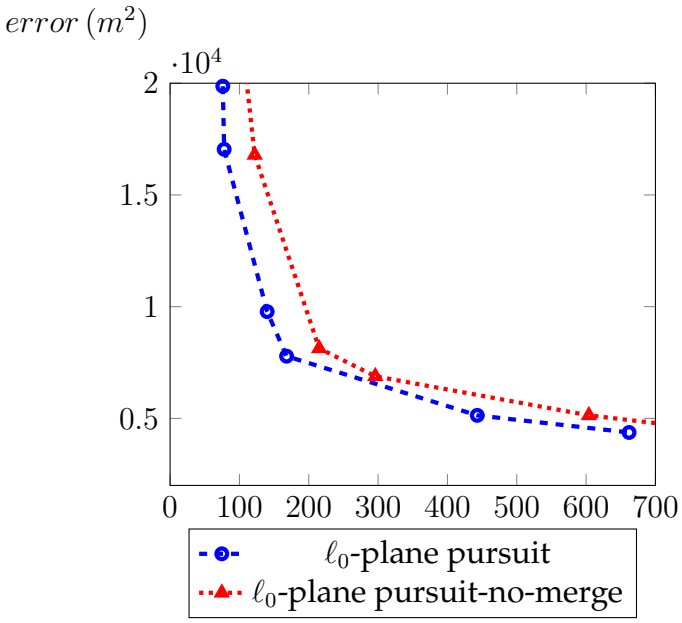
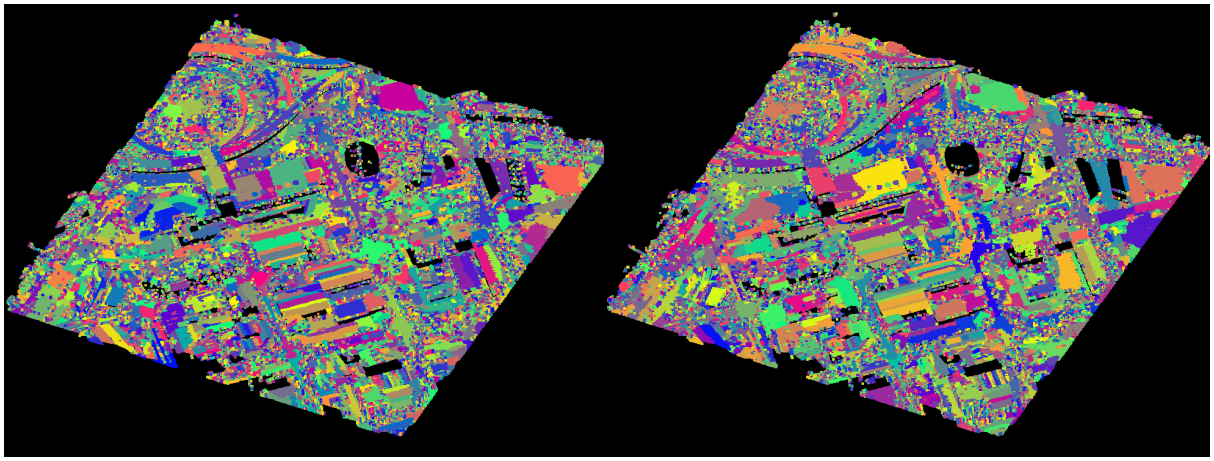


Figure 4.11 – Comparison of the error between our method with and without the merge step on the chapel dataset.

4.3.3.4 Results on Barcelona Dataset

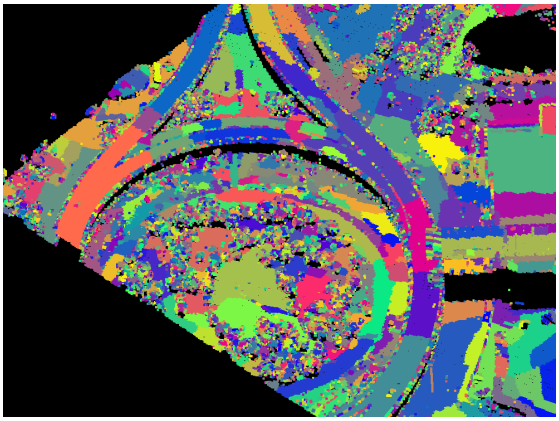
Last, we tested our method on the Barcelona dataset. This dataset is the resulting merge of several flights above the city of Barcelona, so it was not possible to reproduce our simplicial complexes reconstructions, as close points in the cloud may have been acquired by two different flights. We can see that the data is geometrically noisy. In order to limit the approximation error, we had to let our method oversegment the cloud, especially on vegetation areas. This large number of regions also highly increases the computation time. However, when we take a look on planar areas, such as road portions or roofs, we see that our algorithm performs well. Each roof is part of one or two regions, even small roofs that contain a few dozens of points, which is really small compared to the initial size of the data. This shows the adaptability of our method and its scalability.

The use of a semantic information (e.g. a classification), or simplicial complexes reconstruction would be useful to filter out non planar areas in this point cloud.

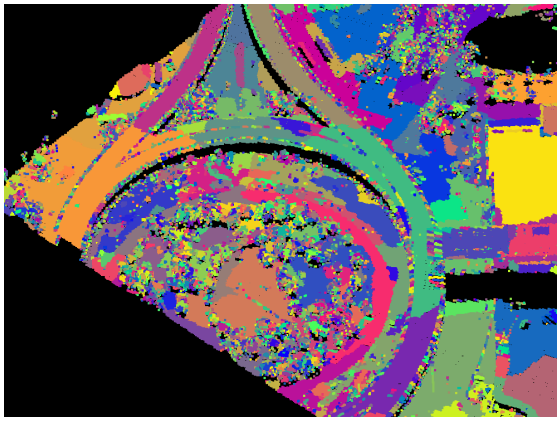


(a) Full results

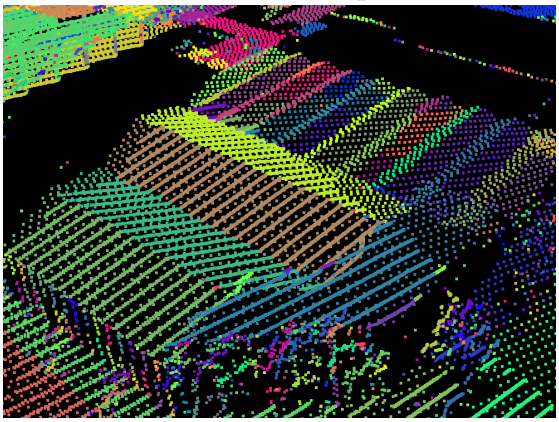
(b) Full results using reflectance



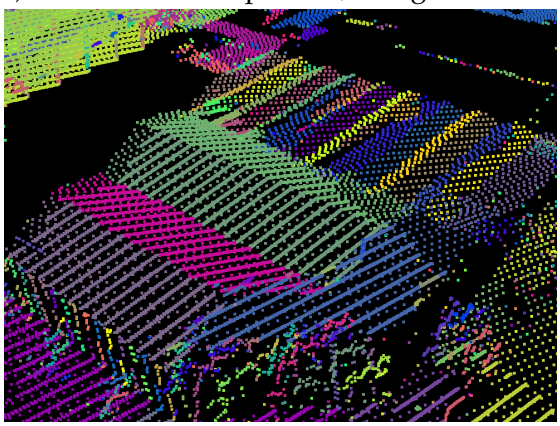
(c) Zoom on a road portion



(d) Zoom on a road portion, using reflectance



(e) Zoom on some roof portions



(f) Zoom on some roof portions, using reflectance

Figure 4.12 – Results on the Barcelona dataset. The first column shows results of the ℓ_0-plane pursuit algorithm. The second column shows results of a modified version of the algorithm where we used the Reflectance values of the scan too. The comparison between both methods is studied in Section 4.3.6. Each color represents a different region. The results are particularly noisy due to the presence of vegetation (which can't be simply approximated with a set of planes).

4.3.4 Influence of the Initialization

The goal of this part is to evaluate the influence of the initialization step, to see whethter the algorithm struggles to make the first split steps, or if it can achieve similar results without the initialization. We also want to see if the segmentation is good even if the initialization is voluntarily inaccurate.

Let $E(n)$ be the error related to the RANSAC-based segmentation at the n^{th} iteration. The criterion used to stop the RANSAC is the following:

$$\frac{E(n-1) - E(n)}{E(0)} > \epsilon, \tag{4.15}$$

with ϵ a threshold set by the user. This threshold is used as an indirect way of selecting the number of primitives retrieved by the RANSAC.

We fixed the regularisation strength of the optimisation process at 5×10^{-3}. We tested the influence of the initialisation on the Paris dataset, with ϵ varying between 0.5 and 5×10^{-4}. We also tested our algorithm with no initialisation and with a voluntarily bad initialisation: we divide the Paris dataset in ten parts regarding the points' acquisition order. A different labl is attributed to each region. This *bad initialisation* can be visualised on Figure 4.14m.

In this experiment, we expect our algorithm to recover from a *bad initialisation* and create output an accurate segmentation of the input point cloud thanks to the algorithm's strategy (alternating between split and merge steps). Some results are shown on Figure 4.14. Detailed results are available on Figure 4.13.

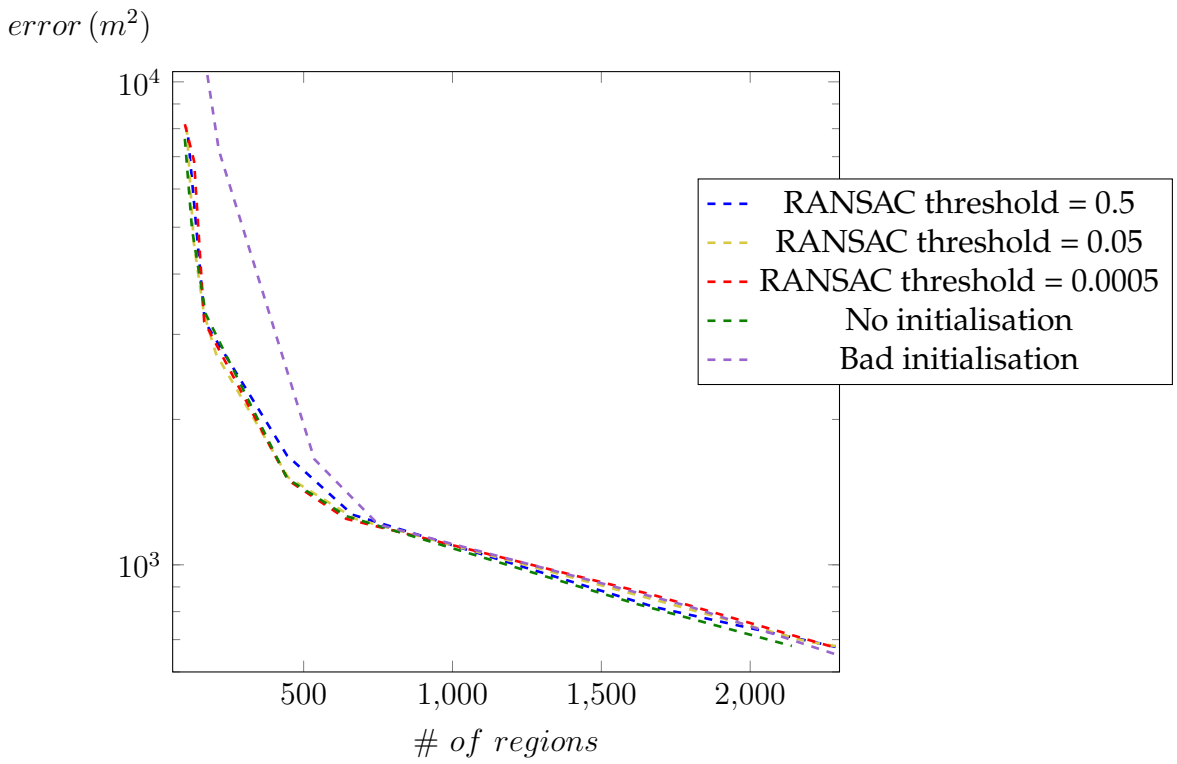
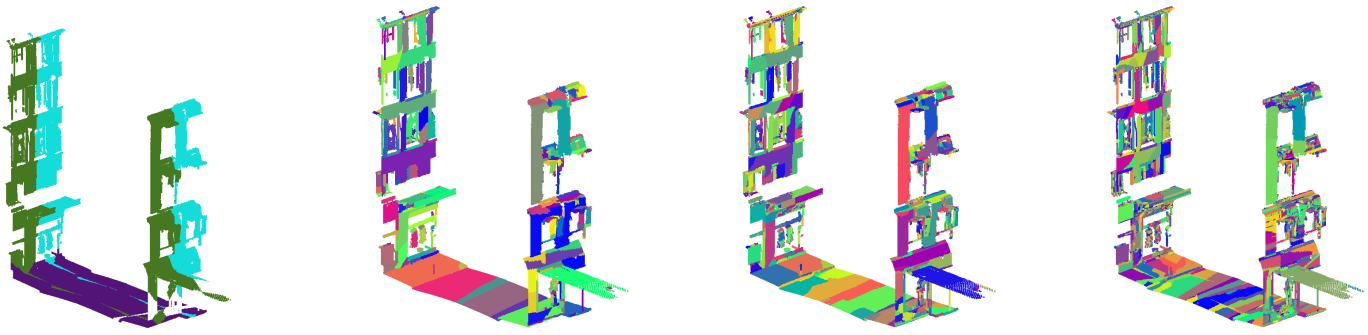


Figure 4.13 – Comparison of the error of our method depending of the quality of the initialisation on the Paris dataset.

We observe that our algorithm's performances are not directly correlated to the number of primitives retrieved by the RANSAC. Moreover, when we run the ℓ_0-plane pursuit algorithm with no initialisation, the results are still similar to the ones obtained

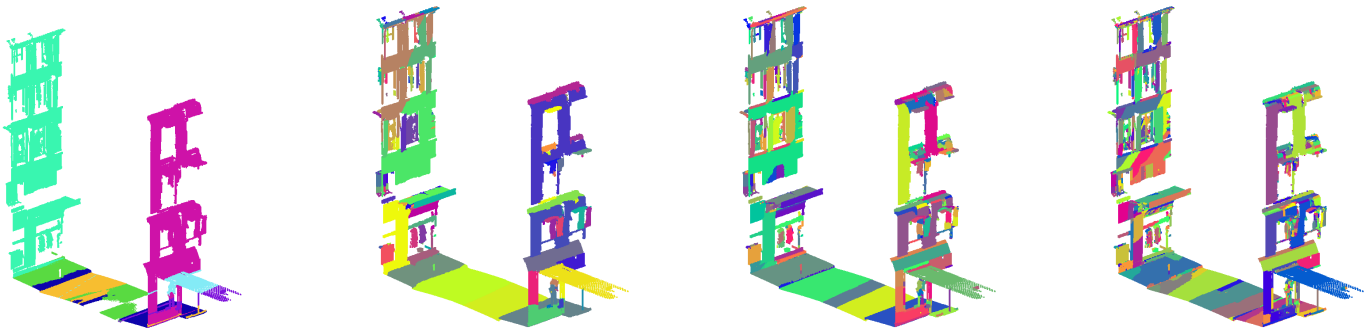


(a) RANSAC-based initialisation
Stopping criterion: $5 \cdot e^{-2}$

(b) RANSAC: $5 \cdot e^{-2}$
$\lambda = 10^{-3}$
165 regions
error: 3314.59 m^2

(c) RANSAC: $5 \cdot e^{-2}$
$\lambda = 10^{-4}$
448 regions
error: 1517.3 m^2

(d) RANSAC: $5 \cdot e^{-2}$
$\lambda = 10^{-5}$
1681 regions
error: 845.70 m^2

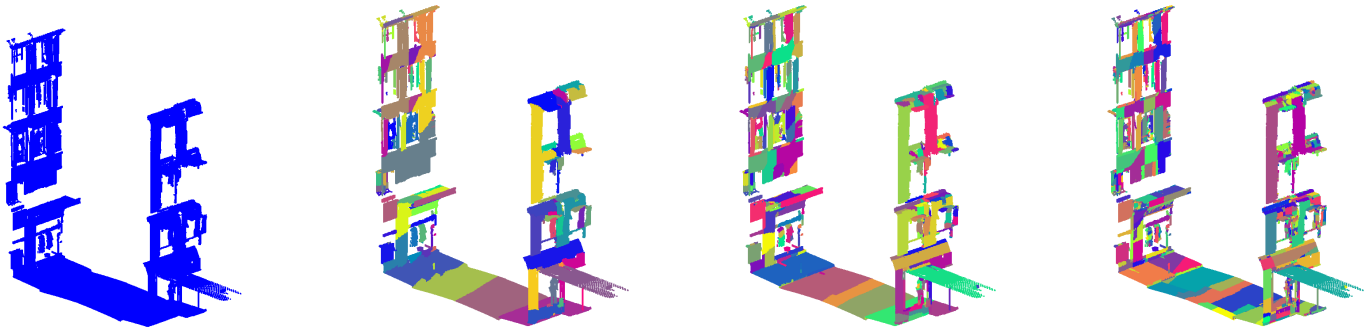


(e) RANSAC-based initialisation
Stopping criterion: $5 \cdot e^{-4}$

(f) RANSAC: $5 \cdot e^{-4}$
$\lambda = 10^{-3}$
166 regions
error: 3186.87 m^2

(g) RANSAC: $5 \cdot e^{-4}$
$\lambda = 10^{-4}$
448 regions
error: 1496.38 m^2

(h) RANSAC: $5 \cdot e^{-4}$
$\lambda = 10^{-5}$
1678 regions
error: 864.85 m^2

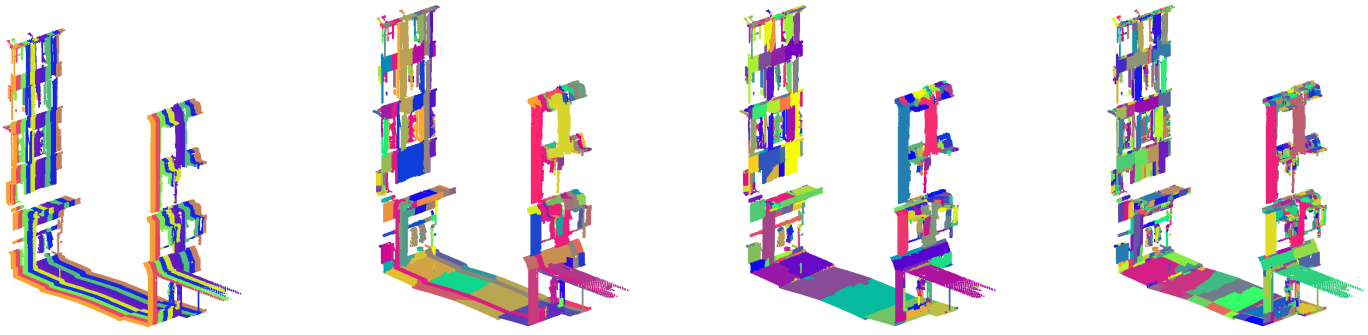


(i) No initialisation

(j) No init. $\lambda = 10^{-3}$
171 regions
error: 3306.24 m^2

(k) No init., $\lambda = 10^{-4}$
445 regions
error: 1508.45 m^2

(l) No init., $\lambda = 10^{-5}$
1618 regions
error: 829.81 m^2



(m) Bad initialisation with 10 regions

(n) Bad init., $\lambda = 10^{-3}$
215 regions
error: 7224.64 m^2

(o) Bad init., $\lambda = 10^{-4}$
536 regions
error: 1654.59 m^2

(p) Bad init., $\lambda = 10^{-5}$
1794 regions
error: 820.60 m^2

Figure 4.14 – Comparison of the results of $\ell - 0$-plane pursuit algorithm on the Paris dataset using different initialisations. The two first rows shows the results using a RANSAC-based initialisation with a stopping criterion of respectively: $5 \cdot e^{-2}$ and $5 \cdot e^{-4}$. The thirs row present results without any initialisation and the last one present the influence of a voluntarily bad initialisation. The first column is for a regularization strength of 10^{-3}, the second one for 10^{-4} and the last one for 10^{-5}. Each color represents a different region.

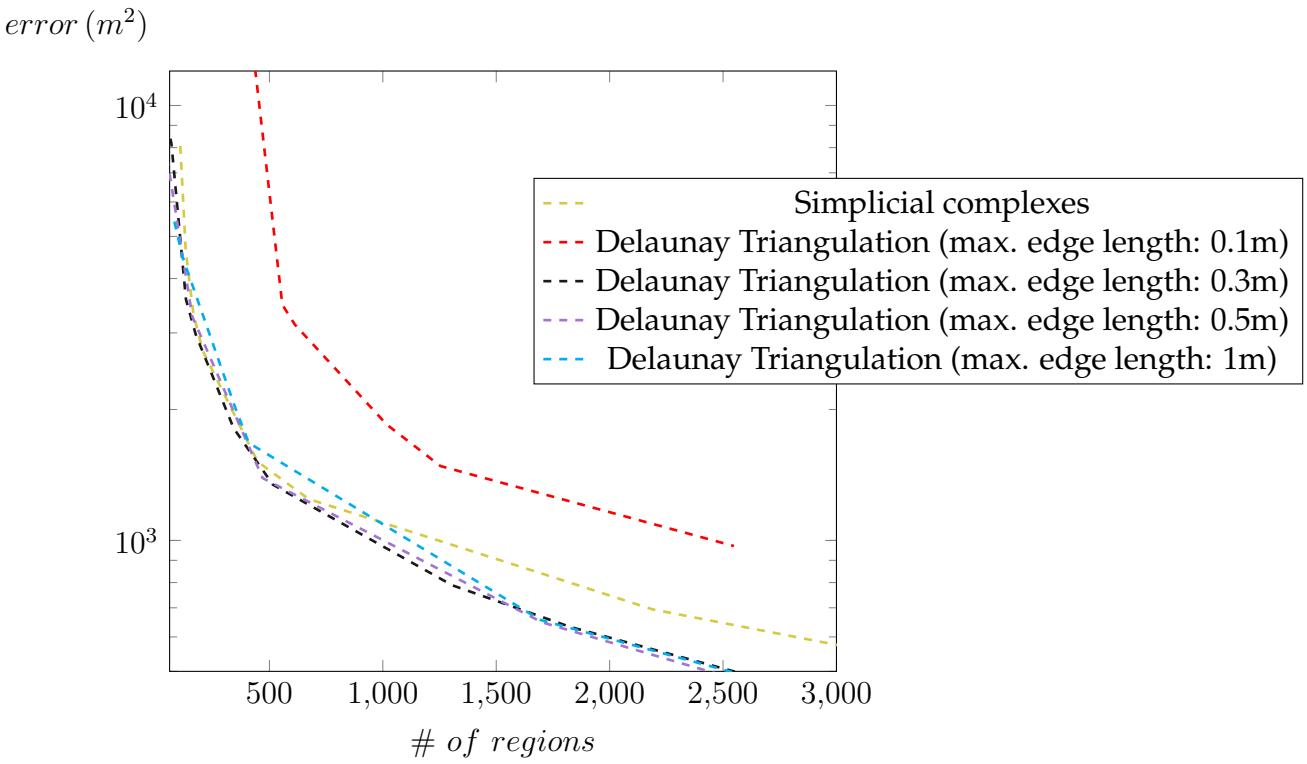


Figure 4.15 – Comparison of the error of our method structured with simplicial complexes and a Delaunay triangulation on the Paris dataset.

using an initialisation. This may sound counter-intuitive, as we are optimising on a non-convex energy that may have local minimas in which the algorithm could get stuck. But these performances shows the advantage of using an alternate optimisation scheme. Furthermore, even if the segmentation created by our algorithm, using a bad initialisation on purpose, may look worse than segmentations using a correct initialisation for low values of λ, our method is able to produce segmentations of similar quality for higher values of λ.

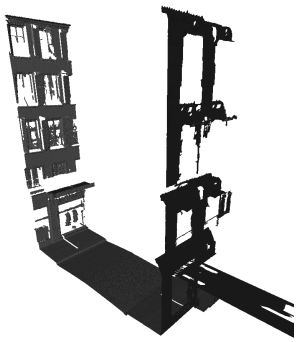
4.3.5 Influence of the Supporting Graph

In this part we discuss the influence of the supporting graph of the input point cloud. The idea is to see wether the sensor topology, when available, gives better results than oher classic graph structures, such as the *Delaunay triangulation*.

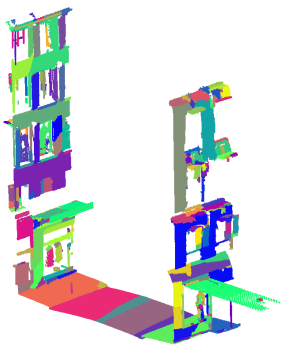
We compared our algorithm's performances using our *simplicial complexes* and a *Delaunay triangulation* using a filter based on edge length to remove faces between objects far from one another (like two facades on both sides ot the road). We chose four different filters: 1m, 0.5m, 0.3m and 0.1m. The RANSAC parameter has been fixed to 5×10^{-2}.

For the following experiments, we denote the *simplicial complexes* graph as Simplicial complexes (SC) and the *Delaunay triangulation* graph as Delaunay triangulation (DT).

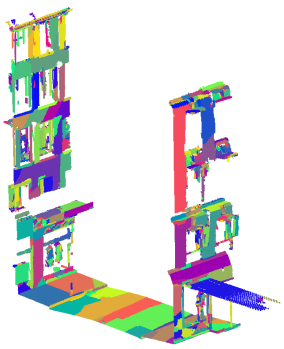
In this experiment, we expect our algorithm to create segmentations of similar quality of the best DT-based segmentations. Some results are shown on Figure 4.16. Detailed results are available on Figure 4.15.



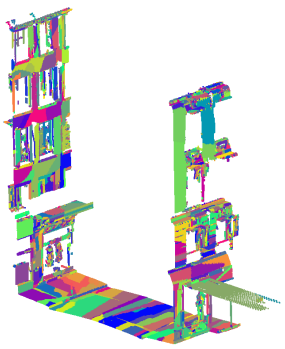
(a) Simplicial complexes mesh



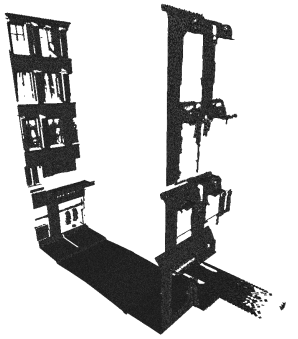
(b) SC mesh, $\lambda = 10^{-4}$
448 regions
error: 1517.3 m^2



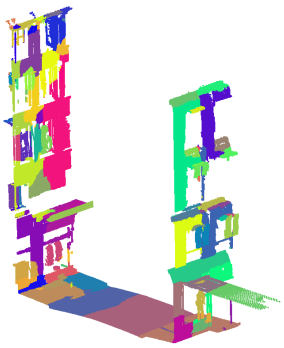
(c) SC mesh, $\lambda = 10^{-5}$
1681 regions
error: 845.70 m^2



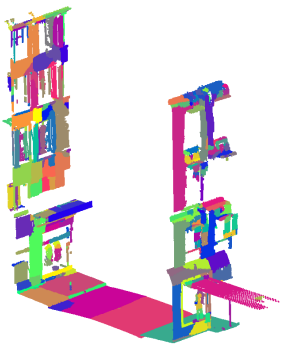
(d) SC mesh, $\lambda = 10^{-6}$
4084 regions
error: 447.61 m^2



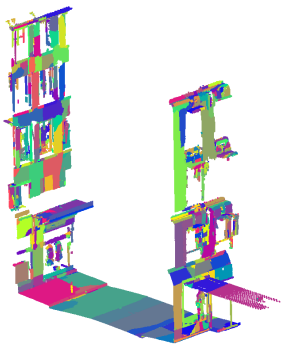
(e) Delaunay mesh, with a maximum edge length of 0.1 m



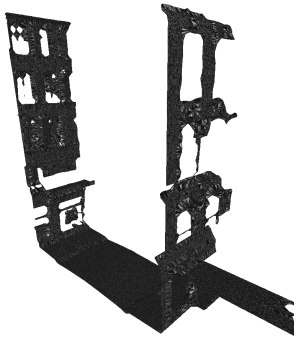
(f) DT mesh, $\lambda = 10^{-4}$
612 regions
error: 3135.33 m^2



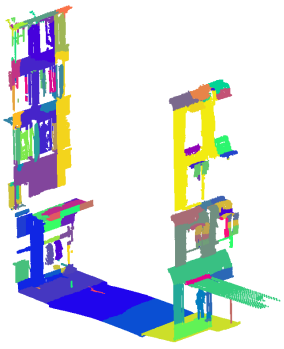
(g) DT mesh, $\lambda = 10^{-5}$
1017 regions
error: 1847.15 m^2



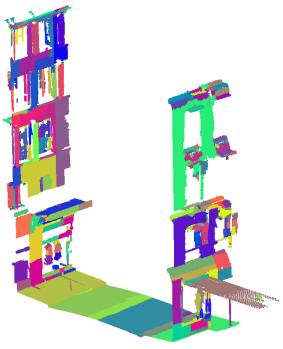
(h) DT mesh, $\lambda = 10^{-6}$
2017 regions
error: 1105.72 m^2



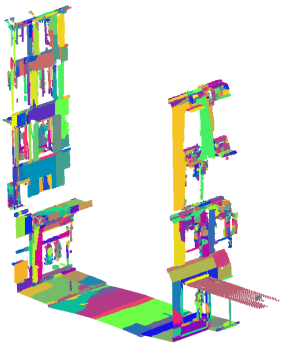
(i) Delaunay mesh, with a maximum edge length of 0.3 m



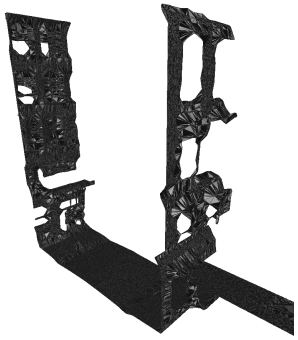
(j) DT mesh, $\lambda = 10^{-4}$
132 regions
error: 3582.08 m^2



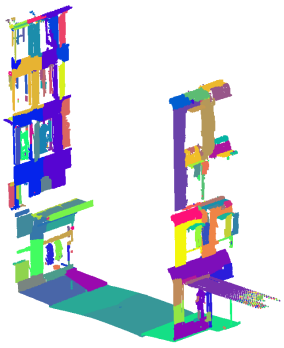
(k) DT mesh, $\lambda = 10^{-5}$
342 regions
error: 1812.53 m^2



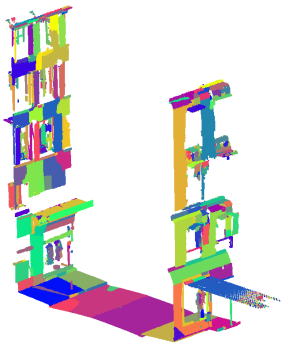
(l) DT mesh, $\lambda = 10^{-6}$
1306 regions
error: 788.92 m^2



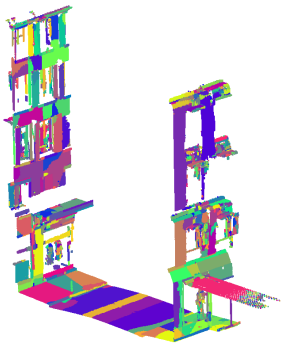
(m) Delaunay mesh, with a maximum edge length of 1 m



(n) DT mesh, $\lambda = 10^{-4}$
117 regions
error: 3135.33 m^2



(o) DT mesh, $\lambda = 10^{-5}$
349 regions
error: 2021.41 m^2



(p) DT mesh, $\lambda = 10^{-6}$
1153 regions
error: 826.676 m^2

Figure 4.16 – Comparison of the results of $\ell - 0$-plane pursuit algorithm on the Paris dataset using different graph structures. The first row shows results using our simplicial complexes. The second to fourth one shows the results using a Delaunay triangulation with a filter based on edge length of respectively: 0.1m, 0.3m and 1m. The first column is for a regularization strength of 10^{-4}, the second one for 10^{-5} and the last one for 10^{-6}. Each color represents a different region.

We observe that the use of *simplicial complexes* allow our method to compete with the best results obtained using a *Delaunay triangulation* for low values of λ. For higher values of λ, the use of *simplicial complexes* can produce slightly worse results according to our metric. However, if the *Delaunay triangulation* is not well parameterized (see the results for a filter based on a maximum edge length of 0.1m), the resulting segmentation can give poor results. The main advantage of *simplicial complexes* here is that they do not need user knowledge of the scene to allow the creation of accurate segmentations. Also, they are fatser to compute than a Delaunay triangulation at a scene scale.

4.3.6 Utilisation of the Reflectance for Segmenting Point Clouds with ℓ_0-plane pursuit

LiDAR sensors often acquire more than XYZ-coordinates. They usually add some *Reflectance* information. In this case, reflectance can give some useful insight on the texture of the real object behind each acquired point. LiDAR sensors such as TLS may be equipped with a camera as well. In this case, after the end of the laser scan, the camera acquires pictures of the whole scene. Then a color can be added to each point based on a visibility map of the scan.

Moreover, 3D descriptors can be computed for each point (Demantke et al., 2011; Weinmann et al., 2015a). These descriptors add strong insights on the local geometry of the data. They have been used for point cloud classification and segmentation (Guinard and Landrieu, 2017).

In this section we investigated the use of the reflectance acquired by an ALS for the piecewise-planar segmentation of LiDAR point clouds. For this purpose we modified the energy presented in equation 4.2 and tested the new algorithm on the Barcelona dataset.

4.3.6.1 Reflectance from LiDAR sensors

LiDAR sensors are not able to compute any color measurement. However, a complentary information is often available: the intensity. The intensity is computed as the ratio between the quantity of emitted light and the quantity of returned light for a given echo. This gives a first information on the encountered object structure and orientation: a planar object, perpendicular to the emitted light beam will reflect more light than an object nearly parallel to the beam and with a granular structure.

The intensity, has an important drawback: in the multi-echo case (as defined in Section 2.1.4.2), the intensity of echoes beyond the first one are greatly decreased. This is due to the fact that most of the light is reflected by the first echo, thus, following echoes of the same pulse receive very little light and reflect even less light.

If we consider that the object reflecting has a *lambertian* surface, we can compute its geometric reflectance using the Bidirectional Reflectance Distribution Function (BRDF) as defined by Nicodemus et al. (1977). We refer the reader to Kavaya et al. (1983) for an extensive explanation of the BRDF formulation. The BRDF is a function used to define how the light is reflected on an opaque surface based on the light direction (the laser beam in our case) and the normal of the surface. Real world surfaces reflects the light in a diffuse manner. To know how the reflected light is diffused, one has to use a *diffusion model*.

There exists several diffusion models, depending on the structure of the material. In computer graphics, most works assume that every material is locally lambertian (Oren and Nayar, 1994). A surface is lambertian when it reflects equally the light in every direction, independently of the source light direction.

Using reflectance information allow us to overcome the natural decrease in intensity in the multi-echo case. The use of reflectance information is especially useful for detecting vegetation (Wei et al., 2012) and mapping seafloor (Chust et al., 2010). In the case of roof segmentation, we argue that a given roof should be composed of the same material, thus having an homogeneous reflectance over its surface. We want to investigate the influence of this reflectance information for segmenting roofs.

4.3.6.2 Energy for ℓ_0-plane pursuit Segmentation Using Reflectance Values

Let $i : V \to \mathbb{R}$ be the function associating for each point of the cloud its reflectance. We denote $i(\pi_v)$ the mean value of points' intensities inside the region to which v belongs. We modified the *data term* of equation 4.2 to minimize the reflectance variation inside a single region:

$$E_i(\Pi) = \underbrace{\sum_{v \in V} (d\,(\,v, \pi_v\,)^{\,2} + \|i(v) - i(\pi_v)\|^2)}_{Data\ term} + \mu \underbrace{\sum_{(u,v)\in E} w_{u,v}\,[\,\pi_u \neq \pi_v\,]}_{Regularizer}. \tag{4.16}$$

Like for the original energy of ℓ_0-plane pursuit algorithm, we want to minimize this new energy. The ℓ_0-plane pursuit algorithm can be adapted in a straightforward manner to account for this change.

4.3.6.3 Experiment

We tested this method on the Barcelona dataset. The RANSAC stopping criterion has been fixed to 0.05 and the regularization strength has been set to 5×10^{-3}. The results are shown on Figure 4.12. The standard ℓ_0-plane pursuit algorithm produces a segmentation composed of 44,000 regions and an error of 91,280 m^2. The modified version of the algorithm, using the reflectance, produces a segmentation composed of 42,000 regions and a geometric error of 105,609 m^2.

The reflectance provides a complementary information to the geometric criteria that we use for segmenting urban scenes. The computed segmentation is thus homogeneous in geometry and in reflectance. The evaluation metric we propose is purely geometric, thus, a method optimising both the geometry and the homogeneity of the reflectance should perform less than a segmentation method based only on geometric criteria. Our experiment shows that combining geometric and reflectance information in ℓ_0-plane pursuit allow the creation of segments homogeneous in reflectance without decreasing the geometric quality of the segmentation.

4.4 Conclusion

In this chapter, we introduced a new method for the piecewise-planar approximation of 3D data based on an adjacency structure. This method is adaptive to the local geometric complexity of the cloud and is suitable for large datasets (i.e. millions of points). Our algorithm only requires a point cloud as input but can benefit from an existing adjacency information such as the one provided by a triangulated mesh.

We acknowledge that our method could be improved by implementing an adaptation of the *merge-resplit* strategy presented in Landrieu and Obozinski, 2017. Indeed, the merge step doesn't seem able to remove all the small artifacts occasioned by early segmentation steps.

However, the main drawback of our method is its tendency to lose the topological connection between adjacent regions. Instead, we obtain a set of planar regions that are not topologically connected. A reconstruction method such as the one presented by Ochmann et al., 2016 could overcome this drawback.

An interesting contribution to improve this method would be the use of multi-primitives suggested by Vidal, Wolf, and Dupont (2014) and Li and Feng (2019) in our RANSAC (such as spheres or cylinders) that are better approximations of some of the objects found in urban areas, such as trunks or poles. Another interesting perspective would be to use the proposed segmentation as part of a polyhedral reconstruction algorithm.

Last, we tested our algorithm with more than geometric features. This experiment demonstrated the ability of our method to combine geometric and reflectance information to produce segments which are both planar and homogeneous in reflectance. We think that the reflectance is useful in our context as it can help detecting texture variations in piecewise-planar objects, such as bricks in a stone wall or solar panels in a roof. Further experiments could focus on testing radiometric features (e.g. color information acquired by a camera coupled to the LiDAR).

5 Polygonalization

Contents

In this chapter, we investigate the possible approaches for the simplification of 3D meshes in order to produce light yet geometrically accurate 3D models of urban scenes. We consider that we have built a simplicial complexe as explained in Chapter 3 and extracted planar regions from its 2D part (the surface mesh) as detailed in Chapter 4. We observe that only a few regions contain most of the triangles of the input surface mesh (e.g. the road may contain more than 50% of the triangles). This means that many triangles are close to being coplanar. Hence, we argue that it is possible to merge some of these nearly coplanar triangles into planar 3D polygons in order to produce a lighter reconstruction while preserving the geometric accuracy of the input mesh.

The simplification process is done in three steps. First, we project points on the least squares supporting planes computed during the segmentation process. Then we remove redundant data. Last, we measure the influence of the ℓ_0-plane pursuit algorithm for point cloud simplification.

5.1 Simplification of Simplicial Complexes

The simplicial complexes, as built in Chapter 3, contain points, edges and triangles. The problem of the simplification of simplicial complexes can be decomposed as three distinct problems: the simplification of points, of eges and of triangles. In this chapter, we only investigate the simplification of triangles, which is more complex than the first two problems. We choose to dedicate the remaining of this section for introducing the simplification of points and edges problems, but no experiment will be done on these problems.

5.1.1 Simplification of Point Clouds

The simplification of 3D point clouds, also called *point cloud decimation*, is mostly based on two different approaches: voxel-based methods and random sampling.

5.1.1.1 Voxel-based approaches

These approaches divide the input point cloud in a subset of regions of similar size. For each region, all the points are discarded except one. This allows to fix a maximal point density in the cloud, and remove more points in dense areas. The subdivision of the cloud in voxels can be based on an octree (Shekhar et al., 1996) or a kd-tree (Xiao and Huang, 2009). Voxels can also be obtained with a previous segmentation step, for instance using the K-Means algorithm (Shi, Liang, and Liu, 2011). In order to select the remaining point for each voxel, researchers usually focus on geometric priors, such as normals or curvature (Han et al., 2015). For an extensive comparison of point cloud decimation algorithms, we refer the reader to the work of Pauly, Gross, and Kobbelt (2002).

5.1.1.2 Random sampling

These methods are based on the random selection of points in the cloud (Winkelbach et al., 2004). These methods allow to fix the final number of points in the cloud, but cannot ensure that the resulting point cloud has a similar density everywhere.

The random-based methods are easy to implement, but usually cannot fit geometric or density-based priors. On the ohter side, voxel-based methods are able to respect geometric or density-based constraints but require to carefully choose the voxel generation method in order to adapt to the geometry of the cloud.

5.1.2 Simplification of Edges

The simplification of edges process is divided between top-down approaches, such as Edge-Collapse (Garland and Heckbert, 1997), and bottom-up approaches, as in Visvalingam and Whyatt (1990).

5.1.2.1 Top-Down Approaches

Simplification of edges can be done by considering the whole point clouds with its edges and collapsing edges (Hoppe et al., 1993; Garland and Heckbert, 1997). This is usually done by computing a *removing cost* for all edges of the graph, based on geometric fidelity and volume preservation (Lindstrom and Turk, 1998). The same approach has been used for simplifying 3D meshes as well. The edge-collapse algorithm is detailed in Section 5.3.3.

Top-down approaches are usually hard to implement but are fast and able to preserve the geometry of the input set of edges.

5.1.2.2 Bottom-Up approaches

Bottom-up approaches are based on the fusion of adjacent edges. The most famous algorithm for line simplification is the *Douglas-Peucker* algorithm (Douglas and Peucker, 1973; Visvalingam and Whyatt, 1990). This algorithm is based on Lang (1969), and simplifies a polyline in a single line if the maximum distance of a node of the polyline to the simplified line is lower than a given threshold. Line simplification can also be performed by iteratively removing triplets of connected points with the lowest areas (Visvalingam and Whyatt, 1993).

Theses approaches perform well in cartography, for instance for road network generalization. However, they do not necessarily respect the non-self-intersections constraints (Wu and Marquez, 2003).

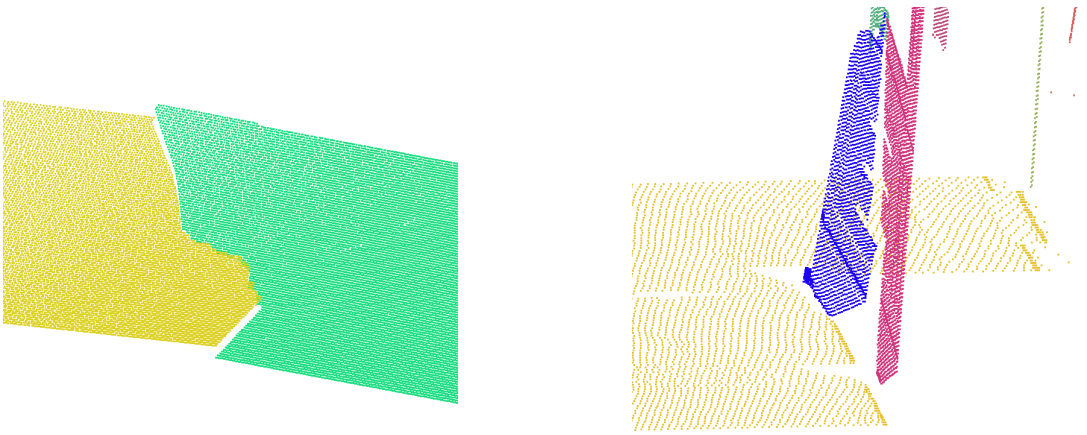
Top-down approaches allow for fast and geometrically accurate simplifications but are usually harder to implement than bottom-up approaches. Also, top-down methods, as they process the whole graph, are able to simplify geometrically simple areas while preserving the geometry of complex areas.

The remaining of this chapter focuses on mesh simplification. Our approach takes as input a mesh, segmented as in Chapter 4. We start by converting our segmented mesh in a piecewise-planar mesh.

5.2 Piecewise-planar Projection of 3D Point Clouds

In this section, we focus on the segmented meshes obtained in Chapter 4. From the segmentation, we want to obtain a piecewise-planar point cloud without losing most of the geometric details. One of the main drawbacks of the ℓ_0-plane pursuit algorithm is that we loose the topological connections between adjacent regions. In fact, if we

project the points directly on the supporting plane of their associated region, we observe that gaps appear in the reconstructed mesh. These gaps appear at the interface between adjacent regions. Some examples are shown on Figure 5.1.



(a) Illustration of a gap appearing in a road due to the segmentation.

(b) Illustration of the gap appearing in a facade due to the segmentation.

Figure 5.1 – Illustration of different gaps appearing between adjacent regions. Each color represents a different region. Points are projected on their respective supporting plane.

In order to overcome the topological errors (gaps between adjacent regions) to appear in the final reconstruction, we decide to project each single point based on their neighborhood. This neighborhood is the same as used for the segmentation method proposed in Chapter 4. If all the neighbors of a point do not belong to the same region, we decide to project the point on the intersection of all the supporting planes of regions present in the neighborhood. This means that:

i) if a point's direct neighborhood belongs to a single region: the point is projected on its associated plane,

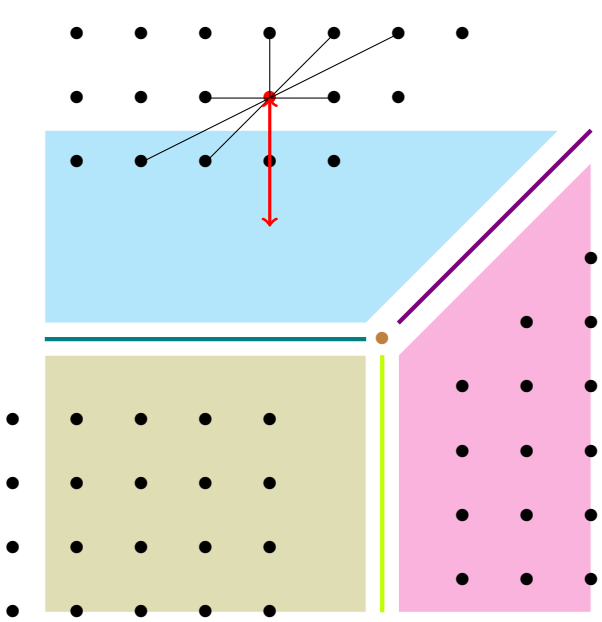
ii) if a point's direct neighborhood belongs to two regions: the point is projected to the intersection of the two planes. This intersection may not be defined (this is the case if the supporting planes are parallel). If this intersection does not exists, the point is projected on the closest plane according to the squared euclidian distance.

iii) if a point's direct neighborhood belongs to three regions: the point is projected to the intersection of the three planes. The intersection of three planes is not always defined [28]. If this intersection does not exist, we consider each pair of planes and project the point on the closest intersection line. If no intersection line exists, we project the point on its closest plane.

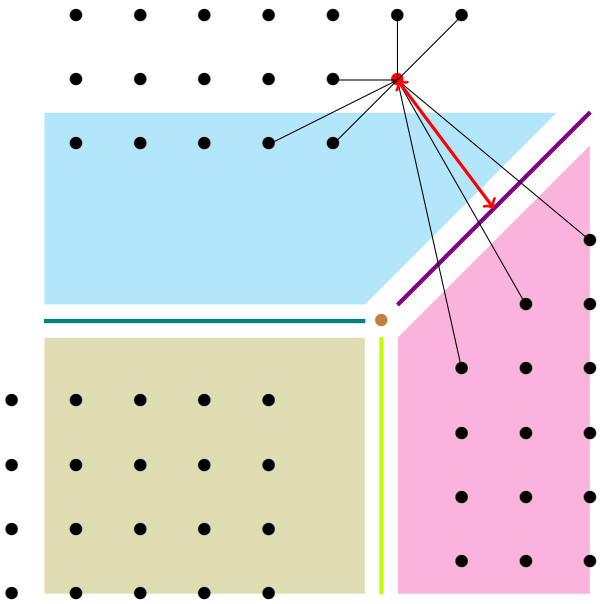
iv) if a point's direct neighborhood belongs to four or more regions: we are in a case where all the planes may not have a defined intersection. In this case, we decide to consider all triplets of adjacent regions and find the best possible projection as defined in previous entry.

We stated at the beginning of this section that we want to keep as much the geometric details of the scene as possible. However, the segmentation algorithm defined

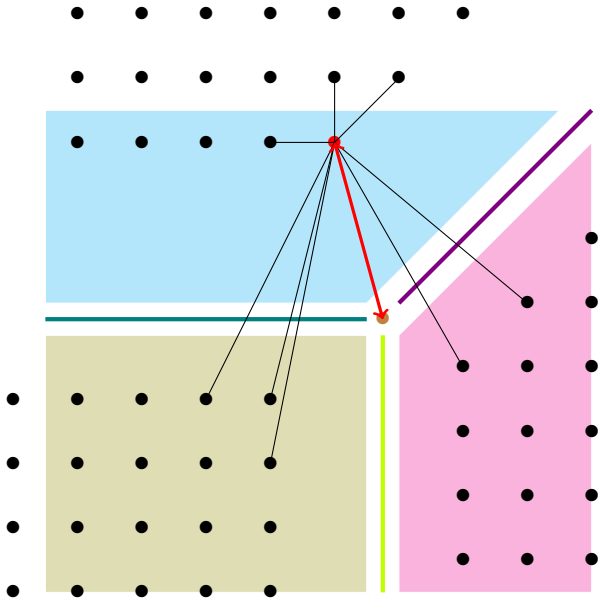
28. See `http://geomalgorithms.com/a05-_intersect-1.html` for an illustration of the different possible configurations.



(a) When a point and its neighbors belong to a single region, the point is projected on this plane.



(b) When the neighborhood of a point belongs to two different regions, the point is projected on the intersection line of the two planes.



(c) When the neighborhood of a point belongs to 3 different regions, the point is projected to the intersection point of the 3 planes.

Figure 5.2 – Illustration of the projection applied for each point according to its neighborhood and the segmentation produced in Chapter 4. Here we represent a set of points segmented in 3 regions. The supporting planes and their intersections with other planes are displayed. The black edges represent the adjacency relationship around the considered point (in red). The red arrow links the considered point to the primitive (plane, line, point) on which it is projected.

in Chapter 4 can associate a point to a region, even if the distance between this point and the supporting plane of the region is high. This is due to the formulation of the energy minimized by this algorithm (cf. Equation 4.2). So, we modify the previous cri-

teria to project a point if and only if the distance between the point and its projection is lower than a given threshold.

For all the experiments done in this chapter, we fix the threshold between a point and its projection to 1 m. We chose this value to show that we think that our segmentation algorithm preserved most of the geometry of the input mesh, and that we only want points not to be projected really far from the input mesh (which can happen when a point is at the edge between two nearly coplanar regions).

5.3 Simplification of 3D Meshes

5.3.1 Definition

We call *simplification of 3D point clouds* the process of decreasing the quantity of information used to represent the cloud (e.g. decreasing the number of points) while preserving its geometrical shape. The goal of the *simplification of 3D meshes* is to obtain a point cloud that is easier to process than the input cloud while being geometrically similar to the input cloud.

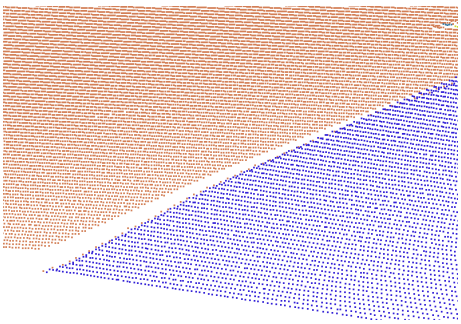
The simplification of 3D meshes is still a complex challenge in the remote sensing community (Heo et al., 2013; Zhu et al., 2018). Point cloud simplification often relies on two different principles: contour extraction (usually performed with the α-shape algorithm (Akkiraju et al., 1995)) and merging points, edges or triangles (Garland and Heckbert, 1997). Well known algorithms for point cloud simplification include Polyfit (Nan and Wonka, 2017) or Variational Shape Approximation (VSA) (Cohen-Steiner, Alliez, and Desbrun, 2004). Li, Wonka, and Nan (2016) argue that the simplification process can be driven by Manhattan-world assumptions even if orthogonlism and parallelism should be favored. We think that urban scenes (especially in historical European cities) are more complex and their modeling can not be limited to Manhattan-world assumptions. The simplification itself can be driven by the necessity of producing digital models of the same area at different scales. For instance, Zhu et al. (2017) builds a detailed reconstruction of an urban scene, and propose to use Markov Random Field (MRF) to extract facade and roof contours and generate simplified models with different LoDs (see Section 1.2.2 for a definition).

In this section, we focus on removing all the redundant data. Ideally, to represent a planar region we only need the contour points of the region. This means that we want to remove all the points and edges inside a single region that do not bring any useful information to the input point cloud.

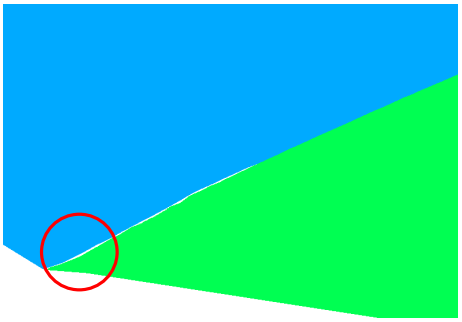
5.3.2 Contour Extraction

A first approach for removing all the data inside a single region is to compute the contour of each region. When processing a mesh, these contours can be easily identified. However, they can be very detailed and noisy, and one may want to obtain simplified contours. In order to achieve this, we use a contour extraction algorithm: we extracted the precise shape of each region by computing their α-shapes [29] (Edelsbrunner, Kirkpatrick, and Seidel, 1983; Akkiraju et al., 1995). The idea here is that the most intuitive way to simplify a piecewise-planar point cloud is to keep only the shape of each planar region. Also, as each planar region can be not convex, we should consider α-shapes and not convex hulls.

29. A clear presentation of α-shapes can be found at: `https://en.wikipedia.org/wiki/Alpha_shape`



(a) Projection of the points of two adjacent regions of the road on their supporting plane. We observe that there is a gap between the two regions.



(b) Visualisation of the α-shape for each region. We can observe that the gap is still present between the two adjacent regions.

Figure 5.3 – Illustration of an α-shape-based simplification. Each color represents a single region. The scene is a zoom on a road portion at the interface between two adjacent regions.

5.3.2.1 Definition

Let $V \in \mathbb{R}^{n\times 3}$ be a set of points and $\alpha \in \mathbb{R}$ a number. We first introduce the notion of *generalized disk* on which α-shapes rely.

Generalized disk: Let $\frac{1}{\alpha}$ be the radius of the generalized disk. Depending on α, the generalized disk is either:

— a disk of radius $\frac{1}{\alpha}$ if $\alpha > 0$,

— a halfplane if $\alpha = 0$,

— the complement of a disk of radius $\frac{1}{\alpha}$ if $\alpha < 0$.

α-shape: Let $\{v_i, v_j\} \in V^2$ be two distinct points of V. An edge is constructed between v_i and v_j if and only if there exists a *generalized disk* of radius $\frac{1}{\alpha}$ containing V and which has the property that v_i and v_j lie on its boundary. We call *α-shape of V* the polygon defined by the set of such constructed edges.

An α-shape is neither necessarily convex nor necessarily connected. For large values of α, the α-shape is identical to the *convex hull* of V.

5.3.2.2 Experiment

We computed the α-shape for each region of our piecewise-planar approximations of meshes. We used the implementation provided by the CGAL library (Da, 2019). This approach allows us to keep the global shape of each region, while discarding unnecessary points and edges inside each region. However, this approach do not overcome the topological problems presented in Section 5.2 as seen on Figure 5.3. In fact the interface between adjacent regions is not necessarily reconstructed, which leads to holes in the final mesh. Moreover, the discontinuities shown on figure 5.1 still appear.

5.3.3 Mesh Decimation

The core idea of mesh decimation is to produce a simpler mesh than the input mesh (with less vertices, edges and faces) that is a geometrically accurate approximation of the input mesh.

In the litterature, this is usually done by means of vertex removal and egde collapse techniques (Garland and Heckbert, 1997). We now describe these techniques, and then present various works of the litterature based on such algorithms.

5.3.3.1 Principles

The most known mesh decimation techniques, such as vertex removal or edge collapse, were proposed by Hoppe et al. (1993), Garland and Heckbert (1997), and Kobbelt, Campagna, and Seidel (1998). Mesh decimation algorithms are based on the following principles:

— **vertex decimation**: consists in selecting a vertex and removing it from the reconstruction. A first approach has been proposed by Schroeder, Zarge, and Lorensen (1992). Their algorithm selects a vertex, removes it and all its adjacent faces. Then the resulting hole is triangulated. A similar approach was proposed by Soucy and Laurendeau (1996). Vertex decimation is illustrated on Figure 5.4.

— **vertex clustering**: it consists in merging a set of close vertices in a single one and to re-triangulate the mesh around this vertex. The *clustered vertex* obtained is not necessarily a vertex of the initial set. For instance, Rossignac and Borrel (1993) divide the scene according to a regular grid, and all the vertices of a single cell are clustered in a single vertex. The grid can be replaced by an octree, or even by a hierarchical set of octrees, where all the vertices inside a single cell are clustered and each hierarchical level leads to more or less simplified reconstruction (Schaefer and Warren, 2003).

— **edge collapse**: edge collapse consists in greedily collapsing edges, starting with the collapses that minimize a certain error (Hoppe, 1996; Cignoni et al., 2000). Because it is the approch that we have chosen, we will detail the algorithm in Section 5.3.3.2.

— **non edge collapse**: following the edge collapse approaches, the algorithm of Garland and Heckbert (1997) proposes not to collapse an edge, but to collapse a pair of vertices. The main difference with classic *edge collapse* techniques is that their algorithm do not consider an existing edge but just a pair of vertices. This allows for reconnecting close parts of the mesh that were unconnected. the *non-edge collapse* technique, after choosing a pair of vertices, proceeds in a similar way as the *edge collapse* technique described in the previous point.

5.3.3.2 Edge Collapse Algorithm and Cost-Strategy

There exists various ways to collapse an existing edge linking a source and a target vertices (v_s, v_t). They are listed hereafter:

1. Collapse the edge on the source vertex v_s (half-edge collapse) (Figure 5.5b). All the edges connected to the target vertex v_t are now connected to the source vertex v_s.
2. Collapse the edge on the target vertex v_t (half-edge collapse) (Figure 5.5c). All the edges connected to the source vertex v_s are now connected to the target vertex v_t.
3. Create a new point between the source and the target vertices of the edge (Figure 5.5d). This new point can be chosen as the barycenter of the former points. All the edges connected to either the source v_s or the target v_t of the former edge are now connected to the newly created vertex.

In our case, we don't process non-manifold meshes, and we already reconnected adjacent regions, hence we will not use the *edge collapse* algorithm of Garland and Heckbert (1997), but the algorithm of Lindstrom and Turk (1998). This algorithm is an appropriate algorithm for simplifying our data while preserving the topology. Let v be a vertex and $e = \{v_0^e, v_1^e\}$ an edge. Let $(e_0, \ldots, e_n) \mid e_i = \{v_0^{e_i}, v_1^{e_i}\}$ be the set of edges affected by the collapse of e. We now present an overview of the algorithm:

1. **Compute cost for collapsing each individual edge of the mesh**. They are then ordered in a priority queue according to their associated cost. The cost is composed of three different terms: a shape, a volume and a boundary preservation terms. Each term is explained hereafter:
   — Volume preservation: collapsing edges usually induces a modification of the global shape of the mesh [30]. In order to preserve the geometry of the input mesh and to prevent over-simplifications, Lindstrom and Turk (1998) added a volume preservation term in the cost of each edge. This volume preservation term is based on the volume change affecting the triangles connected to the removed edge. Let $t = (v_i^e, v_1^t, v_2^t)$ be a triangle, connected with the collapsed edge by vertex v_i^e. Let $t' = (v, v_1^t, v_2^t)$ be the triangle created after collapsing e, leading v_i^e to be shifted to v. The volume difference is described as the volume of the tetahedron $p = (v, v_i^e, v_1^t, v_2^t)$. The authors say that the volume difference is *positive* if v is *above* [31] the supporting plane of t; the counterpart is that the volume difference is *negative* if v is *below* the supporting plane of t. They argue that collapsing an edge should induce a small volume difference in the model. Lindstrom and Turk (1998) show that, if the mesh is locally manifold, the volume difference due to an edge collapse can be computed as the sum of volume differences per triangle:

$$f_V(e, v) = \sum_i V((v, v_0^{t_i}, v_1^{t_i}, v_2^{t_i}))^2 \,. \tag{5.1}$$

   — Boundary preservation: this is the 2D equivalent of the volume preservation. In other words, the boundary preservation aims at minimizing the squared sum of areas impacted by the collapse of a given edge. In the case of a planar mesh, the surface area change is set to be null:

$$\sum_i A((v, v_0^{e_i}, v_1^{e_i}))^2 = 0 \,. \tag{5.2}$$

   However, surface boundaries are usually not planar. Hence, Equation 5.2 has to be adapted. This is done by relaxing the requirement that all impacted edges are coplanar. Instead, each surface change is expressed with a direction, and Equation 5.2 is reformulated so as to minimize the magnitude of the sum of directed area vectors. This magnitude is correlated to the fidelity of the change in each direction. Hence, the authors propose the following reformulation:

$$\sum_i A((v, v_0^{e_i}, v_1^{e_i}))^2 = \frac{1}{4} \left\| v \times e_1 + e_2 \right\|^2 \,, \tag{5.3}$$

   with $e_{1_i} = v_1^{e_i} - v_0^{e_i}$, and $e_{2_i} = v_1^{e_i} \times v_0^{e_i}$. The final goal is to have the smallest possible surface change in the graph while collapsing an edge. Hence,

30. Except in the case where all the adjacent points are coplanar.
31. In this context, above means outside of the model.

Lindstrom and Turk (1998) propose to minimize the following:

$$f_B(e, v) = L(e)^2 \sum_i A((v, v_0^{e_i}, v_1^{e_i}))^2 \,, \tag{5.4}$$

with $L(e)^2$ the squared length of the considered edge. This favors the collapse of short edges.

— Shape optimization: when the set of points that should be re-triangulated is coplanar, there is not a unique way to triangulate it. In this case, the algorithm favors equilateral triangles over elongated ones, as the authors argue that elongated triangles may *"introduce unwanted shading discontinuities and may slow down some rendering methods"*. Hence, they want to minimize the sum of squared lengths of edges between the vertices of the collapsed edge and the newly created vertex v:

$$f_S(e, v) = \sum_i L((v, v_i))^2 \tag{5.5}$$

The overall cost for collapsing an edge is computed as a weighted sum of the costs associated to the constraints: volume and boundary preservation:

$$c(e, v) = c_V f_V(e, v) + c_B f_B(e, v) \,, \tag{5.6}$$

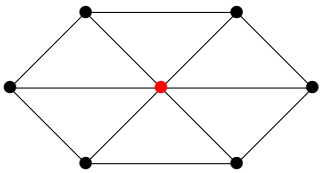
with c_V and c_B some constants used to weight the volume and boundary preservation terms. Note that the shape preservation term is not present here. In fact, this term is added only when f_V and f_B are close to zero, in order to solve placement ambiguities. For more details about the energy associated to each constraint, we refer the reader to Lindstrom and Turk (1998) and Lindstrom and Turk (1999).

Also, when collapsing an edge, the new vertex has to be carefully chosen. If the vertex on which the edge is collapsed is not carefully chosen, large shape variations can appear. For instance, if we consider a mesh shaped like a sphere, and if we collapse all its edges on the middle of each edges, the resulting simplified mesh will be a tetahedron. In the end, the 3D position of vertex v is chosen as:

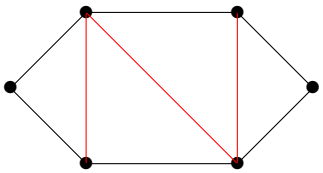
$$v^\star = \arg\min_v \; c(e, v) \,. \tag{5.7}$$

2. **Collapse an edge**. At this point of the algorithm, all edges are sorted in a priority queue, according to the three constraints presented in the previous point. The edge with the lowest overall cost is selected, and if there exists a point (not necessarilly member of the 3D mesh) that allows to collapse the edge while preserving the topology of the mesh, the edge is collapsed. If there exists no such point, then the edge is considered as *non-collapsable* and receives an infinite weight. In this case, the next edge with the lowest cost is considered, and so on until there exists an edge with a finite cost that can be collapsed. If no such edge exists, the algorithm stops.
3. **Recompute the costs of edges that have been affected**. The authors consider that an edge is affected if it belongs to a triangle from which at least one vertex has been modified. Once an edge has been collapsed, a new priority queue is created based on the remaining edges.

4. **Run iteratively until**:
   — No edge with a small enough cost can be collapsed: this means that further collapses will highly increase the geometric error,
   — The sum of all the collapses is equal to or higher than a given threshold, as advocated by Kobbelt et al. (1998),
   — The remaining number of edges or triangles is lower than or equal to a given treshold. We are choosing this last stopping criterion, as we think that a fair comparison between two simplified 3D models should require that the models are composed of the same number of primitives. Garland and Heckbert (1997) similarly propose to perform a given number of collapses.



(a) A graph with a point to remove (in red)



(b) The red point and its associated edges have been removed, creating the new red edges.

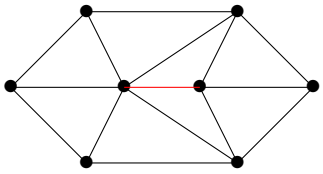
Figure 5.4 – Illustration of a vertex removal.

5.3.3.3 Edge Collapse in the Litterature

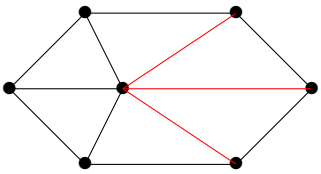
Edge collapse is one of the leading algorithms when it comes to simplifying 3D meshes (Kaick and Pedrini, 2006; Salinas, Lafarge, and Alliez, 2015; Maggiori et al., 2017). In fact, edge collapse have been used for various applications, such as skeleton extraction (Au et al., 2008). Collapsing edges can be performed in parallel when the different collapses do not impact the same edges (Lee and Kyung, 2016). Pan, Zhou, and Shi (2001) propose to merge three vertices at once (i.e. a triangle), leading to multiple edge collapsing in a single operation. Their method has been extended to polygons with more than three sides by Chen, Luo, and Ling (2007). Odaker, Kranzlmueller, and Volkert (2015) use this technique to perform a real time simplification of 3D meshes on the visible part of the model. This can be used for rendering applications. Barmak and Minian (2012) and Boissonnat and Pritam (2019) extended the edge collapse approach to simplicial complexes.

5.3.3.4 Experiment

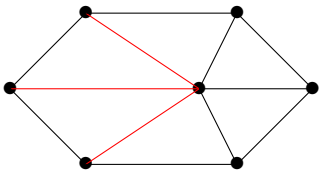
We decided to use the *edge collapase* algorithm of Lindstrom and Turk (1998) to solve our problem. We used the implementation provided by the CGAL library (Cacciola, 2019). Results are visible on Figure 5.6. As expected, edge collapse approaches allow to reduce the number of triangles by several orders of magnitude. Also, the approximation *seems to* preserve the complex geometry of the data. However, we need to find a suitable way of assessing the real quality of the approximation.



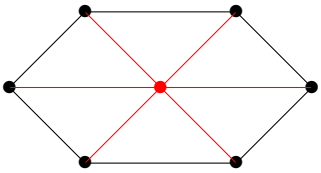
(a) A graph with an edge to remove (in red)



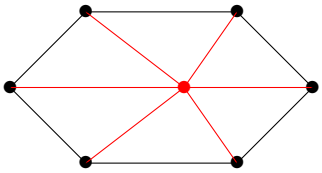
(b) The edge has been collapsed on the the vertex of the left.



(c) The edge has been collapsed on the vertex of the right.

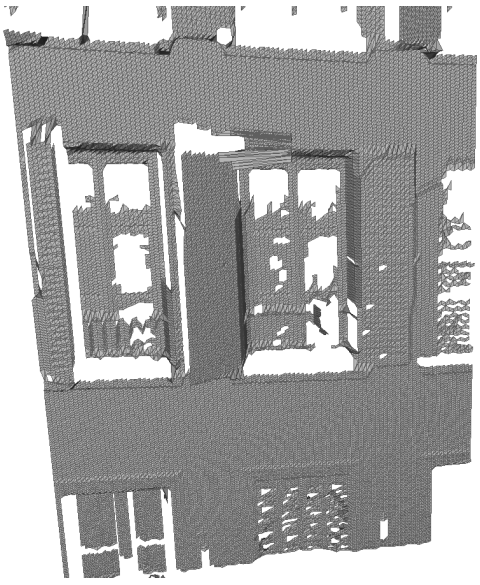


(d) The edge has been collapsed and new point was created at the middle between the source and the target of the former edge.



(e) The edge has been collapsed and new point was created at an optimal place between the source and the target of the former edge.

Figure 5.5 – Illustration of an edge collapse.



(a) Input mesh for edge collapse

(b) Output mesh produced by the edge collapse algorithm

Figure 5.6 – Illustration of the edge collapse algorithm on a geometrically complex facade.

5.4 Evaluation

We want to measure the quality of the of the mesh generalization process. The more we generalize a mesh, the geometrically worse the approximation is. Thus, we decide to evaluate the quality of approximation by computing the *degree of generalization* —that can be measured by the complexity or description length of the geometric structure— and the geometric error introduced by the simplification process. Here, this *degree of generalization* can be interpreted as the number of points and triangles used to represent the final mesh.

One of the main principles in *information theory* is based on the *law of parsimony*[32] and states that "Entities should not be multiplied without necessity" (Jeffreys, 1939; Jefferys and Berger, 1991). In computer science, the law of parsimony can be verified by making sure that only the required information is stored, using as little bytes as possible. The number of bytes used to store the information stored is called the Description Length (DL). This description length metric is discussed first, and Section 5.4.2 present the main geometric evaluation metrics.

5.4.1 Minimum Description Length

5.4.1.1 Definition

We call DL the number of bytes used to encode an information. For instance, 1 B is neccessary to encode a `boolean`, which means its DL is 1.

A description length is not a metric in itself as the same information can be encoded with different techniques resulting in different DLs. For instance, a non degrading compression algorithm can reduce the DL without altering the information. As a result, the appropriate metric to measure a quantity of information is the Minimal Description Length (MDL) which was introduced by Rissanen (1978).

The MDL principle has been used with LiDAR data mostly for two applications: digital model generation and point cloud segmentation. The digital model generation (which comprises DTM and DSM) relies on the MDL for ensuring that the extracted ground has a simple shape (Zhou et al., 2004; Sohn and Dowman, 2007). Along the same line, MDL can also be used to ensure that each segment of a point cloud segmentation has a simple shape (Matei et al., 2008; Jung, Jwa, and Sohn, 2017).

5.4.1.2 Minimum Description Length for a Triangular Polyhedron

In this chapter, the information that we are interested in is a continuous surface representation such as a mesh or a polyhedron. It is composed of 3D points and triangles. Each single point has 3 coordinates. We store them using *float* numbers. Each triangle contains the index of its 3 vertices. We store these indices using a number of bytes equal to the binary logarithm of the number of vertices.

Let V be the set of points and T be the final set of triangles. We compute the MDL for a mesh as the following:

$$\text{MDL}_{\text{meshes}} = 3 \cdot \textit{sizeof(float)} \cdot |V| + 3 \cdot \log_2(|V|) \cdot |T| \,, \tag{5.8}$$

where *sizeof(float)* represents the number of bytes allocated to a float number. All the code of this section was written in C++. In this programming language, the size of a

32. `http://math.ucr.edu/home/baez/physics/General/occam.html` (accessed on 24th February 2020)

float number is: 4 bytes [33].

5.4.1.3 Description Length for a Polyhedron

Ultimately, a mesh may not be the best structure for storing simplified meshes. In the case where we have two or more adjacent triangles that are coplanar, storing a polygon (e.g. the convex hull of the set of coplanar triangles) allows to save a few bytes without losing any information on the data (Miller and Stout, 1988; Olariu, Schwing, and Zhang, 1993).

Let P be the set of polygons composing the final reconstruction of an urban scene. The MDL for a set of polygons is defined as:

$$\text{MDL}_{\text{polygons}} = \textit{sizeof(float)} \times (3 \cdot |V| - \left(\sum_{p \in P} |p| - 3 \right) \ . \tag{5.9}$$

This metric takes advantage of the fact that polygons are planar, and that if a single polygon contains more than three points, the remaining points can be stored only by using only 2 coordinates that locates them in the polygon's supporting plane.

In our case, we chose to restrain to evaluating meshes and not polygons. This is due to the fact that we compare our simplified reconstructions to methods producing meshes, and it would not be fair to compare quantity of information required to store a set of polygons to the one of a set of triangles.

5.4.2 Geometric Error Metrics

In this section, we present the three main types of geometric evaluation metrics: Hausdorff-distance based metrics, sum-of-distances based metrics and quadric based metrics.

5.4.2.1 Hausdorff Distance Based Metrics

Let X and Y be two sets of points. Let $d(x, y)$ be the euclidean distance between a point $x \in X$ and a point $y \in Y$. The euclidean distance between x and Y is then defined as:

$$d(x, Y) = \inf_{y \in Y} d(x, y) \ . \tag{5.10}$$

From this, the Hausdorff distance between the two sets X and Y is defined as the maximum euclidian distance between a point $x \in X$ and Y:

$$d_H(X, Y) = \sup_{x \in X} d(x, Y) \ . \tag{5.11}$$

This distance is non-symetric, which means that:

$$d_H(X, Y) \neq d_H(Y, X) \ . \tag{5.12}$$

However, a symetric distance can be defined from the Hausdorff distance, as explained by Klein, Liebich, and Straßer (1996):

$$d_{H_{\text{sym}}} = max(d_H(X, Y), d_H(Y, X)) \ . \tag{5.13}$$

33. `https://docs.microsoft.com/en-us/cpp/cpp/fundamental-types-cpp?view=vs-2019` (accessed on 12th December 2019)

They argue that this distance is more intuitive for comparing meshes. They propose a mesh-simplification algorithm in which a vertex is removed if and only if the symetric Hausdorff distance between the original mesh and the simplified one is smaller than a given threshold. The METRO software (Cignoni, Rocchini, and Scopigno, 1998), a software for comparing meshes, also uses a variation of the Hausdorff distance for computing signed distances between surfaces.

Borouchaki and Frey (2005) proposed to define a Hausdorff-based envelope for simplifying meshes. They define a global envelope around the surface and a cone tolerance for each node. They simplify an input mesh if and only if the resulting mesh preserves the geometry of the initial mesh (this is verified using the global envelope), and if the local shape of the new mesh is close to the previous local shape (this point is enforced by the tolerance cone around each point).

The Hausdorff distance is usually hard to compute, even if some techniques exists to fasten its computation. For instance, Guthe, Borodin, and Klein (2005) propose an algorithm able to quickly determine the regions of highest geometric distance between two meshes, and use it compute the Hausdorff distance only between such areas.

5.4.2.2 Sum of Distances Based Metrics

As Hausdorff distances usually remains hard to compute, some works proposed to directly compute the euclidian distances between the approximated model and the original one (Hoppe et al., 1993). These metrics are usually computed as the squared sum of distances between every point of the approximated model to the initial model:

$$d(X,Y) = \sum_{x \in X} d(x,Y)^2 \tag{5.14}$$

Cohen-Steiner, Alliez, and Desbrun (2004) proposed to extend this metric to take into account normals. They argue that unlike Hausdorff-based metrics, which are meant to compares triangulated surfaces, sum of distances-based metrics could be used to evaluate the quality of geometric primitives [34] used to simplify a mesh. In this contex, the comparison of normals between geometric primitives and points' neighborhood ones can ensure that the shape of the input surface is preserved. However, neither Hausdorff-based nor sum of distances-based metrics take into account the local shape of the surface.

5.4.2.3 Quadric Based Metrics

Garland and Heckbert (1997) proposed a new type of error metric: quadric-based metrics. They argue that such metric is both fast to compute and provide a geometrically accurate evaluation. Their approach relies on the squared distance of each point to the plane supporting a considered face. Let $\mathcal{Q}$ be a quadric, f a face of the mesh and $v \in \mathbb{R}^3$ a vertex. We note $\mathcal{Q}^f$ the quadric associated to the face f. This quadric is built from the supporting plane of the facet. Then, the per-face quadrics are used to define a per-point quadric, encoding the local shape of the surface. This quadric is noted $\mathcal{Q}^v$ for a given point v. $\mathcal{Q}^v$ is defined as the sum of the quadrics of the faces containing v. Garland and Heckbert (1997) then define a per-vertex error, where the error of a vertex on the input mesh is equal to the sum of the quadrics of its adjacent faces, weighted by their respective area:

$$\mathcal{Q}^v(v) = \sum_{f \ni v} A(f)\mathcal{Q}^f(v)\ . \tag{5.15}$$

34. These geometric primitives area called 'proxies' in the paper.

When collapsing an edge (v_0, v_1), the new vertex v is found by minimizing the following:

$$\mathcal{Q}^v(v) = \mathcal{Q}^{v_0}(v) + \mathcal{Q}^{v_1}(v) \tag{5.16}$$

The quadric metric defined by Garland and Heckbert (1997) is purely geometric. However, Hoppe (1999) propose to extend this metric in order to take into account other attributes such as color, normals or texture. They define the error associated to a face as the sum of its vertices geometric error (as in Equation 5.15) and a sum of attributes error, which are computed as the standard deviation between per-points attributes and the interpolation of points created from the edge collapses attributes.

Using quadrics, encoding the local shape of the surface, favors the decimation process in geometrically simple areas, and help the preservation of the geometric quality of the input surface. Also, it is possible to take into account other attributes, such as normals or texture information in quadric metrics.

There exists few studies comparing evaluation metrics for mesh decimation. Two interesting works on the subjects are the works of Cignoni, Montani, and Scopigno (1998) and Maglo et al. (2015). The mesh visual quality is often not taken into account in the previously mentionned metrics. Recently, some works proposed to evaluate a simplified model both in terms of geometric fidelity and visual quality in order to correlate with human perception (Maglo et al., 2015, Section 7). However, the approach of this thesis is purely geometric, so we do not investigate such metrics.

The geometric simplifications created using the ℓ_0-plane pursuit algorithm (see Chapter 4) were evaluated with a sum-of-distances based metric. Hence, we decide to use the same geometric metric for evaulating the simplification work done in this chapter.

5.5 Experiments

5.5.1 Compared Methods

We used the segmentations produced in the previous chapter and projected points according to the criteria defined in Section 5.2. Then, we performed an edge collapse in order to simplify the meshes. We compare our results with an *edge collapse* simplification based on the raw point cloud, the VSA (Cohen-Steiner, Alliez, and Desbrun, 2004) algorithm [35] and the *Polyfit* algorithm (Nan and Wonka, 2017). Note that the *Polyfit* algorithm does not necessarily produce a mesh. In fact, *Polyfit* produces a set of planar 3D polygons. In order to compute the MDL for Polyfit results, we consider each independent polygon and compute its Constrained Delaunay Triangulation (CDT) (Chew, 1989). We decided to use a CDT in order to preserve the shape of the original polygon. We used the implementation proposed by Yvinec (2020). We also did some experiments with *Poisson reconstruction* (Kazhdan and Hoppe, 2013) which can then be simplified using an edge-collapse algorithm. However, the first results were not satisfying and it seems that even after a simplification process, the algorithm will not be able to compete with the other tested methods, so we stopped these experiments. However, we still decided to report the results of the raw *Poisson reconstruction* in Figure 5.7 for comparison purposes.

35. For this experiment, we used the implementation of the CGAL library (Alliez, Cohen-Steiner, and Zhu, 2019)

5.5.2 Results

Detailed results for all methods are available on Figure 5.7. Visual results for *edge collapse* simplification of our segmentations are displayed on Figure 5.8. The results for VSA are shown on Figure 5.9. Polyfit and Poisson results are displayed on Figure 5.10 and Figure 5.11 respectively.

We observe that the results of the *edge collapse* algorithm is bounded by the initial quality of the segmentation. It should be noted that, when using our segmentations as input for the simplification, the error is almost not decreasing for a MDL higher than 30 kB. This number corresponds approximately to the minimum information needed to represent our urban scene.

We noticed that the VSA algorithm was not performing as well as our method, even if VSA used the raw mesh as input. This is due to the fact that VSA is a bottom-up method, that is not able to handle large geometrically consistent areas in a mesh while preserving small geometric details.

In addition it is interesting to note that the simplification is nearly always accurate on the road, as observed on Figure 5.8. However, complex areas of the scene such as tops of buildings are easily over-simplified by the *edge collapse* algorithm.

Last, it seems obvious that using an already simplified data (see Section 5.2) as input for the *edge collapse* algorithm will lead to worse results than just feeding the *edge collapse* with the raw mesh. However, we see that for over-segmented meshes the results obtained when hugely simplifying the mesh are as good as the results obtained with the raw mesh. This proves that the segmentations computed in Chapter 4 are able to preserve nearly all the geometric information of the scene.

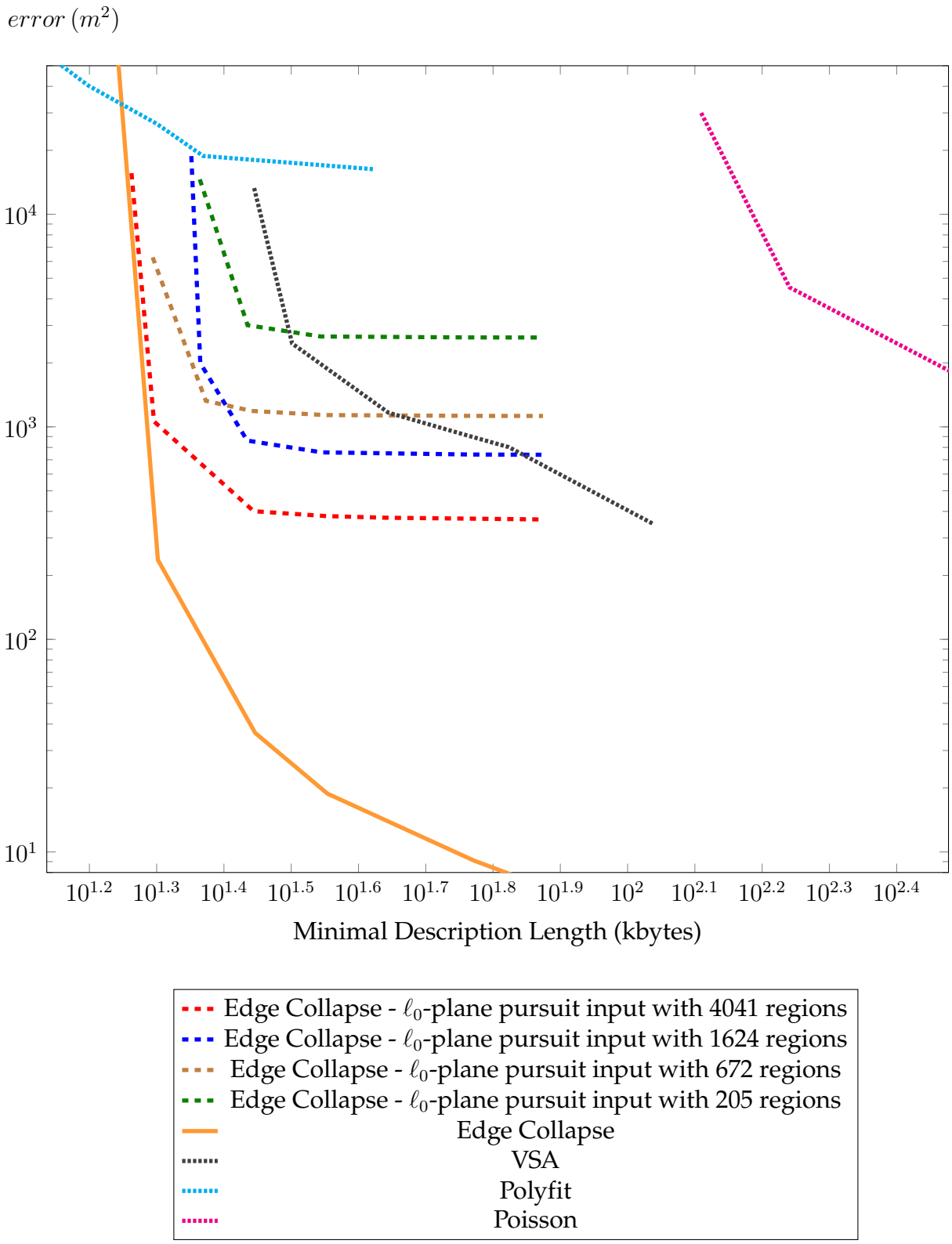
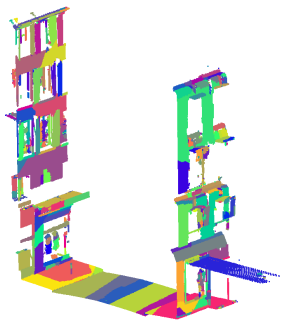
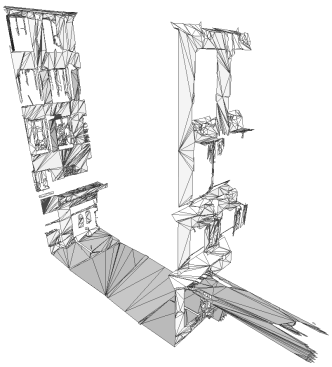


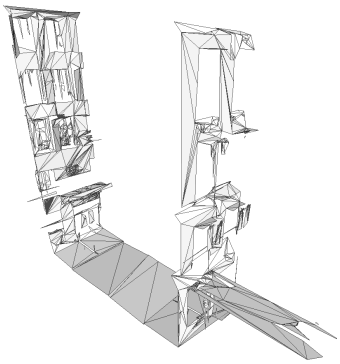
Figure 5.7 – Results of the edge collapse algorithm. For experiments based on ℓ_0-plane pursuit segmentations, we added the number of regions of the final segmentation in the legend. The dashed lines are for experiments based on our segmentations. The plain line represents the *Edge Collapse* algorithm with the raw point cloud as input, this shows the theoretical best results that can be obtained by using the *edge collapse* algorithm. The densely dotted lines stand for the state-of-the-art methods against which we compare our results. This figure is best viewed in color.



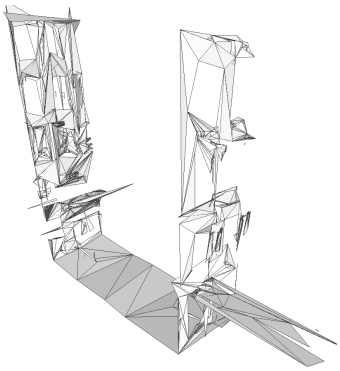
(a) Segmentation, 205 rg



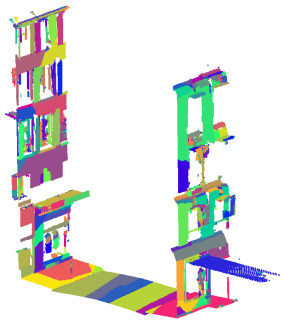
(b) 10,000 eg, ϵ =2631.92 m^2
MDL_{meshes} =50,880.1 B



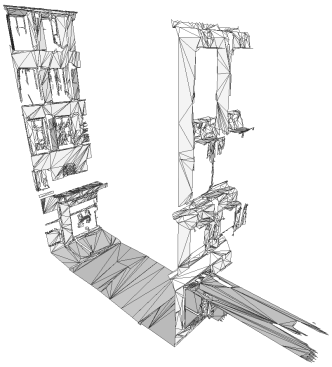
(c) 5000 eg, ϵ =2661.24 m^2
MDL_{meshes} =24,441 B



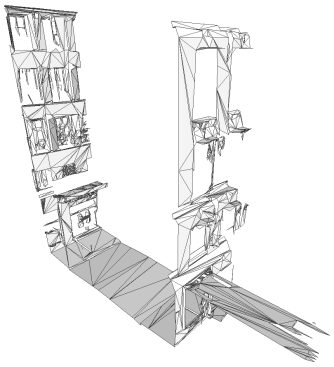
(d) 4000 eg, ϵ =3007.83 m^2
MDL_{meshes} =19,075.9 B



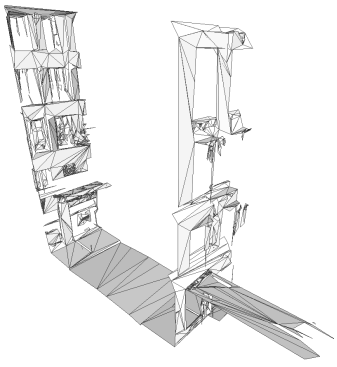
(e) Segmentation, 672 rg



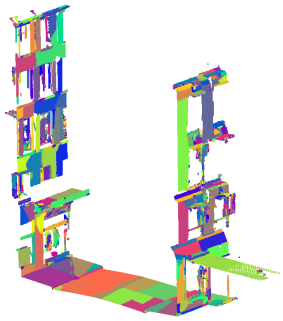
(f) 10,000 eg, ϵ =1125.18 m^2
MDL_{meshes} =51,888.2 B



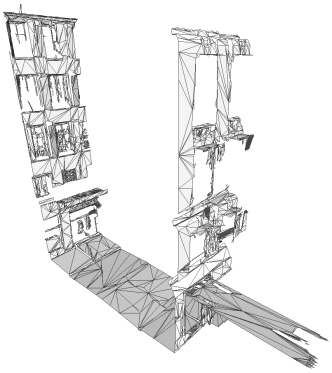
(g) 5000 eg, ϵ =1136.08 m^2
MDL_{meshes} =25,149.1 B



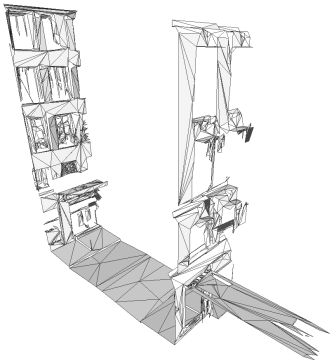
(h) 4000 eg, ϵ =1188.18 m^2
MDL_{meshes} =19,616 B



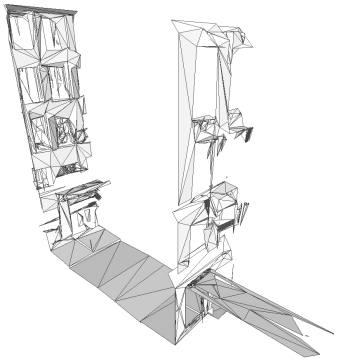
(i) Segmentation, 1624 rg



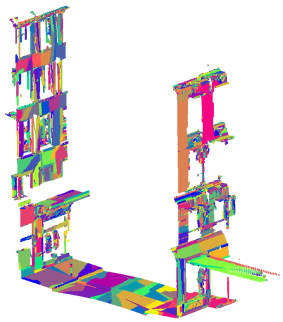
(j) 10,000 eg, ϵ =739.09 m^2
MDL_{meshes} =51,288.2 B



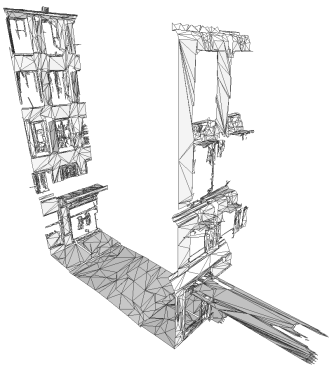
(k) 5000 eg, ϵ =758.15 m^2
MDL_{meshes} =24,633 B



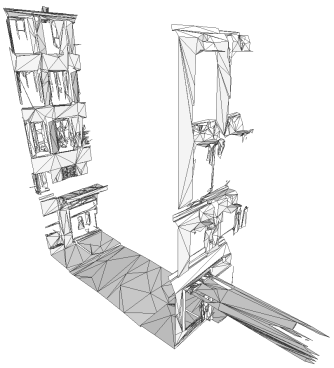
(l) 4000 eg, ϵ =862.11 m^2
MDL_{meshes} =19,015.9 B



(m) Segmentation, 4041 rg



(n) 10,000 eg, ϵ =366.67 m^2
MDL_{meshes} =51,444.2 B

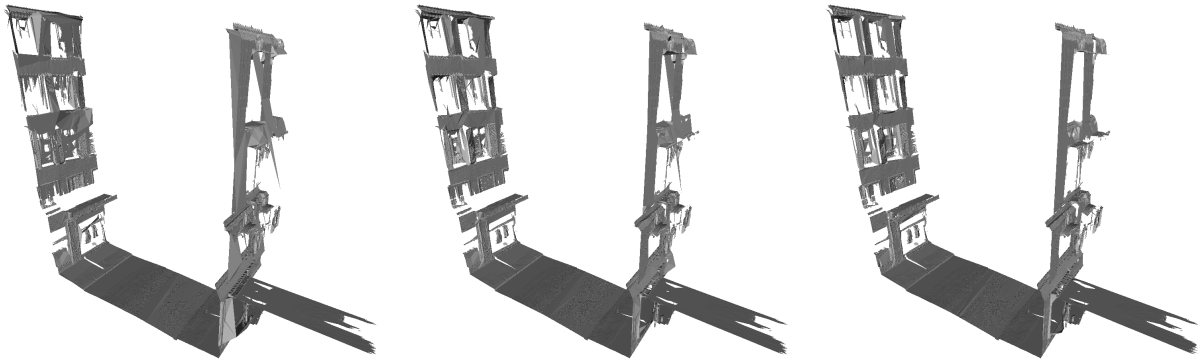


(o) 5000 eg, ϵ =380.25 m^2
MDL_{meshes} =25,413.1 B

(p) 4000 eg, ϵ =400.83 m^2
MDL_{meshes} =19,892.1 B

Figure 5.8 – Illustration of the simplification results. Each row shows results for a different input segmentation. Each color shows results for a different number of edges in the final mesh. For the first column, each color represent a different region. rg means region, eg means edges and ϵ means error.

The VSA algorithm was able to perform an accurate simplification on large and planar areas such as the road. However, VSA has a tendency to over-simplify the facades, leading to sets of points incorrectly agglomerated in a single region. This happens for example between walls and shutters. Results of this algorithm can be seen on Figure 5.9.



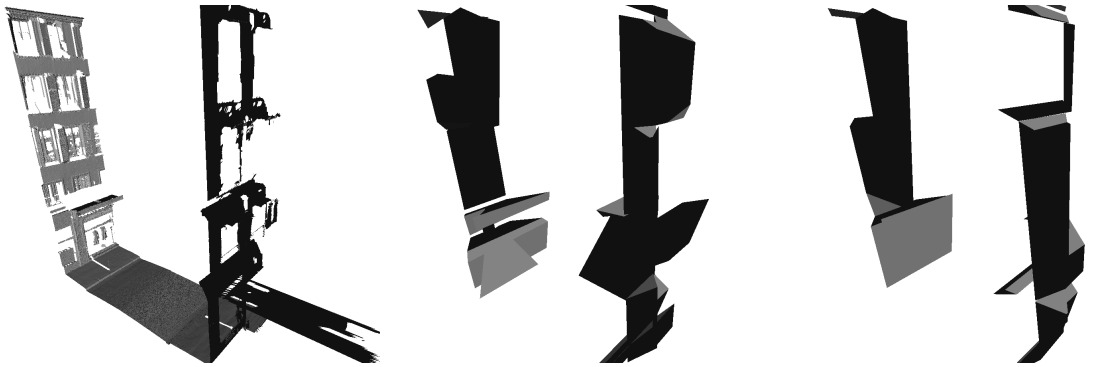
(a) VSA simplification with 200 planar primitives.

(b) VSA simplification with 800 planar primitives.

(c) VSA simplification with 2000 planar primitives.

Figure 5.9 – Examples of reconstructions performed by *VSA*. The simplification performs well on large planar areas such as the road, but is unable to cope with complex areas. This can be seen on the top of facades.

The Polyfit algorithm was unable to perform a geometrically accurate reconstruction as shown on Figure 5.10. This is due to the fact that *Polyfit* tries to reconstruct closed objects, such as houses, which is hard to perform using MLS acquisitions as they usually are not able to acquire the full geometry of the scene. In fact, we do not have enough information here to reconstruct the whole building and it is not the scope of this thesis. However, we can tune the algorithm to detect more planes on each facade (e.g. one plane at the front and one for all the points acquired inside the building). This allows for *Polyfit* to reconstruct some facades. However, this tuning cannot work for the reconstruction of the road and this prevents *Polyfit* from reconstructing the full scene.



(a) Raw point cloud

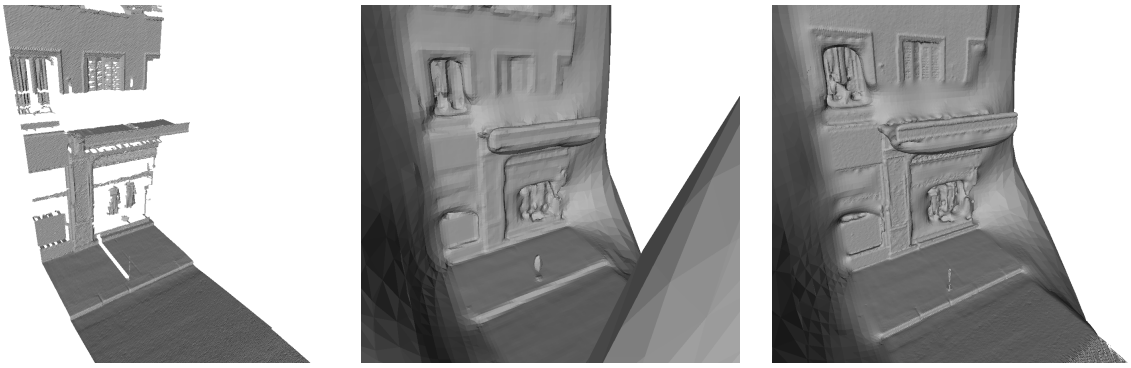
(b) Polyfit reconstruction for a fitness of 0.8, a complexity of 0.15 and a coverage of 0.05.

(c) Polyfit reconstruction for a fitness of 0.5, a complexity of 0.4 and a coverage of 0.1.

Figure 5.10 – Examples of reconstructions performed by *Polyfit*. We see that the road is not reconstructed and that parts of the facades are not reconstructed as well.

Unlike for Polyfit, the Poisson algorithm is able to perform a full reconstruction of the scene. However, the Poisson algorithm smooths the reconstruction, which prevents

it from being geometrically close the original data, especially in complex areas of the scene. The reconstructions with the smoothing effect can be seen on Figure 5.11. We notice that, even when increasing the point's weight to a high value of 1000 (default is 2), the final reconstruction is over-simplified.



(a) Raw point cloud

(b) Poisson reconstruction with a depth of 8.

(c) Poisson reconstruction with a depth of 12 and point weight to 1000.

Figure 5.11 – Examples of reconstructions performed by *Poisson reconstruction*. We see that this method smooths too much to fit the original data.

5.6 Conclusion

In this chapter, we addressed the problem of mesh simplification and argued that an accurate simplification should both preserve the geometry and produce a light model. We presented an approach for generalizing 3D meshes. Our approach is done in three steps, first by constructing simplicial complexes, then by segmenting the surface part of the simplicial complex in piecewise-planar components, and last, we simplify our mesh by means of edge-collapse technique. We evaluate the quality of the simplification by comparing different simplified meshes according to their geometric error and their MDL.

We acknowledge that in the present state, our method can, at best, be as good as an edge collapse on the full non-segmented mesh. However, we think that producing a set of polygons and not a mesh, as in Cohen-Steiner, Alliez, and Desbrun (2004) or Nan and Wonka (2017) and evaluating our approach using the MDL defined in 5.9 would lead to better results, especially by taking advantage of the large planar structures retrieved by ℓ_0-plane pursuit.

Last, the simplification method we used consists in giving already approximated data to the *edge collapse* algorithm (see Section 5.2). This only gives an indirect control on the simplification process. Moreover, the energy of the *edge collapse* implementation is the Lindstrom-Turk cost (Lindstrom and Turk, 1998) defined in Section 5.3.3.2, which is different from the metric we used to evaluate the final simplifciation. This means that the simplification process could be improved by computing a removal cost for each edge based on the squared distance between each removed /shifted point and the input mesh.

6 Weakly Supervised Segmentation-Aided Classification of LiDAR Point Clouds

Contents

In this chapter, we present a method for point cloud classification, which relies on local and global per point descriptors. In this context we studied the influence of simplicial complexes as 3D descritpors for classifying point clouds. We also investigated the use of a pre-segmentation step to improve classification results. We performed some experiments on the Paris dataset introduced in Chapter 3, but also on two public benchmarks: the Oakland (Munoz et al., 2009) and the semantic3D (Hackel, Wegner, and Schindler, 2016) benchmarks.

6.1 Introduction

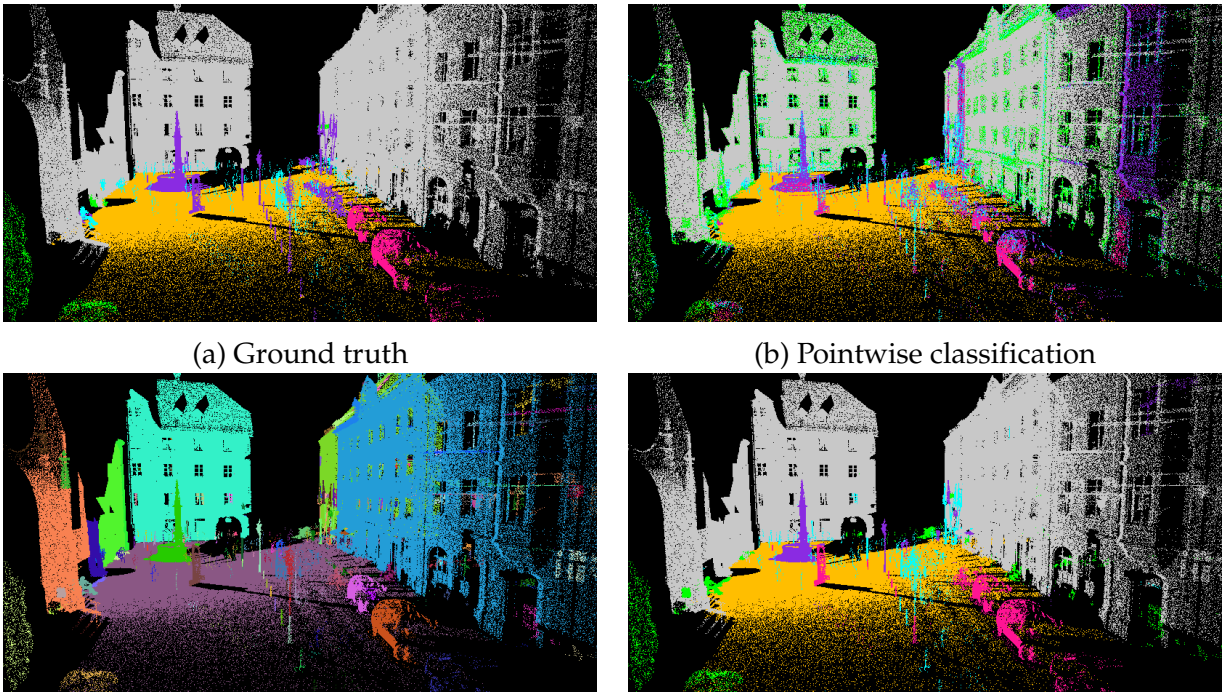
Automatic interpretation of large 3D point clouds acquired from terrestrial and mobile LiDAR scanning systems has become an important topic in the remote sensing community (Munoz et al., 2009; Weinmann et al., 2015a; Xu et al., 2019), yet it presents numerous challenges. Indeed, the high volume and the irregular structure of LiDAR point clouds make assigning a semantic label to each point a difficult endeavor. Furthermore, producing a precise ground truth is particularly difficult, time-consuming and can prove to be expensive. However, LiDAR scans of urban scenes display some form of regularity which can be exploited to improve the accuracy of a noisy semantic labeling.

Foremost, the high precision of LiDAR acquisitions implies that the number of points far exceeds the number of objects in a scene. Consequently, the sought semantic labeling can be expected to display high spatial regularity. In order to take into account the expected spatial regularity, Weinmann et al. (2015a) propose to classify a point cloud using descriptors computed on a local neighborhood for each point. However, the resulting classification is not regular in general, as observed in Figure 6.1b. The regularity prior has been incorporated into context-based graphical models (Anguelov et al., 2005; Shapovalov, Velizhev, and Barinova, 2010; Niemeyer, Rottensteiner, and Soergel, 2014) and a structured regularization framework (Landrieu et al., 2017), significantly increasing the accuracy of input pointwise classifications.

Pre-segmentations of point clouds have been used to model long-range interactions and to decrease the computational burden of the regularization (Rutzinger et al., 2008; Vosselman, Coenen, and Rottensteiner, 2017). The segments thus obtained can be incorporated into multi-scale graphical models to ensure a spatially-regular classification. However, the existing models require setting some constraints on the segments in advance, such as a maximum radius (Niemeyer et al., 2016; Golovinskiy, Kim, and Funkhouser, 2009), a maximum number of points in each segment (Lim and Suter, 2009), or the total number of segments (Shapovalov, Velizhev, and Barinova, 2010).

The aim of the work presented in this chapter is to leverage the underlying structure of the point cloud to improve a weak classification—obtained from very few annotated points—with a segmentation that requires no preset size parameters. We observe that the structure of urban scenes is mostly shaped by man-made objects (roads, façades, cars...), which are geometrically simple in general. Consequently, well-chosen geometric features associated to their respective points can be expected to be spatially regular. Also, the extent and number of points of the segments can vary a lot depending on the nature of the corresponding objects. Hence, we propose a formulation of the segmentation as a structured optimization problem in order to retrieve geometrically simple super-voxels. Unlike other presegmentation approaches, our method allows the segments' size to adapt to the complexity of the local geometry, as illustrated in Figure 6.1c.

Following the machine-learning principle that an ensemble of weak classifiers can perform better than a strong one (Opitz and Maclin, 1999), a consensus prediction is obtained from the segmentation by aggregating over each segment the noisy predictions of its points obtained from a classifier trained with only a few samples. The structure induced by the segmentation and the consensus prediction can be combined into a conditional random field formulation to directly classify the segments, and reach competitive performances from a very small number of hand-annotated points.



(a) Ground truth

(b) Pointwise classification

(c) Geometrically homogeneous segmentation

(d) Segmentation-aided regularization

Figure 6.1 – Illustration of the different steps of our method: the pointwise, irregular classification 6.1b is combined with the geometrically homogeneous segmentation 6.1c to obtain a smooth, objects-aware classification 6.1d. In Figures 6.1a, 6.1b , 6.1d, the semantic classes are represented with the following color code: vegetation, façades, hardscape, acquisition artifacts, cars, roads. In Figure 6.1c, each segment is represented by a random color.

6.1.1 Related Work

In this section we present works in the litterature related to our method. First, we focus on pointwise classification algorithms, then we present some context-based graphical models. Last, we investigate the pre-segmentation approaches.

6.1.1.1 Point-Wise Classification

Weinmann et al. (2015a) propose a classification framework based on 3D geometric features which are derived from local neighborhood of optimal size. This framework is the most common approach for point cloud classification (Vicari et al., 2019; Xing et al., 2019). These 3D geometric features can be combined with more specific descriptors such as the vegetation index (Chen et al., 2017).

6.1.1.2 Context-Based Graphical Models

The spatial regularity of a semantic labeling can be enforced by graphical models such as MRF (Anguelov et al., 2005; Shapovalov, Velizhev, and Barinova, 2010), and its discriminative counterpart, the Conditional Random Field (CRF) (Niemeyer, Rottensteiner, and Soergel, 2014; Landrieu, Weinmann, and Mallet, 2017). The unary terms, encoding data fidelity, are computed by a point-wise classification with a random forest classifier (Breiman, 2001), while the pairwise terms, inducing spatial regularity, encode the probability of transition between the semantic classes.

6.1.1.3 Pre-Segmentation Approaches

A pre-segmentation of the point cloud can be leveraged to improve the classification. Lim and Suter (2009) propose to define each segment as a node in a multi-scale CRF. The super-voxels are defined by a growing region method based on a predefined number of points in each pixel, and a color homogeneity prior. In Niemeyer et al. (2016), the segments are determined using a prior pointwise-classification. A multi-tier CRF is then constructed containing both points and voxels nodes. An iterative scheme is then performed, which alternates between inference in the multi-tier CRF and the computation of the semantically homogeneous segments with a maximum radius constraint. In Shapovalov, Velizhev, and Barinova (2010), the presegmentation is obtained through the k-means algorithm, which requires defining the number of clusters in the scene in advance. Furthermore k-means produces isotropic clusters whose size doesn't adapt to the geometrical complexity of the scene. In Dohan, Matejek, and Funkhouser (2015), a hierarchical segmentation is computed using the foreground/background segmentation of Golovinskiy, Kim, and Funkhouser (2009), which uses a preset horizontal and vertical radius as parameters. The segments are then hierarchically merged then classified.

6.1.1.4 Deep-Learning-based Approaches

Deep-Learning methods are also used for the pointwise classification of point clouds (Qi et al., 2017; Soilán Rodríguez et al., 2019). These methods perform better than classical appraoches based on Random Forests or SVMs (Raczko and Zagajewski, 2017). In order to improve a pointwise classification, some works propose to segment the point cloud and learn to classify the segments (Landrieu and Simonovsky, 2018; Hua, Tran, and Yeung, 2018; Tchapmi et al., 2017). However, training a NN requires a large set of annotations and is time-consuming. In this chapter, we focus on data-efficiency, by training a classifier with a few annotated points and NN are not suited for this task.

6.1.2 Problem Formulation

We consider a 3D point cloud V corresponding to a LiDAR acquisition in an urban scene. Our objective is to obtain a classification of the points in V between a finite set of semantic classes $\mathcal{K}$. We consider that we only have a small number of hand-annotated points as a ground truth from a similar urban scene. This number must be small enough that it can be produced by an operator in a reasonable time, i.e. no more than a few dozen per class.

We present the consituent elements of our approach in this section, in the order in which they are called.

Feature and graph computation: For each point, we compute a vector of geometrical features, described in Sections 6.2.1, 6.2.2 and 6.2.3. In Section 6.2.4 we present how the adjacency relationship between points is encoded into a weighted graph.

Segmentation in geometrically homogeneous segments: The segmentation problem is formulated as a structured optimization problem presented in Section 6.3.1, and whose solution can be approximated by a greedy algorithm. In section 6.3.2, we describe how the higher-level structure of the scene can be captured by a graph obtained from the segmentation.

Contextual classification of the segments: In Section 6.4, we present a CRF which derived its structure from the segmentation, and its unary parameter from the aggregation of the noisy prediction of a weakly supervised classifier. Finally, we associate the label of the corresponding segment to each point in the point cloud.

6.2 Features and Graph Computation

In this section, we present the descriptors chosen to represent the local geometry of 3D points, and the adjacency graph capturing the spatial structure of point clouds.

With a view that the training set is small, and to keep the computational burden of the segmentation to a minimum, we voluntarily limit the number of descriptors used in our pointwise classification. We insist on the fact that the segmentation and the classification do not necessarily use the same descriptors.

We first present some local descriptors computed for the local neighborhood of each point, then some global descritors computed with a knowledge of the whole data, and finally we encode the simplicial complexes as created in Chapter 3. We argue that the combination of such descriptors is able to improve the performances of a pointwise classification compared to using only local or global descriptors.

6.2.1 Local Descriptors

In order to describe the local geometry of each point we define four descriptors: linearity, planarity, scattering and verticality, which we represent in Figure 6.5.

The features are defined from the local neighborhood of each point of the cloud. For each neighborhood, we compute the eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$ of the covariance matrix of the positions of the neighbors. The neighborhood size is chosen such that it minimizes the eigentropy E of the vector $\begin{bmatrix} \lambda_1/\Lambda \\ \lambda_2/\Lambda \\ \lambda_3/\Lambda \end{bmatrix}$ with $\Lambda = \sum_{i=1}^{3} \lambda_i$, in accordance with the optimal neighborhood principle advocated in Weinmann et al. (2015a):

$$E = -\sum_{i=1}^{3} \frac{\lambda_i}{\Lambda} \log(\frac{\lambda_i}{\Lambda}). \tag{6.1}$$

As presented in Demantke et al. (2011), these eigenvalues allow us to qualify the shape

of the local neighborhood by deriving the following vectors:

$$Linearity = \frac{\lambda_1 - \lambda_2}{\lambda_1} \tag{6.2}$$

$$Planarity = \frac{\lambda_2 - \lambda_3}{\lambda_1} \tag{6.3}$$

$$Scattering = \frac{\lambda_3}{\lambda_1} \, . \tag{6.4}$$

The linearity describes how *elongated* the neighborhood is, while the planarity assesses how well it is fitted by a plane. Finally, high-scattering values correspond to an isotropic and spherical neighborhood. The combination of these three features is called dimensionality.

In our experiments, the vertical extent of the optimal neighborhood proved crucial for discriminating roads and façades, and between poles and electric wires, as they share similar dimensionality. To discriminate this class, we introduce a novel descriptor called *verticality* also obtained from the eigen vectors and values defined above. Let u_1, u_2, u_3 be the three eigenvectors associated with $\lambda_1, \lambda_2, \lambda_3$ respectively. We define the unary vector of principal direction in $\mathbb{R}^3_+$ as the sum of the absolute values of the coordinate of the eigenvectors weighted by their eigenvalues:

$$[\hat{u}]_i \propto \sum_{j=1}^{3} \lambda_j \, |[u_j]_i|, \text{ for } i = 1, 2, 3 \text{ and } \|\hat{u}\| = 1 \tag{6.5}$$

We argue that the vertical component of this vector characterizes the verticality of the neighborhood of a point. Indeed it reaches its minimum (equal to zero) for a horizontal neighborhood, and its maximum (equal to 1) for a linear vertical neighborhood. A vertical planar neighborhood, such as a façade, will have an intermediary value (around 0.7). This behavior is illustrated at Figure in Figure 6.5.

To illustrate the expressiveness of the selected features, we represent their respective value and range in Figure 6.2.

6.2.2 Non-Local Descriptors

Although the neighborhoods' shape of 3D points determine their local geometry, and allows us to compute a geometrically homogeneous segmentation, this is not sufficient for classification as we can't distinguish all the classes using only the local descriptors. Consequently, we use two descriptors of the global position of points in the point cloud: elevation and position with respect to the road.

Computing those descriptors first requires determining the extent of the road with a high precision. A binary road/non-road classification is performed using only the local geometry descriptors and a random forest classifier, which achieves very high accuracy and a F-score over 99.5%. This very high score can be explained by the fact that our local descriptors are able to efficiently discriminate the road from other classes, as shown on Figure 6.2. From this classification, a simple elevation model is computed, allowing us to associate a normalized height with respect to the road to each 3D point.

To estimate the position with respect to the road we compute the two-dimensional α-shape (Akkiraju et al., 1995) of the points of the road projected on the zero elevation level, as represented in Figure 6.4. This allows us to compute the *position with respect to the road* descriptor, equal to 1 if a point is outside the extent of the road and -1 otherwise. However, we think that this method does not allow to recover an error of

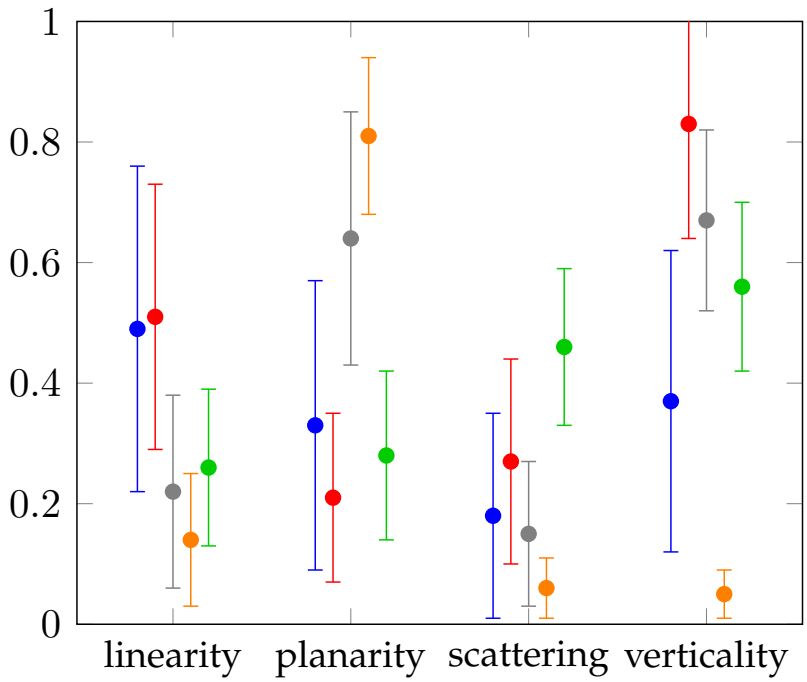


Figure 6.2 – Means and standard deviations of the local descriptors in the Oakland dataset for the following classes: wires, poles, façades, roads, vegetation.

the α-shape algorithm. Thus, for points close to the border (in pratice we chose a 1 m threshold), we chose to set the descriptor not to 1 or -1, but to the distance between the point and the border of the road. Figure 6.3 represents the value of the descriptor *position with respect to the road* compared to the signed euclidian distance between a point on the road boundary.

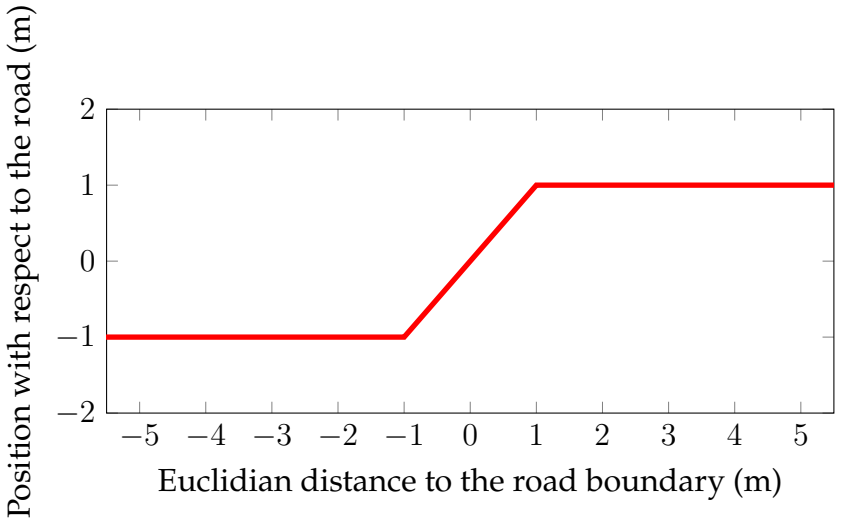


Figure 6.3 – Position with respect to the road, compared to the euclidian distance between a point and road boundary. Points inside the extent to the road have a negative distance to the boundary.

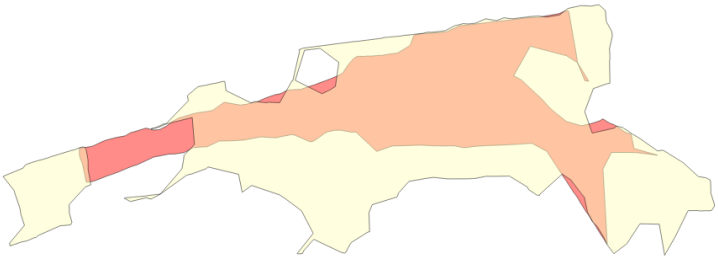
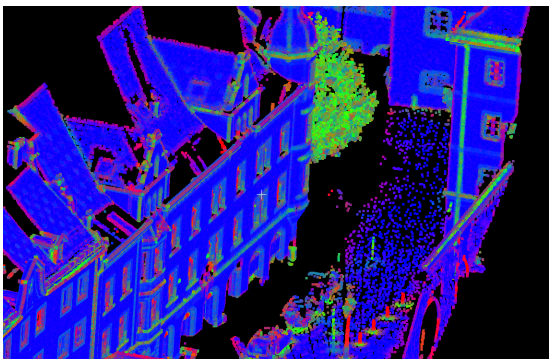
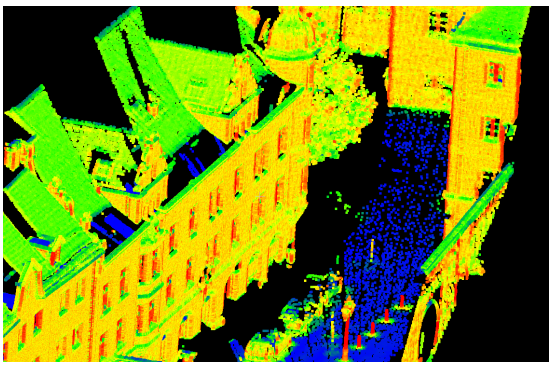


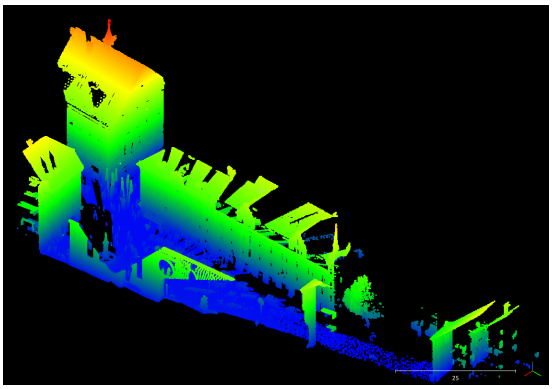
Figure 6.4 – α-shape of the road on our Semantic3D example. In red, the horizontal extent of the road; in yellow, the extent of the non-road class.



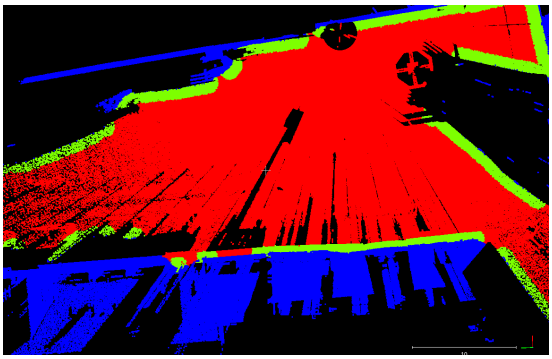
(a) Dimensionality



(b) Verticality



(c) Elevation



(d) Position with respect to the road

Figure 6.5 – Representation of the four local geometric descriptors as well as the two global descriptors. In (a), the dimensionality vector [linearity, planarity, scattering] is color-coded by a proportional [red, green, blue] vector. In (b), the value of the verticality is represented with a color map going from blue (low verticality - roads) to green/yellow (average verticality - roofs and façades) to red (high verticality - poles). In (c), is represented the elevation with respect to the road. In (d), the position with respect to the road is represented with the following color-code: inside the road α-shape in red, bordering in green, and outside in blue.

6.2.3 Simplicial Complexes as Descriptors

We now focus on the influence of simplicial complexes for point cloud classification. The rational is that simplicial complexes could provide some useful geometric information: for instance, roads are composed of triangles and wires are composed of edges (Beksi and Papanikolopoulos, 2016). We now present how a 3D structure such

as simplicial complexes can be converted as a set of 3D descriptors.

6.2.3.1 From Simplicial Complexes to 3D Descriptors

Simplicial complexes are used in this thesis as a 3D structure for point clouds, but it is not straightforward to find a suitable way to encode them as a 3D descriptor for point cloud classification. We chose to represent simplicial complexes based on the dimension of simplices connected to a single point. This means that a single point can be associated to three different informations:

— the point is left alone (simplex of dimension 0),

— the point is connected to at least an edge (simplex of dimension 1),

— the point is connected to at least a triangle (simplex of dimension 2).

In fact, as shown on Figure 6.6, a single point can be connected to simplices of different sizes. Intuitively, one could associate to a point the mean dimension of all simplices to which the point belong. Figure 6.6 shows a point with 5 neighbors. This point is connected with a triangle to 2 of its neighbors, it is connected with edges to 2 other neighbors and it is unconnected to the last neighbor. Here, we define the mean of the dimension of the simplices associated to the a point as the sum of connected simplices' dimensions divided by the number of neighbors. Also, for a triangle, we do not take into account its edges. For the red point of Figure 6.6, this number is 0.8 (1 point unconnected + 2 edges of dimension one + 1 triangle of dimension 2, divided by 5 neighbors). This number is very close to the dimension of an edge. However, in the case of Figure 6.6, we want to let the simplex of higher dimension prevail as this simplex is the one giving the most discriminating information among all the simpleces connected to a single point. In this case, it is a triangle. This means that, for a single point we associate the following codes:

— 2 if the point is connected to at least one triangle,

— 1 if the point is connected to 0 triangles and at least one edge,

— 0 if the point is not connected to any other simplex.

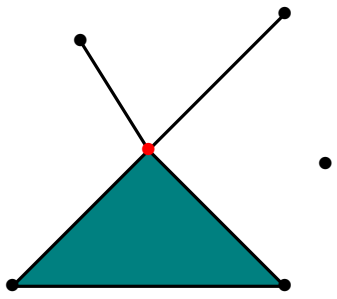


Figure 6.6 – Illustration of a simplicial complex at a point scale. We consider the red point: it is connected to 1 triangle and 2 edges. All the black points are neighbors of the red point.

6.2.3.2 Encoding of Descriptors

The main problem with the encoding presented in the previous section, is that it induces an inaccurate relationship between the different simplices: an edge could be

interpreted as a mean between a point and a triangle. Here, we want to distinguish the three possible cases and give them an equal importance. Hence, we used the one-hot encoding method to represent them.

One-hot encoding [36]: One-hot encoding is based on the one-hot principle (Huffman, 1954). For a single group of bits, we call them *one-hot* if and only if only one of the bits is high (1) and all the others are low (0). It was first used to find the current state of state-machines [37] (Golson, 1993). This concept was then adapted to machine learning. In fact, some algorithms (such as SVMs) are unable to process categorical data [38].

In machine learning, one-hot encoding is used for representing categorical data with no ordering relationship in order to feed algorithms unable to process them otherwise. These algorithms include SVMs and some deep learning approaches (Barsan et al., 2018; Salberg, Trier, and Kampffmeyer, 2017). In this thesis, we represent our simplicial complexes as a set of three descriptors for each point with the code presented in Table 6.1.

	Point	Edge	Triangle
0D	1	0	0
1D	0	1	0
2D	0	0	1

Table 6.1 – Encoding of simplicial complexes as 3D descriptors.

We display the simplicial complexes encoded as shown on Table 6.1 on Figure 6.8. We show in blue the points which are connected to a triangle, in green those which are connected to an edge and no triangles. The remaining points are colored in red. We observe that most of the road and the buildings are colored in blue, whereas the tree is colored in blue for the trunk and main branches, twigs are colored in green and leaves are red. We can observe on this figure that simplicial complexes are extremely noisy, due to their sensitivity to the position of the sensor. This can lead to misclassifications in the case of a weakly supervised classification, as the classifier has very little samples to train on.

6.2.4 Adjacency Graph

The spatial structure of a point cloud can be represented by an unoriented graph $G = (V, E)$, in which the nodes represent the points of the cloud, and the edges encode their *adjacency relationship*. We compute the 10-nearest neighbors graph, as advocated in (Niemeyer et al., 2011). We remark that this graph defines a symmetric graph-adjacency relationship which is *different* from the optimal neighborhood used in Section 6.2.1.

36. Based on `https://machinelearningmastery.com/why-one-hot-encode-data-in-machine-learning/`

37. A state machine is an abstract machine that can be only is one state from a finite number of states at a given time.

38. Categorical data do not contain numeric values but rather labels such as *dog* or *cat*. Such labels may or may not have a relationship between them.

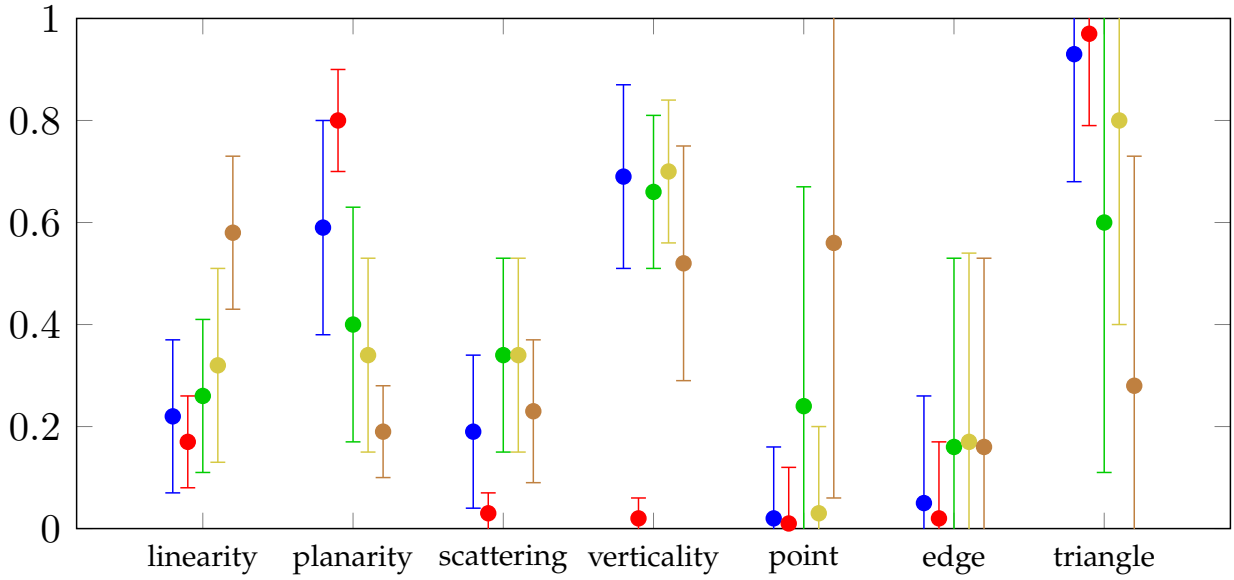
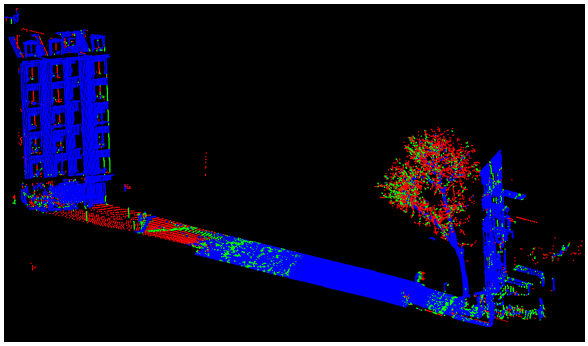
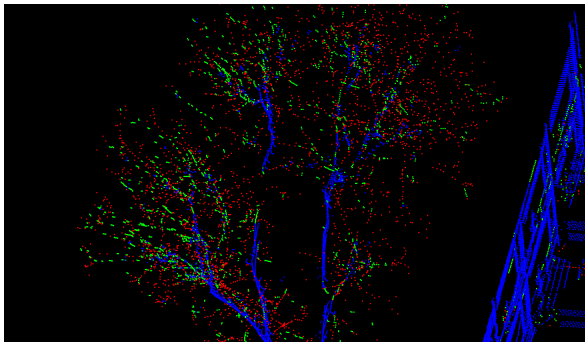


Figure 6.7 – Means and standard deviations of the local descriptors and the simplicial complexes in the Paris dataset for the following classes: Facades, Road, Vegetation, hardscape, Other.



(a) Representation of the simplicial complexes on the Paris dataset.



(b) Representation of the simplicial complexes on the top of a tree on the Paris dataset.

Figure 6.8 – Representation of simplicial complexes on the Paris dataset using the following color code: triangles, edges and points. Note that due to our geometric priors when reconstructing the simplicial complexes, a part of the road that is too far from the sensor is not reconstructed, thus only represented by unconnected points. This figure is best viewed in color.

6.3 Segmentation into Homogeneous Segments

6.3.1 Potts Energy Segmentation

To each point, we associate its local geometric feature vector $f_i \in \mathbb{R}^4$ (dimensionality and verticality), and compute a piecewise constant approximation $g^\star$ of the signal $f \in \mathbb{R}^{V\times 4}$ structured by the graph G. $g^\star$ is defined as the vector of $\mathbb{R}^{V\times 4}$ minimizing the following *Potts segmentation energy*:

$$g^\star = \underset{g\in\mathbb{R}^{4\times V}}{\arg\min} \sum_{i\in V} \|g_i - f_i\|^2 + \rho \sum_{(i,j)\in E} \delta(g_i - g_j \neq 0) \;, \tag{6.6}$$

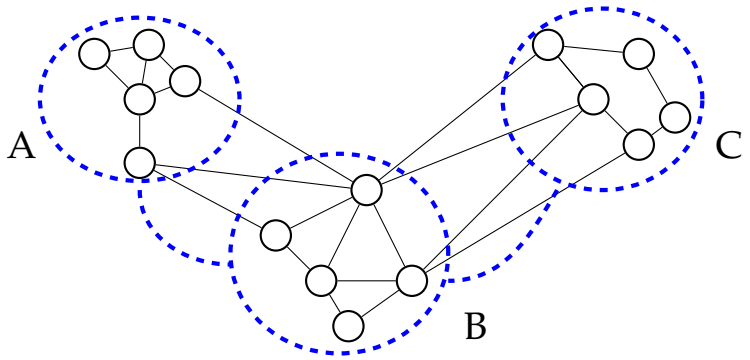


Figure 6.9 – Adjacency structure of the segment-graph. The edges between points are represented in black —— , the segmentation and the adjacency of its components in blue: - - - - . In this figure, we have $\mathbb{S} = \{A, B, C\}$ and $\mathbb{E} = (\{A, B\}, \{B, C\})$.

with $\delta(\cdot \neq 0)$ the function of $\mathbb{R}^4 \mapsto \{0, 1\}$ equal to 0 in 0 and 1 everywhere else. The first part of this energy is the fidelity function, ensuring that the constant components of $g^\star$ correspond to homogeneous values of f. The second part is the regularizer which adds a penalty for each edge linking two components with different values. This penalty enforces the simplicity of the shape of the segments. Finally ρ is the regularization strength, determining the trade off between fidelity and simplicity, and implicitly determining the number of clusters.

This structured optimization problem can be efficiently approximated using the greedy graph-cut based ℓ_0-cut pursuit algorithm presented in Landrieu and Obozinski (2016). The segments are defined as the constant connected components of the piecewise constant signal $g^\star$ obtained.

The benefit of this formulation is that it does not require defining a maximum size for the segments in terms of extent or points. Indeed large segments of similar points, such as roads or façades, can be retrieved. On the other hand, the granularity of the segments will increase where the geometry gets more complex, as illustrated in Figure 6.1c.

For the remainder of the chapter we denote $\mathbb{S} = (S_1, \cdots, S_k)$ the non-overlapping segmentation of V obtained when approximately solving the optimization problem presented in Equation 6.6.

6.3.2 Segment-Graph

We argue that since the segments $(S_1, \cdots, S_k)$ represent objects or significant parts of objects of the scene, the segmentation represents its underlying high-level structure. To obtain the relationship between objects, we build the *segment-graph*, which is defined as $\mathcal{G} = (\mathbb{S}, \mathbb{E}, w)$ in which the segments of $\mathbb{S}$ are the nodes of $\mathcal{G}$. $\mathbb{E}$ represents the adjacency relationship between segments, while w encodes the weight of their boundary, as represented in Figure 6.9. We define two segments as adjacent if there is an edge in E linking them, and w as the total weight of the edges linking those segments:

$$\begin{cases} \mathbb{E} & = \{(s,t) \in \mathbb{S}^2 \mid \exists (i,j) \in E \cap (s \times t)\} \\ w_{s,t} & = |E \cap (s \times t)|\,, \quad \forall (s,t) \in \mathbb{S}^2\,. \end{cases} \tag{6.7}$$

6.4 Contextual Classification of the Segments

To enforce spatial regularity, Niemeyer, Rottensteiner, and Soergel (2014) defines the optimal labeling $l^\star$ of a point cloud as maximizing the posterior distribution $p(l \mid f')$ in a CRF model structured by an adjacency graph G, with f' the vector of local and global features. We denote a labeling of V by a vector l such that $\{l \in \{0,1\}^{V\times\mathcal{K}} \mid \sum_{k\in\mathcal{K}} l_{i,k} = 1, \forall i \in V\}$, with $l_{i,k}$ equal to one if the point i of V is labelled as $k \in \mathcal{K}$, and zero else. For a point i of V, l_i is the vector of $\mathbb{R}^{\mathcal{K}}$ such that it encodes the labeling of i as k. l_i contains zeros except for its k^{th} value, which is 1.

This allows us to define $l^\star$ as the maximizing argument of the following energy:

$$l^\star = \underset{l\in\Delta(V,\mathcal{K})}{\arg\max} \sum_{i\in V} l_i^\intercal p^i + \sum_{(i,j)\in E} l_i^\intercal M^{(i,j)} l_j, \tag{6.8}$$

with $p_k^i = \log(p(l_i = k \mid f_i'))$ the entrywise logarithm of the probability of node i being in state k, and $M_{(i,j),(k,l)} = \log(p(l_i = k, l_j = l \mid f_i', f_j'))$ the entrywise logarithm of the probability of observing the transition (k,l) at (i,j).

As advocated in Niemeyer, Rottensteiner, and Soergel (2014), we can estimate $p(l_i = k \mid f_i')$ with a random forest probabilistic classifier p_{RF}. p_{RF} can be obtained as the number of trees in the Random Forest voting for each class, compared to the total number of trees. To avoid infinite values, the probability p_{RF} is smoothed by taking a linear interpolation with the constant probability: $p(k \mid f_i) = (1-\alpha)p_{\text{RF}}(k \mid f'_i) + \alpha/\left|\mathcal{K}\right|$ with $\alpha = 0.01$ and $|\mathcal{K}|$ the cardinality of the class set. The authors also advocate learning the transition probability from the difference of the features vectors. However, our weak supervision hypothesis prevents us from learning the transitions, as it would require annotations covering the $|\mathcal{K}|^2$ possible combinations extensively. Furthermore the annotation would have to be very precise along the transitions, which are often hard to distinguish in point clouds. We make the simplifying hypothesis that M is of the following form :

$$M_{(k,l)}^{(i,j)} = \begin{cases} 0 & \text{if } k = l \\ \sigma & \text{else,} \end{cases} \tag{6.9}$$

with σ a non-negative value, which can be determined by cross-validation.

Leveraging the hypothesis that the segments obtained in in Section 6.3.1 correspond to semantically homogeneous objects, we can assume that the optimal labeling will be constant over each segment of $\mathbb{S}$. Consequently, we propose a formulation of a CRF structured by the segment-graph $\mathcal{G}$ to capture the organization of the segments. We denote $L^\star$ the labeling of $\mathbb{S}$ defined as:

$$L^\star = \underset{L\in\Delta(\mathbb{S},\mathcal{K})}{\arg\max} \sum_{s\in\mathbb{S}} L_s^\intercal P^s + \sum_{(s,t)\in\mathbb{E}} w_{s,t} L_s^\intercal M L_t^\intercal, \tag{6.10}$$

with $P_k^{(s)} = |s| \log(p(L_s = k \mid \{f_i'\}_{i\in s}))$ the logarithm of the probability of segment s being in state k multiplied by the cardinality of s. We define this probability as the average of the probability of each point of the segment to be labeled as k:

$$p(L_s = k \mid \{f_i'\}_{i\in s}) = \frac{1}{|s|} \sum_{i\in s} p(l_i = k \mid f_i'). \tag{6.11}$$

Note that the influence of the data term of a segment is determined by its cardinality, since the classification of the points remains the final objective. Likewise, the cost of

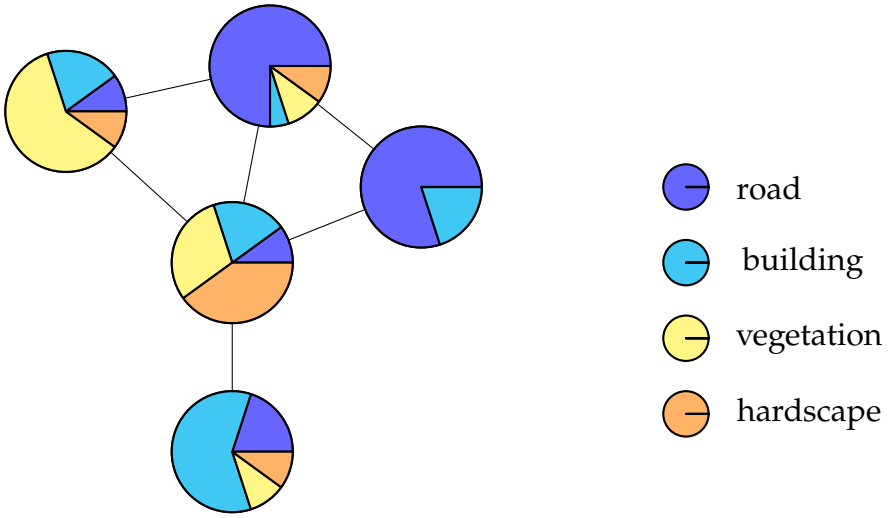


Figure 6.10 – Illustration of the classification of the segments, based on the sum of point's classifications probabilities. Each pie corresponds to a single segment, and the adjacency relationship illustrates the segment-graph. Each color represents a different class.

a transition between two segments is weighted by the total weight of the edges at their interface $w_{s,t}$, and represents the magnitude of the interaction between those two segments.

Following the conclusions of Landrieu, Weinmann, and Mallet (2017), we approximate the labelling maximizing the log-likelihood with the maximum-a-priori principle using the α-expansion algorithm of Boykov, Veksler, and Zabih (2001), with the implementation of Schmidt (2007). An illustration of the classification of the segments is presented on Figure 6.10.

It is important to remark that the segment-based CRF only involves the segment-graph $\mathcal{G}$, which can be expected to be much smaller than G, making inference potentially much faster.

6.5 Numerical Experiments

We now demonstrate the advantages of our approach through numerical experiments on three different datasets. First, we introduce the data and our evaluation metric, then present the classification results compared to state-of-the-art methods at the time of Guinard and Landrieu (2017).

6.5.1 Data

To validate our approach, we consider two publicly available data sets and one private dataset.

The first dataset is the Paris dataset which has been acquired using the Stereopolis vehicle (Paparoditis et al., 2012) in Paris. It is divided in 5 classes: facade, road, vegetation, hardscape (which comprises poles and traffic signs) and a class for unkown objects named *other*. This dataset has been manually classified using the CloudCompare software [39]. The labeling process is presented in Appendix A. We used this dataset

39. `https://www.danielgm.net/cc/`

because it contains the raw acquisition data, allowing for a simplicial complex reconstruction of the scene (see Chapter 3). The only experiment done on this dataset is the evaluation of the influence of simplicial complexes for the pointwise classification.

We then consider the urban part of the Oakland benchmark introduced in Munoz et al. (2009), comprised of 655,297 points acquired by mobile LiDAR. Some classes have been removed from the acquisition (i.e. cars or pedestrians) such that there are only 5 left: electric wires, poles/trunks, façcades, roads and vegetation. We choose to exclude the tree-rich half of the set as the segmentation results are not yet satifying at the trunk-tree interface.

We also consider one of the urban scenes in the Semantic3D benchmark [40] (Hackel, Wegner, and Schindler, 2016), downsampled to 3.5 millions points for memory reasons. This scene, acquired with a fixed LiDAR, contains 6 classes : road, façade, vegetation, car, acquisition artifacts and hardscape. Although the original dataset comprises 8 different classes, there was no *natural terrain* in the scene we chose for testing. Also we decided to merge the two vegetation classes (*high vegetation* and *low vegetation*) as they were really similar regarding our descriptors.

For each class and dataset we hand-pick a small number of representative points such that the discriminative nature of our features illustrated in Figure 6.2 is represented. We select 10 points per classes for Paris, 15 for Oakland and 25 to 35 points for semantic3D, for respective totals of 50, 75 and 180 points.

6.5.2 Metric

To take into account the imbalanced distribution of each class (roads and façades comprise up to 80% of the points), we use the unweighted average of the F-score to evaluate the classification results. The F-Score is based on the notion of *precision* and *recall*, which are themselves based on the notions of True Positives and Negatives, and False Positives and Negatives. In this context, we consider a single point i and a single class k. The point is:

— True Positives (TP): if i belongs to class k and has been labeled as k,

— True Negatives (TN): if i does not belong to class k and has not been labeled as k,

— False Positives (FP): if i belongs to class k and has not been labeled as k,

— False Negatives (FN): if i does not belong to class k and has been labeled as k.

Precision and recall are introduced respectively in Equations 6.12:

$$\text{precision} = \frac{\text{TP}}{\text{TP + FP}} \,, \tag{6.12}$$

and 6.13:

$$\text{recall} = \frac{\text{TP}}{\text{TP + FN}} \,. \tag{6.13}$$

The F-Score per class is given in Equation 6.14. The overall F-Score is computed as the mean of all per-class F-Scores. Consequently, a classification with decent accuracy over all classes will have a higher score than a method with high accuracy over some classes but poor results for others.

$$\text{FScore} = 2 \cdot \frac{precision \cdot recall}{precision + recall} \,. \tag{6.14}$$

40. http://www.semantic3d.net/

6.5.3 Competing Methods

To evaluate the efficiency of our implementation, we have implemented the following methods:

— **Pointwise:** we implemented the pointwise classification relying on optimal neighborhoods of Weinmann et al. (2015a), with a random forest (Breiman, 2001) and restricted ourselves to the six geometric features presented in Section 6.2.1.

— **CRF regularization:** we implemented a CRF-model as defined in (6.8) without aid from the segmentation.

Note that we do not compare our approach to data hungry methods such as *deep learning*-based methods as we are in a weakly-supervised context.

6.5.4 Results

We first present the evaluation of the influence of simplicial complexes for pointwise classification. This experiment was done on the Paris dataset. Then we compare our pointwise classification to a CRF-based approach and a pre-segmentation-based approach. The last experiments are done on the Oakland dataset and on the Semantic3D dataset.

6.5.4.1 Experiments with Simplicial Complexes

In this experiment, we trained a Random Forest classifier on the 50 points manually selected in the dataset (10 per class). We used the following parameters as they give better results:

— Maximum depth of a single tree: 50,

— Minimum sample of point per leaf: 1,

— Number of variables randomly selected at each node: 2,

— Maximum number of trees in the forest: 50.

We compared:

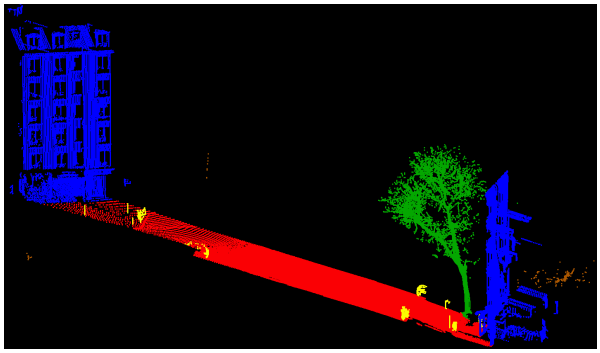
— a pointwise classification with only local descriptors named: *local*,

— a pointwise classification with the local descriptors, the elevation and the position with respect to the road named: *local* + *global*,

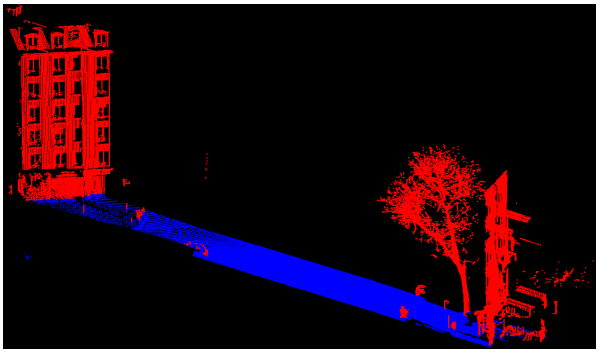
— a pointwise classification with the local descriptors and the one-hot encoded simplicial complexes named: *local* + *sc*,

— a pointwise classification with all the descriptors named: *local* + *global* + *sc*.

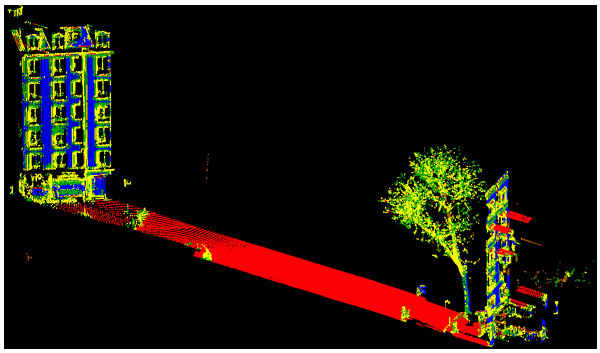
The results are visible on Figure 6.11 with a zoom on a tree in front of a facade in Figure 6.12. We observe that the classification *local* + *global* gives the best results, and is much less noisy than the classifications using the simplicial complexes. This is due to the fact that simplicial complex reconstructions are extremely noisy and they may confuse the classifier. The detailed results per class are available on Table 6.2. For the following experiments, we will use the best configuration here: local descriptors, plus the elevation and the position with respect to the road.



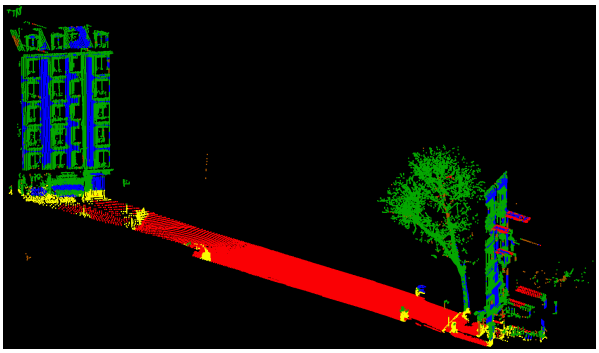
(a) Ground Truth for the Paris Dataset



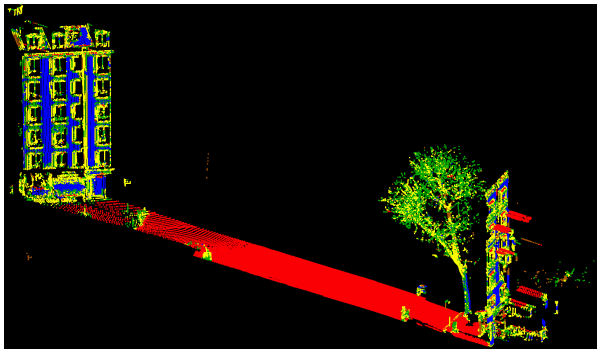
(b) Binary classification in road (blue) / non road (red)



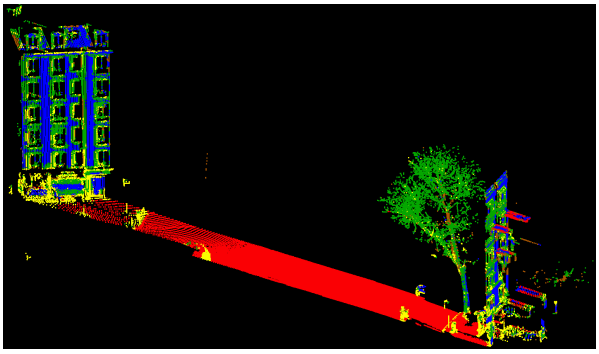
(c) Pointwise classification: local



(d) Pointwise classification: local + global

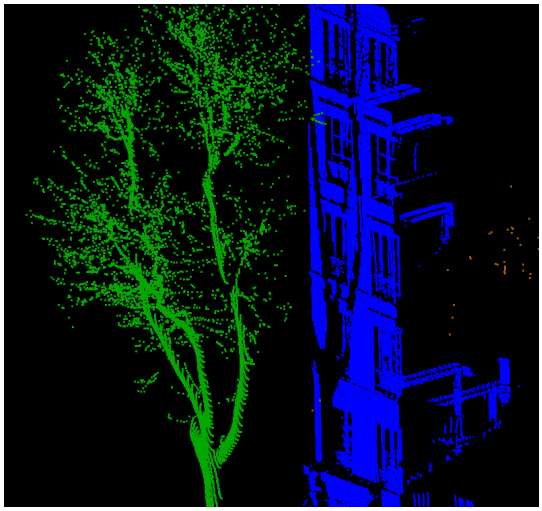


(e) Pointwise classification: local + sc

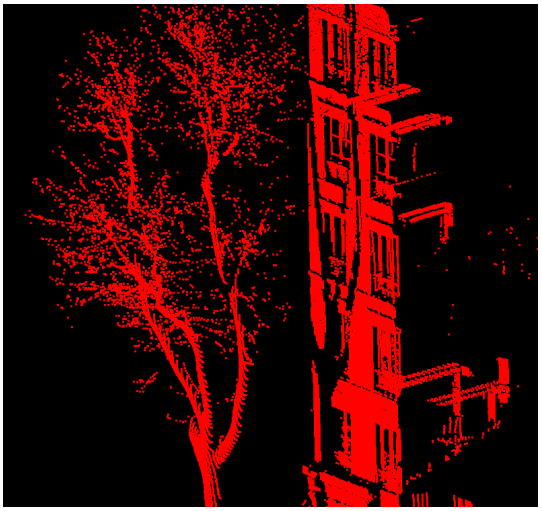


(f) Pointwise classification: local + global + sc

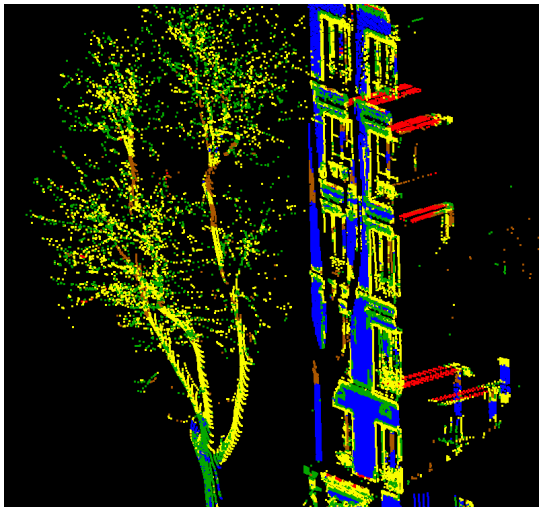
Figure 6.11 – Comparison of the classifications on the Paris Dataset, with only local descriptors (c), local descriptors + elevation and position with respect to the road (d), local descriptors and one-hot encoded simplicial complexes (e) and with all the descriptors (f). The semantic classes are represented with the following color code: buildings, road, vegetation, hardscape and other.



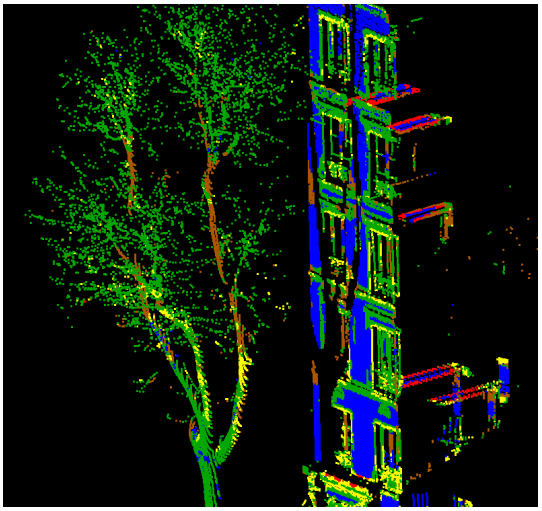
(a) Ground Truth for the Paris Dataset



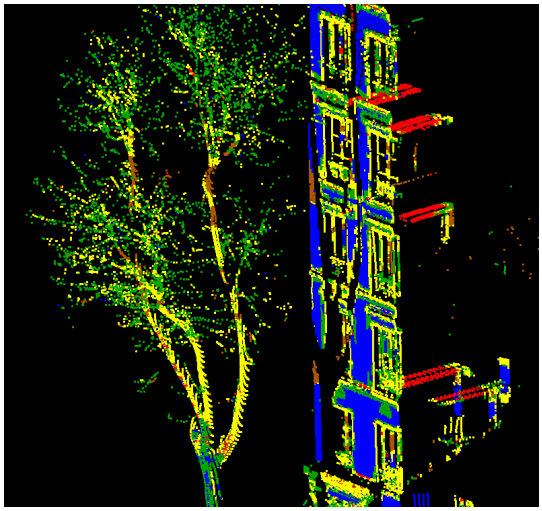
(b) Binary classification in road (blue) / non road (red)



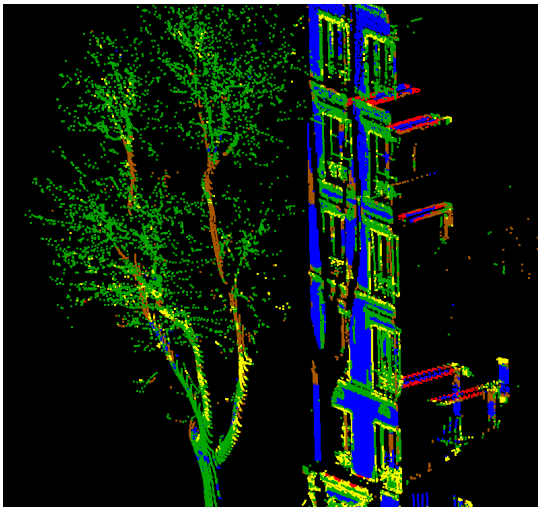
(c) Pointwise classification: local



(d) Pointwise classification: local + global



(e) Pointwise classification: local + sc



(f) Pointwise classification: local + global + sc

Figure 6.12 – Comparison of the classifications on a portion of the Paris dataset representing a tree in front of a facade, with only local descriptors (c), local descriptors + elevation and position with respect to the road (d), local descriptors and one-hot encoded simplicial complexes (e) and with all the descriptors (f). The semantic classes are represented with the following color code: buildings, road, vegetation, hardscape and other.

	Pointwise - local			Pointwise - local + global			Pointwise - local + sc			Pointwise - local + global + sc		
classes	precision	recall	FScore	precision	recall	FScore	precision	recall	FScore	precision	recall	FScore
building	94.1	28.9	44.3	**98.1**	30.2	46.2	93.0	28.4	43.5	95.2	**30.8**	**46.5**
road	98.2	98.8	98.5	**99.0**	**99.1**	**99.1**	97.2	98.9	98.0	98.9	98.9	98.9
vegetation	33.3	41.0	36.7	34.3	**89.4**	49.6	**37.4**	47.1	41.7	37.3	74.9	**49.8**
hardscape	5.2	57.7	9.6	**27.3**	**85.2**	**41.3**	4.9	51.9	8.9	14.9	84.1	25.3
other	6.8	66.4	12.3	**7.4**	65.7	**13.3**	6.9	63.5	12.5	4.5	**67.5**	8.5
Total	47.5	58.6	40.3	**53.2**	**73.9**	**49.9**	47.8	58.0	40.9	50.1	71.3	45.8

Table 6.2 – Precision, recall and FScore in % for the Paris dataset. The global accuracies are respectively 72.1%, 77.6%, 72.4% and 76.3%. In bold, we represent the best value in each category.

The classification using simplicial complexes did not perform as well as the classification with only 6 descriptors. This is mostly due to the noisy reconstruction: we can observe that on simple areas such as the road or the central part of buildings, the classification performs well. However, as soon as the geometry of the scene becomes more complex (e.g. close to windows), the classifier is unable to tell the difference between a building, vegetation or hardscape. As illustrated on Figure 6.7, the three descriptors associated to simplicial complex reconstructions are not discriminant enough for classifying the scene. This leads to an overall FScore lower than the one of the pointwise classification using only 6 descriptors. For the remaining experiments we decided to use only the first 6 descriptors. Some future experiments may focus on the regularization of simplicial complexes descriptors to improve point cloud classification.

6.5.4.2 Experiments on Public Benchmarks

In Tables 6.3 and 6.4, we represent the classification results of our method and the competing methods for both datasets. We observe that both the CRF and the presegmentation approach significantly improve the results compared to the pointwise classification. Although the improvement in term of global accuracy of our method compared to the CRF-regularization is limited (a few percents at best), the quality of the classification is improved significantly for some hard-to-retrieve classes such as poles, wires, and cars. Furthermore, our method provides us with an object-level segmentation as well.

For both experiments, we set the Random Forest parameters to:

— Maximum depth of a single tree: 25,

— Minimum sample of point per leaf: 1,

— Number of variables randomly selected at each node: 4,

— Maximum number of trees in the forest: 25.

We changed the parameters from the previous experiments due to the size difference of the point clouds compared to the Paris one.

	pointwise classification			CRF-regularization			our method		
classes	precision	recall	FScore	precision	recall	FScore	precision	recall	Fscore
wires	4.2	37.1	7.5	**87.8**	32.1	**47.0**	51.2	**35.4**	41.9
poles	9.0	**67.7**	15.9	**78.6**	37.7	51.0	66.1	48.3	**55.8**
façades	57.5	74.9	65.1	79.2	**98.0**	87.6	**91.0**	96.5	**93.6**
road	**99.9**	86.7	92.8	99.6	95.2	97.4	99.6	**99.1**	**99.4**
vegetation	85.5	82.8	84.1	93.5	93.1	93.3	**95.5**	**94.4**	**95**
total	51.2	69.8	53.1	**87.7**	71.2	75.2	80.7	**74.7**	**77.1**

Table 6.3 – Precision, recall and FScore in % for the Oakland benchmark. The global accuracies are respectively 85.2%, 94.8% and 97.3%. In bold, we represent the best value in each category.

	pointwise classification			CRF-regularization			our method		
classes	precision	recall	FScore	precision	recall	FScore	precision	recall	Fscore
road	**98.7**	96.8	97.7	97.6	**99.0**	**98.3**	97.5	98.7	98.1
vegetation	14.2	82.9	24.2	49.7	84.7	62.6	**52.1**	**93.7**	**67.0**
façade	99.6	88.1	93.5	99.5	97.9	98.7	**99.7**	**98.2**	**98.8**
hardscape	74.2	71.4	73.1	**93.7**	88.7	91.2	92.7	**90.4**	**91.5**
artifacts	18.3	37.5	24.6	**77.9**	**42.1**	**54.7**	73.8	39.3	51.3
cars	28.6	54.8	37.6	66.5	86.2	75.1	**84.0**	**90.0**	**82.3**
total	55.7	71.9	58.4	80.8	83.1	80.1	**83.3**	**85.0**	**82.3**

Table 6.4 – Precision, recall and FScore in % for the Semantic3D benchmark. The global accuracies are respectively 88.4%, 96.9% and 97.2%. In bold, we represent the best value in each category.

6.6 Conclusion

In this chapter, we presented a point cloud classification method aided by a geometric pre-segmentation capturing the high-level organization of urban scenes. We first investigated the influence of simplicial complexes as per points descriptors. In our experiments, simplicial complexes did not help the classifier. Instead they tend to harm the final classification. This is due to the high level of noise of our simplicial complexes. However, such structure provide useful information for a classification (e.g. the facades are mostly composed of triangles, poles or electric wires are composed of edges), so we think that a regularization step of the simplicial complexes descriptors could improve the overall classification.

We also showed that the pre-segmentation step allowed us to formulate a CRF model to directly classify the segments, improving the results over the CRF regularization. The same approach has been studied and tested at a larger scale with the SuperPoint Graph method (Landrieu and Simonovsky, 2018; Landrieu and Boussaha, 2019).

Further developments should focus on improving the quality of the segmentation near loose and scattered acquisition such as foliage. Another possible improvement would be to better exploit the context of the transition. Indeed the form of the transition matrix in (6.9) is very restrictive, as it does not take into account rules such as "roads are below the façade" or the "tree-trunk transitions are more likely than foliage-road transition". Although the weakly-supervised context excludes learning the transition, it would nonetheless be beneficial to incorporate such expertise from an operator.

7 Conclusion

Contents

7.1 Summary

In this thesis, we investigated the use of simplicial complexes for reconstructing 3D urban scenes acquired with a LiDAR sensor. We presented the constraints associated to the processing of LiDAR data, and notably the problems of missing data, and the lack of internal structure connecting 3D points. In urban scenes, there exists various cases in which an object hides a part of the scene. This can lead to holes and occlusions in LiDAR scans. In order to recover the missing geometry, there exists some techniques, such as inpainting or continuous reconstruction, that infer the missing geometry and add plausible information to the input scan. In this thesis, we chose to not add any information to the original data. Instead, we take into account all available information and rely on the topology of LiDAR sensors for reconstructing 3D scenes. We argue that some objects in urban scenes can have a higher geometric complexity than the scan resolution. Hence, we propose to only infer a continuous geometry (such as lines or planes) where the scan information allows it. The resulting object is a collection of points, edges, and triangles, and is called a simplicial complex. Reconstructing simplicial complexes allows us to preserve small geometrical details and gives meaningful information on the local geometry of the scene. We acknowledge that simplicial complexes do not allow for a watertight reconstruction and that their high locality makes them harder to interprete for visualization.

Moreover, simplicial complexes on urban scenes are composed of hundreds of thousands of simplices. Geometrically simple objects, such as roads or facades are composed of thousands of nearly coplanar triangles. We argue that such geometry can be precisely approximated by a small number of primitives. Hence, we decided to add a generalisation process focusing on simplices of dimension 2 (i.e. triangles). We designed a global approach that takes as input a set of points with an adjacency relationship (such as the triangles of our simplicial complexes) and iteratively divide a scene into a set of planar regions. Our approach, named ℓ_0-plane pursuit, iteratively segments an input point cloud in a set of planar regions. This method is adaptive to the geometry of the cloud, thus preserves the highly resolute geometry by locally refining the segmentation, while aggregating thousands of simplices for geometrically simple areas such as roads or facades. Our method showed promising results, both in terms of computation speed and geometric quality of the approximation.

Based on ℓ_0-plane pursuit results, we proposed an approach to merge coplanar triangles with the edge collapse algorithm. We evaluated the quality of the approximation by projecting each point of the original cloud to the simplified model. We also evaluated the *degree of generalization* of our models by computing their MDL, which corresponds to the quantity of information needed to represent the data. Our approach showed better results than classic 3D modeling approaches. This means that aggregating 2D-simplices within simplicial complexes is a valid method for generalising the 3D point clouds of urban scenes.

Last, we investigated the performances of simplicial complexes as data structure. More specifically, we evaluated the performance of classification algorithms operating on 3D point clouds. We showed how simplicial complexes could be used as 3D geometric descriptors. Following our objective of relying on as little extraneous information as possible, we trained a Random Forest with only a few samples for each semantic class. We argued that a pointwise classification of 3D point clouds tends to not be of high quality, but they can be spatially regularised using a segmentation algorithm, such as ℓ_0-cut pursuit. However, our experiments showed that simplicial complexes were too dependent from the local geometry of the scene to meaningfully

improve such pointwise classification.

In the end, we proposed a simple and lightweight method to reconstruct simplicial complexes from 3D LiDAR data acquisitions of urban scenes. We argued that simplicial complexes are an alternative for meshes that allows to preserve all the geometrical informations without modifying the data. We showed that simplicial complexes can be generalised with a limited number of primitives as well, and that global approaches can be used to produce geometrically accurate, yet compact, models of 3D urban scenes.

7.2 Perspectives

We now present some ideas for future works, based on simplicial complex reconstructions of urban scenes, taking advantage of the geometrical quality of such reconstructions.

7.2.1 Generalization of Simplicial Complexes

The main problem that arises when using simplicial complexes as structure is that their high locality can actually hinder classification algorithms, as seen in Chapter 6. Hence, future work might focus on the generalisation of simplicial complexes. A first approach, using a *wedge-based* regularization, has been presented in Section 3.2.3. This approach allows us to recover missing edges in the reconstruction. The next step would be to better use global knowledge of the scene, by regularizing the reconstruction in areas where simplices of low dimension are surrounded by simplices of a larger dimension. In fact, holes and occlusions can lead to missing information locally, which can prevent the reconstructions of lower dimensions. Hence, a regularization step, processing the scene with a global point of view should improve the reconstruction issues due to small holes.

This regularization step could be done, for instance, by optimising over the entire reconstruction. Such regularization should penalize the transitions between simplices of different dimensions, while taking into account the C_0 and C_1 regularities defined in Section 3.2.2.2. Also, geometrical descriptors, such as the dimensionality (Demantke et al., 2011) could be useful to improve the reconstruction as they provide complementary information to our own geometrical contraints.

7.2.2 Multi-Primitive-Based Generalization

The generalisation work undertaken in this thesis focused only on triangles. This was motivated by the fact that urban scenes are mostly composed of planar areas (such as roads). However, even if they may look planar at a very local scale, many objects are not planar. Poles and tree trunks are cylindrical, bushes, pedestrians, or cars actually have more complex shapes. Thus, future works on simplicial complexes may focus on multi-primitive-based generalisation. This generalisation could include cylinders, but also spheres or tores (Wang and Shi, 2014). Complex objects such as cars can in turn be represented as a set of primitives.

Moreover, in this thesis, we did not apply any generalisation algorithm to edges and unconnected point of our simplicial complexes. Future work may look for designing an algorithm, similar to ℓ_0-plane pursuit, but operating on sets of edges. This algorithm would iteratively segment sets of edges into sets of linear or cylindrical regions. The final representation of such sets of edges will be a set of polylines and cylinders. This

will be consistent with the fact that most thin objects, for which we reconstructed only edges, are in fact thin cylinders (poles, small branches, wires).

Lastly, it is hard to find the underlying 3D shape of a set of unconnected points. Since we chose to let single points unconnected in the final reconstruction, we only know that their local neighborhood presents high geometrical variations. However, we noticed that when a large number of points are left unconnected in a small area, this usually correspond to the presence of vegetation in the scan. To reconstruct and visualise this vegetation, one may reconstruct a freeform surface, based on the global shape of this set of points. For instance, Lafarge and Alliez (2013) propose an algorithm combining mesh surfaces detected via a planar-based segmentation approach, and freeform surfaces for non-planar areas.

7.2.3 LOD-3 City-Scale Reconstruction

In this thesis, simplicial complexes are used to reconstruct urban scenes while preserving the geometric details of the scene. This could be used as input for further reconstructions as proposed in Verdie, Lafarge, and Alliez (2015). However, their method operates on 3D meshes covering entire scenes. As simplicial complexes are composed of points, edges and triangles, we think that simplices of each dimension can be generalized using different methods. Also, as in Verdie, Lafarge, and Alliez (2015), we think that a point-wise or simplex-wise semantic classification of the entire scene would be helpful to guide the reconstruction process. In fact, this classification could be used to cluster simplices in a similar manner as the work presented in Chapter 6. Then, for each cluster, a dedicated reconstruction method, taking into account all the geometric information of the cluster would be applied. Note that every step of this reconstruction process should include a data fitting prior, based on the local geometry of the scene, in order to preserve the geometric quality of the reconstruction. We argue that such method, combined with the high geometric precision of our reconstructed simplicial complexes could lead to a geometrically detailed modelisation of an entire city.

One of the first problems that needs to be solved is the missing geometry due to occlusion. A hole-filling step as in Biasutti et al. (2019) would be helpful to overcome missing data. However, we think that, in our case, classic inpainting techniques may not be able to recover all missing data, as occlusions can block large part of the scene. Hence, one could consider learning-based methods, such as the use of NN to complete missing data in urban scenes (Liu, He, and Salzmann, 2016). For instance, Dai, Diller, and Nießner (2019) propose to train a NN to fill incomplete 3D scenes by hiding parts of a partially occluded dataset and evaluate the network performances for reconstructing the geometry of the hidden areas. Their network is self-supervised, and even if it does not produce watertight reconstructions, it is able to reconstruct the geometry in unseen areas with low volumetric error.

The possibility of reconstructing the full geometry from a LiDAR scan, while preserving a high LoD (for instance, LoD-3 for buildings), could simplify the production of 3D city models, especially for urban simulations (Piepereit et al., 2019) and 3D areas in films or video games (Gabellone et al., 2017). Such urban models usually require a large amount of data from various sources, and long fusion and generalization steps. Hence, the ability to reconstruct 3D cities with a high geometrical precision, and based on a single acquisition, and whose missing parts are completed by a dedicated algorithm, would save both time and computations.

Last, our data showed that simplicial complexes could also be used on scans of building interiors thanks to openings, such as windows. Indeed, our scans usually

include part of floors and roofs inside buildings. Here as well, learning-based methods would be useful to fill missing data and help the joint interior and exterior reconstruction simultaneously, and from a single LiDAR acquisition. Such reconstruction for buildings could be classified as a LoD-4 reconstruction. At the moment of writing this thesis, few studies exists that combine geometrically detailed interior and exterior reconstruction, and they usually need several acquisitions to be performed and co-registered (Yilmaz and Buyuksalih, 2015; Buyuksalih et al., 2019).

7.2.4 Real Time Analysis

The reconstruction of simplicial complexes, as defined in Section 3.2.2.2, is highly local and requires no global knowledge of the scene. Hence, we think that this reconstruction could be perfomed in real time with a mobile mapping platform. This opens new perspectives to real-time monitoring and autonomous driving.

Simplicial complexes are a great tool for detecting vegetation and thin objects such as electric wires. This makes simplicial complexes suitable for representing railroad infrastructures (Arastounia, 2015). Being able to reconstruct the 3D environment around rail tracks, and labeling surrounding point clouds is highly valuable for improving maintenance processes. In fact, if many trains were equipped with a LiDAR sensor, real-time reconstructions could be performed while the train is running and railway companies would be able to monitor the evolution of the whole network in real time.

The ability to perform a simplicial complex-based reconstruction in real time opens many applications. In fact, when time is an issue (for instance, with environmental hazards, such as earthquakes), it would be benefical to perform a fast acquisition of the impacted area with an ALS or a UAV-borne LiDAR leading to the fast reconstruction of the scene while maintaining precise geometrical informations, even on small objects (Olsen and Kayen, 2013). In fact, simplicial complexes would also help the identification of thin objects by modeling such objects as sets of lines. This would make it possible to quickly assess damages and detect damaged critical objects, such as electric wires.

Real-time simplicial complex reconstruction can also be used in the context of autonomous driving. In fact, real-time object identification based on LiDAR data is still a current challenge in the remote sensing community (Wu et al., 2018; Wang et al., 2018) and we think that strong geometric hints provided by simplicial complexes could help the identification of the geometrically complex objects such as street furniture or moving pedestrians and cyclists.

Appendices

A Creating a Classification Ground Truth with Cloud Compare

A.1 Cloud Compare

CloudCompare (Girardeau-Montaut, 2015) is an open-source software for visualizing and processing 3D point clouds and meshes. This software is able to read and process multiple file formats, including but not limited to: ASCII clouds, PLY, LAS, OBJ and OFF. CloudCompare is able to perform measurments between point clouds, apply 3D transformations, registrate, segment and compute statistics on 3D point clouds and meshes. Also, the software contains some additional plug-ins allowing for Shape extraction (Schnabel, Wahl, and Klein, 2007), Poisson reconstruction (Kazhdan and Hoppe, 2013), hidden points removal (Katz, Tal, and Basri, 2007) and so on.

In this appendix, we present the approach used to create the manual classification of the Paris dataset as presented in Section 6.5.1, using the CloudCompare software.

A.2 Point Cloud Labeling

The approach we present for labeling 3D point clouds in CloudCompare is based on three main steps:

1. manual segmentation of the input cloud
2. labeling of each segment
3. merge of all the segments

A.2.1 Manual Segmentation of the point cloud

From the input point cloud visible in Figure A.1, a manual segmentation can be performed using the segmentation tool of CloudCompare. This tool allow the user to create a polyline by sucessives left-clicks. The polyline is closed with a right click. CloudCompare propose to divide the selected cloud in two different clouds. The first one contains all the points inside the 3D extent of the polyline and the second cloud contains the remaining points. This is done by clicking on the *segment in* option. An illustration of this step is shown on Figure A.2.

A.2.2 Labeling of the Segments

Once the whole point cloud has been manually segmented, we have to label each segment. To this end, we select a segment and manually add the label, as a new scalar

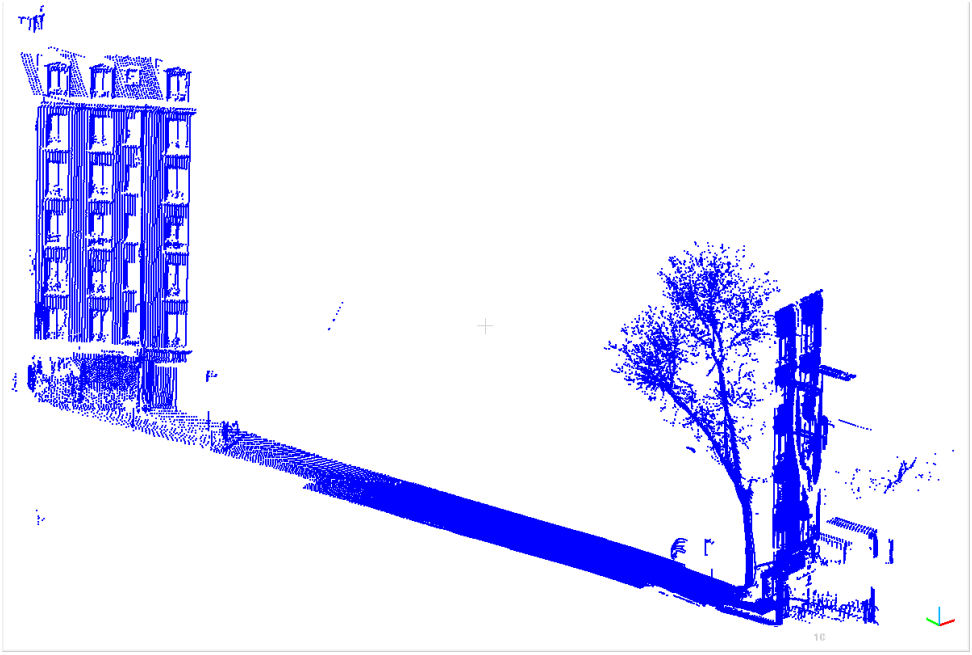


Figure A.1 – Input point cloud loaded in CloudCompare.

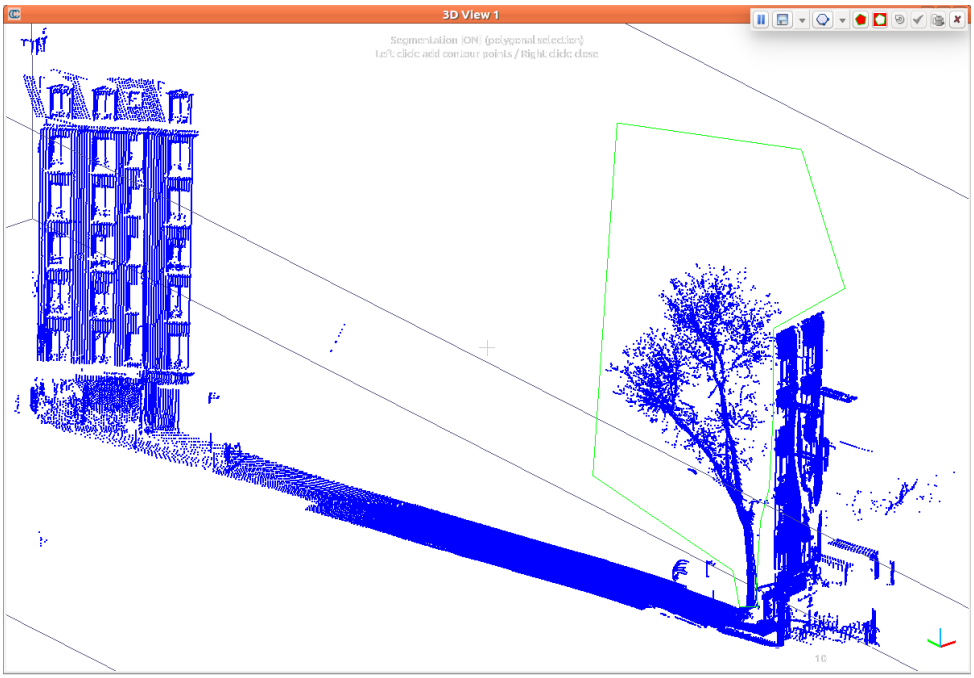


Figure A.2 – Segmenting a point cloud by drawing a polyline (in green).

field, for the segment. This label is set by clicking on the *Add Constant SF* option. Then, a first window appears, in which we set the name of the new scalar field. This name has to be excalty the same for all segments. Next, the software asks for a scalar value. We set the value according to the class to which the points of the segment belong. This process has to be done independently for each segment.

A.2.3 Merge of the Segments

The last step is simply done that selecting all the segments and using the *merge* tool of CloudCompare. An illustration of the merged labeled segments is presented in

Figure A.3.

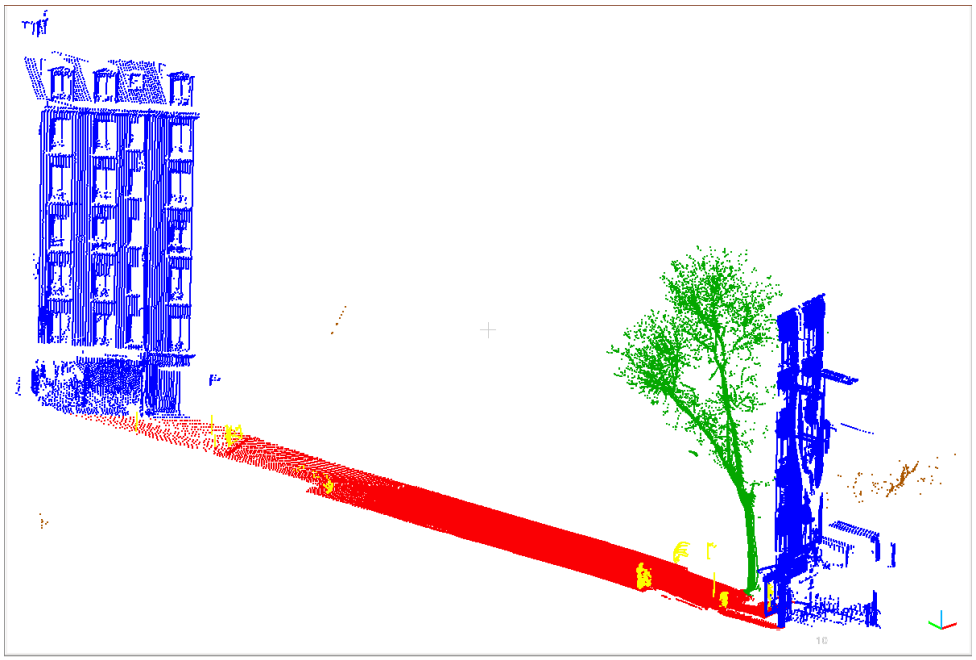


Figure A.3 – Final labeled point cloud. The semantic classes are represented with the following color code: buildings, road, vegetation, hardscape and other.

B List of Publications

Here is the list of publications written during this PhD:

B.1 Publications Related to this Thesis

B.1.1 International Peer-Reviewed Conferences

— Stéphane Guinard and Loïc Landrieu (2017). "Weakly Supervised Segmentation-Aided Classification of Urban Scenes from 3D LiDAR Point Clouds." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42

— Stéphane Guinard and Bruno Vallet (2018a). "Sensor-topology based simplicial complex reconstruction from mobile laser scanning." In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 4.2

— Stéphane Guinard et al. (2019). "Piecewise-planar approximation of large 3D data as graph structured optimization". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W5, pp. 365–372

B.1.2 National Peer-Reviewed Conferences

— Stéphane Guinard, Loïc Landrieu, and Bruno Vallet (June 2017). "Pré-segmentation pour la classification faiblement supervisée de scènes urbaines à partir de nuages de points 3D LIDAR". in: *ORASIS 2017*. GREYC. Colleville-sur-Mer, France *(in French)*

— Stéphane Guinard and Bruno Vallet (2018b). "Weighted simplicial complex reconstruction from mobile laser scanning using sensor topology". In: *Revue Française de Photogrammétrie et de Télédétection n* 217.218, p. 218

B.2 Other Publications

— Stéphane Guinard et al. (2020). "Planar Polygons Detection in LiDAR Scans Based on Sensor Topology Enhanced RANSAC". in: *Accepted to XXIVth ISPRS Congress*

Bibliography

Abdelmonem, Mohamed Gamal (2017). "Navigating virtual heritage applications for historic cities in the middle east". In: *2017 23rd International Conference on Virtual System & Multimedia (VSMM)*. IEEE, pp. 1–8 (cit. on p. 87).

Adam, Katrina, Victoria Hoolohan, James Gooding, Thomas Knowland, Catherine SE Bale, and Alison S Tomlin (2016). "Methodologies for city-scale assessment of renewable energy generation potential to inform strategic energy infrastructure investment". In: *Cities* 54, pp. 45–56 (cit. on p. 87).

Aglyamov, Yury, Dustin M Schroeder, and Steven D Vance (2017). "Bright prospects for radar detection of Europa's ocean". In: *Icarus* 281, pp. 334–337 (cit. on p. 66).

Ahmed, Mahmuda and Carola Wenk (2012). "Constructing street networks from GPS trajectories". In: *European Symposium on Algorithms*. Springer, pp. 60–71 (cit. on p. 93).

Akkiraju, Nataraj, Herbert Edelsbrunner, Michael Facello, Ping Fu, EP Mucke, and Carlos Varela (1995). "Alpha shapes: definition and software". In: *Proceedings of the 1st International Computational Geometry Software Workshop*. Vol. 63, p. 66 (cit. on pp. 53, 150, 172).

Alamús, R, F Pérez, L Pipia, and J Corbera (2018). "URBAN SUSTAINABLE ECOSYSTEMS ASSESSMENT THROUGH AIRBORNE EARTH OBSERVATION: LESSONS LEARNED." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42.1 (cit. on p. 83).

Alliez, Pierre, David Cohen-Steiner, and Lingjie Zhu (2019). "Triangulated Surface Mesh Approximation". In: *CGAL User and Reference Manual*. 5.0. CGAL Editorial Board (cit. on pp. 130, 160).

Alonzo, Michael, Bodo Bookhagen, and Dar A Roberts (2014). "Urban tree species mapping using hyperspectral and lidar data fusion". In: *Remote Sensing of Environment* 148, pp. 70–83 (cit. on p. 88).

Alonzo, Michael, Joseph P McFadden, David J Nowak, and Dar A Roberts (2016). "Mapping urban forest structure and function using hyperspectral imagery and lidar data". In: *Urban forestry & urban greening* 17, pp. 135–147 (cit. on p. 88).

Althausen, Dietrich, Detlef Müller, Albert Ansmann, Ulla Wandinger, Helgard Hube, Ernst Clauder, and Steffen Zörner (2000). "Scanning 6-wavelength 11-channel aerosol lidar". In: *Journal of Atmospheric and Oceanic Technology* 17.11, pp. 1469–1482 (cit. on p. 81).

Anguelov, Dragomir, B Taskarf, Vassil Chatalbashev, Daphne Koller, Dinkar Gupta, Geremy Heitz, and Andrew Ng (2005). "Discriminative learning of markov random fields for segmentation of 3d scan data". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2. IEEE, pp. 169–176 (cit. on pp. 168, 170).

Arastounia, Mostafa (2015). "Automated recognition of railroad infrastructure in rural areas from LiDAR data". In: *Remote Sensing* 7.11, pp. 14916–14938 (cit. on pp. 58, 193).

Arefi, Hossein (2009). "From LiDAR point clouds to 3D building models". PhD thesis (cit. on p. 70).

Armeni, Iro, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese (2016). "3d semantic parsing of large-scale indoor spaces". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1534–1543 (cit. on p. 82).

Askne, Jan IH, Maciej J Soja, and Lars MH Ulander (2017). "Biomass estimation in a boreal forest from TanDEM-X data, lidar DTM, and the interferometric water cloud model". In: *Remote Sensing of Environment* 196, pp. 265–278 (cit. on p. 86).

Asvadi, Alireza, Cristiano Premebida, Paulo Peixoto, and Urbano Nunes (2016). "3D Lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes". In: *Robotics and Autonomous Systems* 83, pp. 299–311 (cit. on p. 118).

Asvadi, Alireza, Luis Garrote, Cristiano Premebida, Paulo Peixoto, and Urbano J Nunes (2017). "Real-time deep convnet-based vehicle detection using 3d-lidar reflection intensity data". In: *Iberian Robotics conference*. Springer, pp. 475–486 (cit. on p. 88).

Attene, Marco and Giuseppe Patanè (2010). "Hierarchical structure recovery of point-sampled surfaces". In: *Computer Graphics Forum*. Vol. 29. 6. Wiley Online Library, pp. 1905–1920 (cit. on p. 120).

Au, Oscar Kin-Chung, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee (2008). "Skeleton extraction by mesh contraction". In: *ACM transactions on graphics (TOG)* 27.3, pp. 1–10 (cit. on p. 155).

Aubry, Mathieu, Bryan C Russell, and Josef Sivic (2014). "Painting-to-3D model alignment via discriminative visual elements". In: *ACM Transactions on Graphics (ToG)* 33.2, p. 14 (cit. on p. 94).

Aull, Brian F, Andrew H Loomis, Douglas J Young, Alvin Stern, Bradley J Felton, Peter J Daniels, Debbie J Landers, Larry Retherford, Dennis D Rathman, Richard M Heinrichs, et al. (2004). "Three-dimensional imaging with arrays of Geiger-mode avalanche photodiodes". In: *Semiconductor Photodetectors*. Vol. 5353. International Society for Optics and Photonics, pp. 105–116 (cit. on p. 82).

Azadbakht, Mohsen, Clive S Fraser, and Kourosh Khoshelham (2018). "Synergy of sampling techniques and ensemble classifiers for classification of urban environments using full-waveform LiDAR data". In: *International journal of applied earth observation and geoinformation* 73, pp. 277–291 (cit. on p. 81).

Azadbakht, Mohsen, Clive S Fraser, and Chunsun Zhang (2015). "Separability of targets in urban areas using features from full-waveform LiDARA data". In: *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, pp. 5367–5370 (cit. on p. 81).

Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla (2017). "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12, pp. 2481–2495 (cit. on p. 63).

Baker, Wayman E, George D Emmitt, Franklin Robertson, Robert M Atlas, John E Molinari, David A Bowdle, Jan Paegle, R Michael Hardesty, Robert T Menzies, TN Krishnamurti, et al. (1995). "Lidar-measured winds from space: a key component for weather and climate prediction". In: *Bulletin of the American Meteorological Society* 76.6, pp. 869–888 (cit. on p. 86).

Balsa-Barreiro, José and Dieter Fritsch (2018). "Generation of visually aesthetic and detailed 3D models of historical cities by using laser scanning and digital photogrammetry". In: *Digital applications in archaeology and cultural heritage* 8, pp. 57–64 (cit. on p. 62).

Barmak, Jonathan Ariel and Elias Gabriel Minian (2012). "Strong homotopy types, nerves and collapses". In: *Discrete & Computational Geometry* 47.2, pp. 301–328 (cit. on p. 155).

Barrett, Earl W and Oded Ben-Dov (1967). "Application of the lidar to air pollution measurements". In: *Journal of Applied Meteorology* 6.3, pp. 500–515 (cit. on p. 76).

Barrick, Donald E, MW Evans, and BL Weber (1977). "Ocean surface currents mapped by radar". In: *Science* 198.4313, pp. 138–144 (cit. on p. 66).

Barsan, Ioan Andrei, Shenlong Wang, Andrei Pokrovsky, and Raquel Urtasun (2018). "Learning to Localize Using a LiDAR Intensity Map." In: *CoRL*, pp. 605–616 (cit. on p. 176).

Basov, Nikolai G and AM Prokhorov (1954). "Application of molecular beams to the radio spectroscopic study of the rotation spectra of molecules". In: *Zh Eksp Theo Fiz* 27, p. 431 (cit. on p. 76).

Bauchet, Jean-Philippe and Florent Lafarge (2019). "City reconstruction from airborne LiDAR: a computational geometry approach". In: (cit. on p. 70).

Bauwens, Sébastien, Harm Bartholomeus, Kim Calders, and Philippe Lejeune (2016). "Forest inventory with terrestrial LiDAR: A comparison of static and hand-held mobile laser scanning". In: *Forests* 7.6, p. 127 (cit. on p. 86).

Becker, Susanne and Norbert Haala (2009). "Grammar supported facade reconstruction from mobile lidar mapping". In: *ISPRS Workshop, CMRT09-City Models, Roads and Traffic*. Vol. 38, p. 13 (cit. on pp. 70, 83, 116).

Behncke, Boris, Alessandro Fornaciai, Marco Neri, Massimiliano Favalli, Gaetana Ganci, and Francesco Mazzarini (2016). "Lidar surveys reveal eruptive volumes and rates at Etna, 2007–2010". In: *Geophysical Research Letters* 43.9, pp. 4270–4278 (cit. on pp. 37, 86).

Behrenfeld, Michael J, Yongxiang Hu, Robert T O'Malley, Emmanuel S Boss, Chris A Hostetler, David A Siegel, Jorge L Sarmiento, Jennifer Schulien, Johnathan W Hair, Xiaomei Lu, et al. (2017). "Annual boom–bust cycles of polar phytoplankton biomass revealed by space-based lidar". In: *Nature Geoscience* 10.2, p. 118 (cit. on p. 86).

Beksi, William J and Nikolaos Papanikolopoulos (2016). "3D point cloud segmentation using topological persistence". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5046–5051 (cit. on pp. 72, 174).

Berger, Matthew, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva (2017). "A survey of surface reconstruction from point clouds". In: *Computer Graphics Forum*. Vol. 36. 1. Wiley Online Library, pp. 301–329 (cit. on p. 120).

Berman, Gennady P, Alan R Bishop, Boris M Chernobrod, Marilyn E Hawley, and Geoffrey W Brown (2006). "Sensitivity analysis of a proposed novel opto-nano-mechanical photodetector for improving the performance of LIDAR and local optical sensors". In: *Journal of Physics: Conference Series*. Vol. 38. 1. IOP Publishing, p. 171 (cit. on p. 77).

Bernard, C, JP Mills, J Talaya, and F Remondino (2019). "Investigation into the potential of single photon airborne laser scanning technology". In: *ISPRS Geospatial Week 2019* (cit. on p. 82).

Bernardini, Fausto and Chandrajit L Bajaj (1997). "Sampling and Reconstructing Manifolds Using Alpha-Shapes". In: *In Proc. 9th Canad. Conf. Comput. Geom*. Citeseer (cit. on p. 93).

Bertalmio, Marcelo, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester (2000). "Image inpainting". In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 417–424 (cit. on p. 69).

Biasutti, Pierre, Jean-François Aujol, Mathieu Brédif, and Aurélie Bugeau (2019). "Diffusion and inpainting of reflectance and height LiDAR orthoimages". In: *Computer Vision and Image Understanding* 179, pp. 31–40 (cit. on pp. 69, 192).

— (2017). "Disocclusion of 3D LiDAR point clouds using range images". In: (cit. on p. 83).

Bigdeli, Behnaz, Hamed Amini Amirkolaee, and Parham Pahlavani (2018). "DTM extraction under forest canopy using LiDAR data and a modified invasive weed optimization algorithm". In: *Remote sensing of environment* 216, pp. 289–300 (cit. on pp. 37, 86).

Biljecki, Filip, Hugo Ledoux, and Jantien Stoter (2016). "An improved LOD specification for 3D building models". In: *Computers, Environment and Urban Systems* 59, pp. 25–37 (cit. on p. 68).

Blanchard, Yves (2016). "A French Pre-WW II attempt at air-warning radar: Pierre David's "electromagnetic barrier"". In: *URSI Radio Science Bulletin* 2016.358, pp. 18–34 (cit. on p. 65).

Blaugrund, AE (1966). "Notes on Doppler-shift lifetime measurements". In: *Nuclear Physics* 88.3, pp. 501–512 (cit. on p. 85).

Blomley, R and M Weinmann (2017). "USING MULTI-SCALE FEATURES FOR THE 3D SEMANTIC LABELING OF AIRBORNE LASER SCANNING DATA." In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 4 (cit. on p. 95).

Bock, Jérôme, Catherine Riond, Alain Munoz, Jean-Matthieu Monnet, and Frederic Berger (2017). "Airborn LIDAR: a new technology for mapping alpin complex forest stands". In: (cit. on p. 86).

Boerner, R, L Hoegner, and U Stilla (2017). "VOXEL BASED SEGMENTATION OF LARGE AIRBORNE TOPOBATHYMETRIC LIDAR DATA." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42 (cit. on p. 119).

Boissonat, Jean-Daniel (1984). "Representing 2D and 3D shapes with the Delaunay triangulation". In: *InSeventh International Conference on Pattern Recognition (ICPR'84)* (cit. on pp. 39, 95).

Boissonnat, Jean-Daniel (1984). "Geometric structures for three-dimensional shape representation". In: *ACM Transactions on Graphics (TOG)* 3.4, pp. 266–286 (cit. on p. 120).

Boissonnat, Jean-Daniel and Siddharth Pritam (2019). "Edge Collapse and Persistence of Flag Complexes". In: (cit. on p. 155).

Borouchaki, H and PJ Frey (2005). "Simplification of surface mesh using Hausdorff envelope". In: *Computer methods in applied mechanics and engineering* 194.48-49, pp. 4864–4884 (cit. on p. 159).

Bose, Supratik, Amin Nozari, Mohammad Ebrahim Mohammadi, Andreas Stavridis, Moaveni Babak, Richard Wood, Dan Gillins, and Andre Barbosa (2016). "Structural assessment of a school building in Sankhu, Nepal damaged due to torsional response during the 2015 Gorkha earthquake". In: *Dynamics of Civil Structures, Volume 2*. Springer, pp. 31–41 (cit. on p. 87).

Boulch, Alexandre, Martin de La Gorce, and Renaud Marlet (2014). "Piecewise-planar 3D reconstruction with edge and corner regularization". In: *Computer Graphics Forum*. Vol. 33. 5. Wiley Online Library, pp. 55–64 (cit. on p. 118).

Boulch, Alexandre, Bertrand Le Saux, and Nicolas Audebert (2017). "Unstructured Point Cloud Semantic Labeling Using Deep Segmentation Networks." In: *3DOR* 2, p. 7 (cit. on p. 94).

Boussaha, Mohamed, Eduardo Fernandez-Moral, Bruno Vallet, and Patrick Rives (2018). "On the production of semantic and textured 3d meshes of large scale urban environments from mobile mapping images and lidar scans". In: *Reconnaissance des Formes, Image, Apprentissage et Perception (RFIAP), Marne-la-Vallée, France* (cit. on p. 87).

Boykov, Yuri, Olga Veksler, and Ramin Zabih (2001). "Fast approximate energy minimization via graph cuts". In: *IEEE Transactions on pattern analysis and machine intelligence* 23.11, pp. 1222–1239 (cit. on pp. 124, 180).

Breiman, Leo (2001). "Random forests". In: *Machine learning* 45.1, pp. 5–32 (cit. on pp. 53, 81, 170, 182).

Brenner, Claus and Nora Ripperda (2006). "Extraction of facades using RJMCMC and constraint equations". In: *Photogrammetric Computer Vision* 36, pp. 155–160 (cit. on p. 117).

Bruhn, R, J McCalpin, T Pavlis, F Gutierrez, J Guerrero, and P Lucha (2006). "Active tectonics of western Saint Elias orogen, Alaska: Integration of LIDAR and field geology". In: *AGU Fall Meeting Abstracts* (cit. on p. 86).

Bruzzone, Lorenzo, Jeffrey J Plaut, Giovanni Alberti, Donald D Blankenship, Francesca Bovolo, Bruce A Campbell, Adamo Ferro, Yonggyu Gim, Wlodek Kofman, Goro Komatsu, et al. (2013). "RIME: Radar for icy moon exploration". In: *2013 IEEE International geoscience and remote sensing symposium-IGARSS*. IEEE, pp. 3907–3910 (cit. on p. 66).

Buckley, Simon J, JA Howell, HD Enge, and TH Kurz (2008). "Terrestrial laser scanning in geology: data acquisition, processing and accuracy considerations". In: *Journal of the Geological Society* 165.3, pp. 625–638 (cit. on p. 86).

Buyuksalih, G, P Baskaraca, S Bayburt, I Buyuksalih, and A Abdul Rahman (2019). "3D CITY MODELLING OF ISTANBUL BASED ON LIDAR DATA AND PANORAMIC IMAGES–ISSUES AND CHALLENGES". In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 42.4/W12 (cit. on p. 193).

Cacciola, Fernando (2019). "Triangulated Surface Mesh Simplification". In: *CGAL User and Reference Manual*. 5.0. CGAL Editorial Board (cit. on p. 155).

Caltagirone, Luca, Mauro Bellone, Lennart Svensson, and Mattias Wahde (2019). "LIDAR–camera fusion for road detection using fully convolutional neural networks". In: *Robotics and Autonomous Systems* 111, pp. 125–131 (cit. on p. 78).

— (2017). "LIDAR-based driving path generation using fully convolutional neural networks". In: *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 1–6 (cit. on p. 77).

Caraffa, Laurent, Mathieu Brédif, and Bruno Vallet (2016). "3D watertight mesh generation with uncertainties from ubiquitous data". In: *Asian Conference on Computer Vision*. Springer, pp. 377–391 (cit. on pp. 65, 69 sq.).

Cavallari, Tommaso, Stuart Golodetz, Nicholas Lord, Julien Valentin, Victor Prisacariu, Luigi Di Stefano, and Philip HS Torr (2019). "Real-time RGB-D camera pose estimation in novel scenes using a relocalisation cascade". In: *IEEE transactions on pattern analysis and machine intelligence* (cit. on p. 94).

Chambers, Erin W, Vin De Silva, Jeff Erickson, and Robert Ghrist (2010). "Vietoris–rips complexes of planar point sets". In: *Discrete & Computational Geometry* 44.1, pp. 75–90 (cit. on p. 72).

Chanin, ML, A Garnier, A Hauchecorne, and J Porteneuve (1989). "A Doppler lidar for measuring winds in the middle atmosphere". In: *Geophysical research letters* 16.11, pp. 1273–1276 (cit. on p. 85).

Chase, Adrian SZ, Diane Z Chase, and Arlen F Chase (2017). "LiDAR for archaeological research and the study of historical landscapes". In: *Sensing the Past*. Springer, pp. 89–100 (cit. on p. 87).

Chauve, Adrien, Clément Mallet, Frédéric Bretar, Sylvie Durrieu, Marc Pierrot Deseilligny, and William Puech (2008). "Processing full-waveform lidar data: modelling raw signals". In: *International archives of photogrammetry, remote sensing and spatial information sciences 2007*, pp. 102–107 (cit. on p. 79).

Chauve, Anne-Laure, Patrick Labatut, and Jean-Philippe Pons (2010). "Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data". In: *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, pp. 1261–1268 (cit. on pp. 113, 118).

Chazal, Frédéric, Leonidas J Guibas, Steve Y Oudot, and Primoz Skraba (2009). "Analysis of scalar fields over point cloud data". In: *Proceedings of the twentieth annual ACM-SIAM symposium on Discrete algorithms*. SIAM, pp. 1021–1030 (cit. on p. 72).

Chen, Biwu, Shuo Shi, Wei Gong, Qingjun Zhang, Jian Yang, Lin Du, Jia Sun, Zhenbing Zhang, and Shalei Song (2017). "Multispectral LiDAR point cloud classification: A two-step approach". In: *Remote Sensing* 9.4, p. 373 (cit. on p. 169).

Chen, Hua-hong, Xiao-nan Luo, and Ruo-tian Ling (2007). "Surface Simplification Using multi-edge mesh collapse". In: *Fourth International Conference on Image and Graphics (ICIG 2007)*. IEEE, pp. 954–959 (cit. on p. 155).

Chen, Jingdao and Yong K Cho (2016). "Real-time 3D mobile mapping for the built environment". In: *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*. Vol. 33. Vilnius Gediminas Technical University, Department of Construction Economics …, p. 1 (cit. on p. 83).

Chen, Liang-Chieh, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam (2018). "Encoder-decoder with atrous separable convolution for semantic image segmentation". In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 801–818 (cit. on p. 63).

Chen, VC and M Martorella (2014). "Inverse synthetic aperture radar". In: *Scitech Publishing 2014, and references therein* (cit. on p. 65).

Cheng, Liang, Jianya Gong, Manchun Li, and Yongxue Liu (2011). "3D building model reconstruction from multi-view aerial imagery and lidar data". In: *Photogrammetric Engineering & Remote Sensing* 77.2, pp. 125–139 (cit. on p. 68).

Cheng, Liang, Yajun Wang, Yanming Chen, and Manchun Li (2016). "Using LiDAR for digital documentation of ancient city walls". In: *Journal of Cultural Heritage* 17, pp. 188–193 (cit. on p. 87).

Cheng, Yizong (1995). "Mean shift, mode seeking, and clustering". In: *IEEE transactions on pattern analysis and machine intelligence* 17.8, pp. 790–799 (cit. on p. 119).

Chew, L Paul (1989). "Constrained delaunay triangulations". In: *Algorithmica* 4.1-4, pp. 97–108 (cit. on p. 160).

Chisholm, Ryan A, Jinqiang Cui, Shawn KY Lum, and Ben M Chen (2013). "UAV LiDAR for below-canopy forest surveys". In: *Journal of Unmanned Vehicle Systems* 1.01, pp. 61–68 (cit. on p. 84).

Cho, Hyunggi, Young-Woo Seo, BVK Vijaya Kumar, and Ragunathan Raj Rajkumar (2014). "A multi-sensor fusion system for moving object detection and tracking in urban driving environments". In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1836–1843 (cit. on p. 78).

Chung, Minkyung, Changjae Kim, Kanghyeok Choi, DongKi Chung, and Yongil Kim (2017). "Development of LiDAR simulator for backpack-mounted mobile indoor mapping system". In: *Korean Journal of Geomatics* 35.2, pp. 91–101 (cit. on p. 82).

Chust, Guillem, Maitane Grande, Ibon Galparsoro, Adolfo Uriarte, and Ángel Borja (2010). "Capabilities of the bathymetric Hawk Eye LiDAR for coastal habitat mapping: A case study within a Basque estuary". In: *Estuarine, Coastal and Shelf Science* 89.3, pp. 200–213 (cit. on p. 143).

Cignoni, Paolo, Claudio Montani, and Roberto Scopigno (1998). "A comparison of mesh simplification algorithms". In: *Computers & Graphics* 22.1, pp. 37–54 (cit. on p. 160).

Cignoni, Paolo, Claudio Rocchini, and Roberto Scopigno (1998). "Metro: measuring error on simplified surfaces". In: *Computer graphics forum*. Vol. 17. 2. Wiley Online Library, pp. 167–174 (cit. on p. 159).

Cignoni, Paolo, D Costanza, Claudio Montani, Claudio Rocchini, and Roberto Scopigno (2000). "Simplification of tetrahedral meshes with accurate error evaluation". In: *Proceedings Visualization 2000. VIS 2000 (Cat. No. 00CH37145)*. IEEE, pp. 85–92 (cit. on p. 152).

Clifton, William E, Bradley Steele, Graham Nelson, Antony Truscott, Mark Itzler, and Mark Entwistle (2015). "Medium altitude airborne Geiger-mode mapping LIDAR system". In: *Laser Radar Technology and Applications XX; and Atmospheric Propagation XII*. Vol. 9465. International Society for Optics and Photonics, p. 946506 (cit. on p. 82).

Cohen-Steiner, David, Pierre Alliez, and Mathieu Desbrun (2004). "Variational shape approximation". In: *ACM Transactions on Graphics (ToG)*. Vol. 23. 3. ACM, pp. 905–914 (cit. on pp. 47, 51, 119, 128, 130, 150, 159 sq., 165).

Collin, Antoine, Camille Ramambason, Yves Pastol, Elisa Casella, Alessio Rovere, Lauric Thiault, Benoît Espiau, Gilles Siu, Franck Lerouvreur, Nao Nakamura, et al. (2018). "Very high resolution mapping of coral reef state using airborne bathymetric LiDAR surface-intensity and drone imagery". In: *International journal of remote sensing* 39.17, pp. 5676–5688 (cit. on p. 84).

Collis, RTH (1969). "Lidar". In: *Advances in Geophysics*. Vol. 13. Elsevier, pp. 113–139 (cit. on p. 76).

Colton, Roger B (1945). "Radar in the United States Army History and Early Development at the Signal Corps Laboratories, Fort Monmouth, NJ". In: *Proceedings of the IRE* 33.11, pp. 740–753 (cit. on p. 65).

Crisp, David J (2004). *The state-of-the-art in ship detection in synthetic aperture radar imagery*. Tech. rep. Defence Science And Technology Organisation Salisbury (Australia) Info … (cit. on p. 66).

Csatho, bea, Toni Schenk, Philip Kyle, Terry Wilson, and William B Krabill (2008). "Airborne laser swath mapping of the summit of Erebus volcano, Antarctica: applications to geological mapping of a volcano". In: *Journal of Volcanology and Geothermal Research* 177.3, pp. 531–548 (cit. on p. 86).

Curme, Henry and Royden N Rand (1997). "Early history of Eastman Kodak Ektachem slides and instrumentation". In: *Clinical Chemistry* 43.9, pp. 1647–1652 (cit. on p. 61).

Da, Tran Kai Frank (2019). "2D Alpha Shapes". In: *CGAL User and Reference Manual*. 5.0. CGAL Editorial Board (cit. on p. 151).

Dai, Angela, Christian Diller, and Matthias Nießner (2019). "SG-NN: Sparse Generative Neural Networks for Self-Supervised Scene Completion of RGB-D Scans". In: *arXiv preprint arXiv:1912.00036* (cit. on pp. 58, 192).

Dai, Wenxia, Bisheng Yang, Zhen Dong, and Ahmed Shaker (2018). "A new method for 3D individual tree extraction using multispectral airborne LiDAR point clouds". In: *ISPRS journal of photogrammetry and remote sensing* 144, pp. 400–411 (cit. on p. 119).

Davenport, Alma (1999). *The history of photography: an overview*. UNM Press (cit. on p. 61).

De Almeida, Danilo Roberti Alves, Bruce Walker Nelson, Juliana Schietti, Eric Bastos Gorgens, Angélica Faria Resende, Scott C Stark, and Rubén Valbuena (2016). "Contrasting fire damage and fire susceptibility between seasonally flooded forest and upland forest in the Central Amazon using portable profiling LiDAR". In: *Remote Sensing of Environment* 184, pp. 153–160 (cit. on pp. 37, 86).

De Goes, Fernando, David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun (2011). "An optimal transport approach to robust reconstruction and simplification of 2D shapes". In: *Computer Graphics Forum*. Vol. 30. 5. Wiley Online Library, pp. 1593–1602 (cit. on p. 93).

Degnan, John (2016). "Scanning, multibeam, single photon lidars for rapid, large scale, high resolution, topographic and bathymetric mapping". In: *Remote Sensing* 8.11, p. 958 (cit. on p. 82).

Degnan, John, David Wells, Roman Machan, and Ed Leventhal (2007). "Second generation airborne 3D imaging lidars based on photon counting". In: *Advanced photon counting techniques II*. Vol. 6771. International Society for Optics and Photonics, 67710N (cit. on p. 82).

Dehbi, Youness, Fabian Hadiji, Gerhard Gröger, Kristian Kersting, and Lutz Plümer (2017). "Statistical relational learning of grammar rules for 3D building reconstruction". In: *Transactions in GIS* 21.1, pp. 134–150 (cit. on p. 117).

Delaunay, Boris et al. (1934). "Sur la sphere vide". In: *Izv. Akad. Nauk SSSR, Otdelenie Matematicheskii i Estestvennyka Nauk* 7.793-800, pp. 1–2 (cit. on p. 95).

DeLong, Stephen B, Ann M Youberg, Whitney M DeLong, and Brendan P Murphy (2018). "Post-wildfire landscape change and erosional processes from repeat terrestrial lidar in a steep headwater catchment, Chiricahua Mountains, Arizona, USA". In: *Geomorphology* 300, pp. 13–30 (cit. on pp. 37, 86).

Demantke, Jerome, Clément Mallet, Nicolas David, and Bruno Vallet (2011). "Dimensionality based scale selection in 3D lidar point clouds". In: *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 38.Part 5, W12 (cit. on pp. 95, 118, 142, 171, 191).

Demir, Ilke, Daniel G Aliaga, and Bedrich Benes (2016). "Proceduralization for editing 3d architectural models". In: *2016 Fourth International Conference on 3D Vision (3DV)*. IEEE, pp. 194–202 (cit. on p. 117).

Devillers, Olivier, Samuel Hornus, and Clément Jamin (2019). "dD Triangulations". In: *CGAL User and Reference Manual*. 4.14. CGAL Editorial Board (cit. on pp. 128 sq.).

Dey, Tamal K, Jiayuan Wang, and Yusu Wang (2017). "Improved road network reconstruction using discrete morse theory". In: *Proceedings of the 25th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. ACM, p. 58 (cit. on p. 93).

Dick, Anthony R, Philip HS Torr, and Roberto Cipolla (2002). "A Bayesian estimation of building shape using MCMC". In: *European Conference on Computer Vision*. Springer, pp. 852–866 (cit. on p. 117).

Digne, Julie, David Cohen-Steiner, Pierre Alliez, Fernando De Goes, and Mathieu Desbrun (2014). "Feature-preserving surface reconstruction and simplification from defect-laden point sets". In: *Journal of mathematical imaging and vision* 48.2, pp. 369–382 (cit. on p. 93).

Dogon-Yaro, MA, P Kumar, A Abdul Rahman, and G Buyuksalih (2016). "SEMI-AUTOMATED APPROACH FOR MAPPING URBAN TREES FROM INTEGRATED AERIAL LiDAR POINT CLOUD AND DIGITAL IMAGERY DATASETS." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42 (cit. on p. 88).

Dohan, David, Brian Matejek, and Thomas Funkhouser (2015). "Learning hierarchical semantic segmentations of LIDAR data". In: *2015 International Conference on 3D Vision*. IEEE, pp. 273–281 (cit. on pp. 88, 170).

Dokter, Adriaan M, Felix Liechti, Herbert Stark, Laurent Delobbe, Pierre Tabary, and Iwan Holleman (2011). "Bird migration flight altitudes studied by a network of operational weather radars". In: *Journal of the Royal Society Interface* 8.54, pp. 30–43 (cit. on p. 66).

Dong, Hang, Sean Anderson, and Timothy D Barfoot (2013). "Two-axis scanning lidar geometric calibration using intensity imagery and distortion mapping". In: *2013 IEEE International Conference on Robotics and Automation*. IEEE, pp. 3672–3678 (cit. on p. 77).

Dore, Conor and Maurice Murphy (2017). "Current state of the art historic building information modelling". In: (cit. on p. 116).

Doria, David and Richard J Radke (2012). "Filling large holes in lidar data by inpainting depth gradients". In: *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. IEEE, pp. 65–72 (cit. on p. 69).

Dorninger, Peter and Norbert Pfeifer (2008). "A comprehensive automated 3D approach for building extraction, reconstruction, and regularization from airborne laser scanning point clouds". In: *Sensors* 8.11, pp. 7323–7343 (cit. on p. 88).

Douglas, David H and Thomas K Peucker (1973). "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature". In: *Cartographica: the international journal for geographic information and geovisualization* 10.2, pp. 112–122 (cit. on p. 147).

Douillard, Bertrand, James Underwood, Noah Kuntz, Vsevolod Vlaskine, Alastair Quadros, Peter Morton, and Alon Frenkel (2011). "On the segmentation of 3D LIDAR point clouds". In: *2011 IEEE International Conference on Robotics and Automation*. IEEE, pp. 2798–2805 (cit. on p. 119).

Du, Qiang and Desheng Wang (2006). "Recent progress in robust and quality Delaunay mesh generation". In: *Journal of Computational and Applied Mathematics* 195.1-2, pp. 8–23 (cit. on p. 96).

Dunham, Lisa, Joseph Wartman, Michael J Olsen, Matthew O'Banion, and Keith Cunningham (2017). "Rockfall Activity Index (RAI): A lidar-derived, morphology-based method for hazard assessment". In: *Engineering geology* 221, pp. 184–192 (cit. on p. 87).

Dupré, Sven (2008). "Inside the camera obscura: Kepler's experiment and theory of optical imagery". In: *Early Science and Medicine* 13.3, pp. 219–244 (cit. on p. 61).

Durupt, Mélanie and Franck Taillandier (2006). "Automatic building reconstruction from a digital elevation model and cadastral data: an operational approach". In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 36.3, pp. 142–147 (cit. on p. 120).

Dutta, Avishek, Johannes Engels, and Michael Hahn (2018). "Segmentation of Laser Point Clouds in Urban Areas by a Modified Normalized Cut Method". In: *IEEE transactions on pattern analysis and machine intelligence* (cit. on p. 120).

Dzitsiuk, Maksym, Jürgen Sturm, Robert Maier, Lingni Ma, and Daniel Cremers (2017). "De-noising, stabilizing and completing 3D reconstructions on-the-go using plane

priors". In: *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, pp. 3976–3983 (cit. on pp. 117 sq.).

Edelsbrunner, Herbert, David Kirkpatrick, and Raimund Seidel (1983). "On the shape of a set of points in the plane". In: *IEEE Transactions on information theory* 29.4, pp. 551–559 (cit. on p. 150).

Einstein, Albert (1917). "Zur quantentheorie der strahlung". In: *Phys. Z.* 18, pp. 121–128 (cit. on p. 76).

Emery, William and Adriano Camps (2017). *Introduction to satellite remote sensing: atmosphere, ocean, land and cryosphere applications*. Elsevier (cit. on p. 66).

Ennafii, Oussama, Clément Mallet, Arnaud Le Bris, and Florent Lafarge (2019). "The necessary yet complex evaluation of 3D city models: a semantic approach". In: *2019 Joint Urban Remote Sensing Event (JURSE)*. IEEE, pp. 1–4 (cit. on p. 120).

Esaias, Wayne E (1980). "Remote sensing of oceanic phytoplankton: Present capabilities and future goals". In: *Primary productivity in the sea*. Springer, pp. 321–337 (cit. on p. 66).

Fang, Hao, Florent Lafarge, and Mathieu Desbrun (2018). "Planar Shape Detection at Structural Scales". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 119).

Ferraz, António, Frédéric Bretar, Stéphane Jacquemoud, Gil Gonçalves, Luisa Pereira, Margarida Tomé, and Paula Soares (2012). "3-D mapping of a multi-layered Mediterranean forest using ALS data". In: *Remote Sensing of Environment* 121, pp. 210–223 (cit. on p. 119).

Fiocco, G and G Grams (1964). "Observations of the aerosol layer at 20 km by optical radar". In: *Journal of the Atmospheric Sciences* 21.3, pp. 323–324 (cit. on p. 76).

Fiocco, G and LD Smullin (1963). "Detection of scattering layers in the upper atmosphere (60–140 km) by optical radar". In: *Nature* 199.4900, pp. 1275–1276 (cit. on p. 76).

Fischer, A, B Seiser, M Stocker Waldhuber, C Mitterer, and J Abermann (2015). "Tracing glacier changes in Austria from the Little Ice Age to the present using a lidar-based high-resolution glacier inventory in Austria". In: *The Cryosphere* 9.2, pp. 753–766 (cit. on p. 86).

Fischler, Martin A and Robert C Bolles (1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6, pp. 381–395 (cit. on p. 118).

Fissore, F and F Pirotti (2019). "Dsm and DTM for Extracting 3d Building Models: Advantages and Limitations". In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 42.2/W13 (cit. on pp. 83, 88).

Franke, Uwe, Dariu Gavrila, Steffen Görzig, Frank Lindner, Frank Paetzold, and Christian Wöhler (1998). "Autonomous driving goes downtown". In: *IEEE Intelligent systems* 6, pp. 40–48 (cit. on p. 88).

Fukunaga, Keinosuke and Larry Hostetler (1975). "The estimation of the gradient of a density function, with applications in pattern recognition". In: *IEEE Transactions on information theory* 21.1, pp. 32–40 (cit. on p. 119).

Gabellone, Francesco, Antonio Lanorte, Nicola Masini, and Rosa Lasaponara (2017). "From remote sensing to a serious game: digital reconstruction of an abandoned medieval village in Southern Italy". In: *Journal of Cultural Heritage* 23, pp. 63–70 (cit. on pp. 58, 192).

Galati, Gaspare, Mauro Leonardi, Alessio Cavallin, and Gabriele Pavan (2010). "Airport surveillance processing chain for high resolution radar". In: *IEEE Transactions on Aerospace and Electronic Systems* 46.3, pp. 1522–1533 (cit. on p. 66).

Gao, Yanbin, Shifei Liu, Mohamed Atia, and Aboelmagd Noureldin (2015). "INS/GPS/LiDAR integrated navigation system for urban and indoor environments using hybrid scan matching algorithm". In: *Sensors* 15.9, pp. 23286–23302 (cit. on p. 77).

Gardner, Chester S, Alan Z Liu, DR Marsh, Wuhu Feng, and JMC Plane (2014). "Inferring the global cosmic dust influx to the Earth's atmosphere from lidar observations of the vertical flux of mesospheric Na". In: *Journal of Geophysical Research: Space Physics* 119.9, pp. 7870–7879 (cit. on p. 85).

Gargoum, Suliman, Karim El-Basyouny, Joseph Sabbagh, and Kenneth Froese (2017). "Automated highway sign extraction using lidar data". In: *Transportation research record* 2643.1, pp. 1–8 (cit. on p. 88).

Garland, Michael and Paul S Heckbert (1997). "Surface simplification using quadric error metrics". In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., pp. 209–216 (cit. on pp. 147, 150, 152 sq., 155, 159 sq.).

Geiger, Andreas, Philip Lenz, and Raquel Urtasun (2012). "Are we ready for autonomous driving? the kitti vision benchmark suite". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 3354–3361 (cit. on p. 78).

Gernsheim, Helmut (1986). *A concise history of photography*. 10. Courier Corporation (cit. on p. 61).

Ghallabi, Farouk, Fawzi Nashashibi, Ghayath El-Haj-Shhade, and Marie-Anne Mittet (2018). "LIDAR-Based Lane Marking Detection For Vehicle Positioning in an HD Map". In: *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 2209–2214 (cit. on pp. 38, 88).

Giannaka, Olga, Efi Dimopoulou, and Andreas Georgopoulos (2014). "Investigation on the contribution of LiDAR data in 3D cadastre". In: *Second International Conference on Remote Sensing and Geoinformation of the Environment (RSCy2014)*. Vol. 9229. International Society for Optics and Photonics, p. 922905 (cit. on p. 88).

Girardeau-Montaut, Daniel (2015). "Cloud compare—3d point cloud and mesh processing software". In: *Open Source Project* (cit. on p. 197).

Glennie, Craig, Benjamin Brooks, Todd Ericksen, Darren Hauser, Kenneth Hudnut, James Foster, and Jon Avery (2013). "Compact multipurpose mobile laser scanning system—Initial tests and results". In: *Remote Sensing* 5.2, pp. 521–538 (cit. on p. 83).

Goit, Jay Prakash, Susumu Shimada, and Tetsuya Kogaki (2019). "Can LiDARs Replace Meteorological Masts in Wind Energy?" In: *Energies* 12.19, p. 3680 (cit. on pp. 37, 85).

Goldstein, Richard M and HA Zebker (1987). "Interferometric radar measurement of ocean surface currents". In: *Nature* 328.6132, pp. 707–709 (cit. on p. 66).

Golias, NA and RW Dutton (1997). "Delaunay triangulation and 3D adaptive mesh generation". In: *Finite Elements in Analysis and Design* 25.3-4, pp. 331–341 (cit. on p. 96).

Golovinskiy, Aleksey, Vladimir G Kim, and Thomas Funkhouser (2009). "Shape-based recognition of 3D point clouds in urban environments". In: *2009 IEEE 12th International Conference on Computer Vision*. IEEE, pp. 2154–2161 (cit. on pp. 168, 170).

Golson, Steve (1993). "One-hot state machine design for FPGAs". In: *Proc. 3rd Annual PLD Design Conference & Exhibit*. Vol. 1. 3 (cit. on p. 176).

Gong, Xiaojin, Ying Lin, and Jilin Liu (2013). "3D LIDAR-camera extrinsic calibration using an arbitrary trihedron". In: *Sensors* 13.2, pp. 1902–1918 (cit. on p. 77).

Gooding, James, Rolf Crook, and Alison S Tomlin (2015). "Modelling of roof geometries from low-resolution LiDAR data for city-scale solar energy applications using

a neighbouring buildings method". In: *Applied Energy* 148, pp. 93–104 (cit. on pp. 38, 88).

Google. *View of the Mabillon street next to the corner of Lobineau street (Paris)*. `https://goo.gl/maps/FQTvA7oaiNm` [Accessed: 2018-01-09] (cit. on pp. 43, 108).

Gordon, James P, Herbert J Zeiger, and Charles H Townes (1954). "Molecular microwave oscillator and new hyperfine structure in the microwave spectrum of N H 3". In: *Physical Review* 95.1, p. 282 (cit. on p. 77).

Goyer, Guy G and Robert Watson (1963). "The laser and its application to meteorology". In: *Bulletin of the American Meteorological Society* 44.9, pp. 564–570 (cit. on p. 76).

Graber, Richard R and Sylvia Sue Hassler (1962). "The effectiveness of aircraft-type (APS) radar in detecting birds". In: *The Wilson Bulletin*, pp. 367–380 (cit. on p. 66).

Gressin, Adrien, Clément Mallet, Jérôme Demantké, and Nicolas David (2013). "Towards 3D lidar point cloud registration improvement using optimal neighborhood knowledge". In: *ISPRS journal of photogrammetry and remote sensing* 79, pp. 240–251 (cit. on p. 95).

Gromov, Mikhael (1987). "Hyperbolic groups". In: *Essays in group theory*. Springer, pp. 75–263 (cit. on p. 72).

Groueix, Thibault, Matthew Fisher, Vladimir G Kim, Bryan C Russell, and Mathieu Aubry (2018). "A papier-mâché approach to learning 3d surface generation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 216–224 (cit. on p. 63).

Grube, Nikolai, Kai Delvendahl, Nicolaus Seefeld, and Beniamino Volta (2012). "Under the Rule of the Snake Kings: Uxul in the 7th and 8th Centuries". In: *Estudios de cultura maya* 40, pp. 11–49 (cit. on p. 87).

Grund, Christian J, Robert M Banta, Joanne L George, James N Howell, Madison J Post, Ronald A Richter, and Ann M Weickmann (2001). "High-resolution Doppler lidar for boundary layer and cloud research". In: *Journal of Atmospheric and Oceanic Technology* 18.3, pp. 376–393 (cit. on p. 85).

Guélis, Thibault Vaillant de, Hélène Chepfer, Vincent Noel, Rodrigo Guzman, David M Winker, and Riwal Plougonven (2017). "Using space lidar observations to decompose longwave cloud radiative effect variations over the last decade". In: *Geophysical Research Letters* 44.23, pp. 11–994 (cit. on p. 85).

Guenther, G and H Mesick (1988). "Analysis of airborne lidar bathymetric waveforms". In: *Proc. of the 9th Ocean Optics. Orlando, FA, USA, SPIE*, pp. 4–6 (cit. on p. 79).

Guerlac, Henry and Marie Boas (1950). "The Radar War Against the U-Boat". In: *The Journal of Military History* 14, p. 99 (cit. on p. 66).

Guibas, Leonidas J and Steve Y Oudot (2008). "Reconstruction using witness complexes". In: *Discrete & computational geometry* 40.3, pp. 325–356 (cit. on p. 93).

Guilbert, Eric, Sylvain Jutras, and Thierry Badard (2018). "THALWEG DETECTION FOR RIVER NETWORK CARTOGRAPHY IN FOREST FROM HIGH-RESOLUTION LIDAR DATA." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* (cit. on p. 86).

Guinard, Stéphane. *Sensor-Topology based simplicial complex reconstruction from mobile laser scanning*. `https://youtu.be/zJn3YF8eer4` [Accessed: 2018-04-06] (cit. on p. 107).

Guinard, Stéphane and Loïc Landrieu (2017). "Weakly Supervised Segmentation-Aided Classification of Urban Scenes from 3D LiDAR Point Clouds." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42 (cit. on pp. 142, 180, 201).

Guinard, Stéphane, Loïc Landrieu, and Bruno Vallet (June 2017). "Pré-segmentation pour la classification faiblement supervisée de scènes urbaines à partir de nuages de points 3D LIDAR". In: *ORASIS 2017*. GREYC. Colleville-sur-Mer, France (cit. on p. 201).

Guinard, Stéphane and Bruno Vallet (2018a). "Sensor-topology based simplicial complex reconstruction from mobile laser scanning." In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 4.2 (cit. on pp. 47, 112, 130, 201).

— (2018b). "Weighted simplicial complex reconstruction from mobile laser scanning using sensor topology". In: *Revue Française de Photogrammétrie et de Télédétection n* 217.218, p. 218 (cit. on pp. 112, 201).

Guinard, Stéphane, Zoumana Mallé, Oussama Ennafii, Pascal Monasse, and Bruno Vallet (2020). "Planar Polygons Detection in LiDAR Scans Based on Sensor Topology Enhanced RANSAC". In: *Accepted to XXIVth ISPRS Congress* (cit. on p. 201).

Guinard, Stéphane, Loïc Landrieu, Laurent Caraffa, and Bruno Vallet (2019). "Piecewise-planar approximation of large 3D data as graph structured optimization". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* IV-2/W5, pp. 365–372 (cit. on p. 201).

Guo, Li, Nesrine Chehata, Clément Mallet, and Samia Boukir (2011). "Relevance of airborne lidar and multispectral image data for urban scene classification using Random Forests". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 66.1, pp. 56–66 (cit. on p. 88).

Guo, Qinghua, Yanjun Su, Tianyu Hu, Xiaoqian Zhao, Fangfang Wu, Yumei Li, Jin Liu, Linhai Chen, Guangcai Xu, Guanghui Lin, et al. (2017). "An integrated UAV-borne lidar system for 3D habitat mapping in three forest ecosystems across China". In: *International journal of remote sensing* 38.8-10, pp. 2954–2972 (cit. on p. 84).

Guthe, Michael, Pavel Borodin, and Reinhard Klein (2005). "Fast and accurate Hausdorff distance calculation between meshes". In: (cit. on p. 159).

Haala, Norbert and Martin Kada (2010). "An update on automatic 3D building reconstruction". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 65.6, pp. 570–580 (cit. on p. 116).

Hackel, Timo, Jan D Wegner, and Konrad Schindler (2016). "Fast semantic segmentation of 3D point clouds with strongly varying density". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Prague, Czech Republic* 3, pp. 177–184 (cit. on pp. 55, 82, 168, 181).

Hakala, Teemu, Juha Suomalainen, Sanna Kaasalainen, and Yuwei Chen (2012). "Full waveform hyperspectral LiDAR for terrestrial laser scanning". In: *Optics express* 20.7, pp. 7119–7127 (cit. on p. 81).

Haken, Hermann (1985). *Laser light dynamics*. Vol. 1. North-Holland Amsterdam (cit. on pp. 36, 77).

Hallert, Bertil (1960). "Photogrammetry, basic principles and general survey". In: (cit. on p. 63).

Han, Huiyan, Xie Han, Fusheng Sun, and Chunyan Huang (2015). "Point cloud simplification with preserved edge based on normal vector". In: *Optik-International Journal for Light and Electron Optics* 126.19, pp. 2157–2162 (cit. on p. 146).

Han, Jen-Yu, Jenny Guo, and Yi-Syuan Jiang (2013). "Monitoring tunnel profile by means of multi-epoch dispersed 3-D LiDAR point clouds". In: *Tunnelling and underground space technology* 33, pp. 186–192 (cit. on p. 65).

Hapke, Bruce (1990). "Coherent backscatter and the radar characteristics of outer planet satellites". In: *Icarus* 88.2, pp. 407–417 (cit. on p. 66).

Harary, Gur, Ayellet Tal, and Eitan Grinspun (2014). "Context-based coherent surface completion". In: *ACM Transactions on Graphics* 33.1, p. 5 (cit. on p. 113).

Hartzell, Preston, Craig Glennie, Kivanc Biber, and Shuhab Khan (2014). "Application of multispectral LiDAR to automated virtual outcrop geology". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 88, pp. 147–155 (cit. on p. 86).

Hausmann, Jean-Claude (1995). "On the Vietoris—Rips Complexes and a Cohomology Theory for Metric Spaces". In: *Prospects in Topology (AM-138), Volume 138: Proceedings of a Conference in Honor of William Browder.(AM-138)*. Vol. 320. Princeton University Press, p. 175 (cit. on p. 72).

Heinzel, Johannes and Barbara Koch (2011). "Exploring full-waveform LiDAR parameters for tree species classification". In: *International Journal of Applied Earth Observation and Geoinformation* 13.1, pp. 152–160 (cit. on p. 81).

Henderson, Sammy and Don Jacob (2018). "Space-Based Coherent Lidar for Wind Measurements with High-Percentage Tropospheric Coverage". In: *19th Coherent Laser Radar Conference* (cit. on p. 85).

Heo, Joon, Seongsu Jeong, Hyo-Keun Park, Jaehoon Jung, Soohee Han, Sungchul Hong, and Hong-Gyoo Sohn (2013). "Productive high-complexity 3D city modeling with point clouds collected from terrestrial LiDAR". In: *Computers, Environment and Urban Systems* 41, pp. 26–38 (cit. on p. 150).

Hervieu, Alexandre and Bahman Soheilian (2013). "Road side detection and reconstruction using LIDAR sensor". In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 1247–1252 (cit. on p. 83).

Holm, Sören, Ross Nelson, and Göran Ståhl (2017). "Hybrid three-phase estimators for large-area forest inventory using ground plots, airborne lidar, and space lidar". In: *Remote sensing of environment* 197, pp. 85–97 (cit. on pp. 37, 86).

Holzmann, Thomas, Martin R Oswald, Marc Pollefeys, Friedrich Fraundorfer, and Horst Bischof (2017). "Plane-based Surface Regularization for Urban 3D Reconstruction". In: *28th British Machine Vision Conference* (cit. on p. 118).

Holzmann, Thomas, Michael Maurer, Friedrich Fraundorfer, and Horst Bischof (2018). "Semantically Aware Urban 3D Reconstruction with Plane-Based Regularization". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 468–483 (cit. on p. 118).

Hoppe, Hugues (1999). "New quadric metric for simplifying meshes with appearance attributes". In: *Proceedings Visualization'99 (Cat. No. 99CB37067)*. IEEE, pp. 59–510 (cit. on p. 160).

— (1996). "Progressive meshes". In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. ACM, pp. 99–108 (cit. on pp. 65, 93, 113, 152).

Hoppe, Hugues, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle (1993). "Mesh optimization". In: *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pp. 19–26 (cit. on pp. 147, 152, 159).

Hostetler, Chris A, Michael J Behrenfeld, Yongxiang Hu, Johnathan W Hair, and Jennifer A Schulien (2018). "Spaceborne lidar in the study of marine systems". In: *Annual review of marine science* 10, pp. 121–147 (cit. on p. 86).

Hotine, M (1930). "The application of stereoscopic photography to mapping". In: *The Geographical Journal* 75.2, pp. 144–159 (cit. on p. 63).

Hu, Yongxiang, Knut Stamnes, Mark Vaughan, Jacques Pelon, Carl Weimer, Dongxiao Wu, Mike Cisewski, Wenlu Sun, Ping Yang, Bing Lin, et al. (2008). "Sea surface wind speed estimation from space-based lidar measurements". In: *Atmospheric Chemistry and Physics* 8.13, pp. 3593–3601 (cit. on p. 86).

Hua, Binh-Son, Minh-Khoi Tran, and Sai-Kit Yeung (2018). "Pointwise convolutional neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 984–993 (cit. on p. 170).

Huang, Hai, Claus Brenner, and Monika Sester (2013). "A generative statistical approach to automatic 3D building roof reconstruction from laser scanning data". In: *ISPRS Journal of photogrammetry and remote sensing* 79, pp. 29–43 (cit. on p. 117).

Huffman, David A (1954). "The synthesis of sequential switching circuits". In: (cit. on p. 176).

Hyyppä, JUHA, Hannu Hyyppä, Xiaowei Yu, HARRI Kaartinen, ANTERO Kukko, and Markus Holopainen (2017). "Forest inventory using small-footprint airborne lidar". In: *Topographic Laser Ranging and Scanning*. CRC Press, pp. 335–370 (cit. on pp. 65, 83, 86).

Ibisch, André, Stefan Stümper, Harald Altinger, Marcel Neuhausen, Marc Tschentscher, Marc Schlipsing, Jan Salinen, and Alois Knoll (2013). "Towards autonomous driving in a parking garage: Vehicle localization and tracking using environment-embedded lidar sensors". In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 829–834 (cit. on pp. 65, 88).

Inomata, Takeshi, Daniela Triadan, Flory Pinzón, Melissa Burham, José Luis Ranchos, Kazuo Aoyama, and Tsuyoshi Haraguchi (2018). "Archaeological application of airborne LiDAR to examine social changes in the Ceibal region of the Maya lowlands". In: *PloS one* 13.2, e0191619 (cit. on pp. 38, 87).

Ishimura, Daisuke, Shinji Toda, Sakae Mukoyama, Shin'ichi Homma, Kyoko Yamaguchi, and Naoya Takahashi (2019). "3D Surface Displacement and Surface Ruptures Associated with the 2014 M w 6.2 Nagano Earthquake Using Differential Lidar". In: *Bulletin of the Seismological Society of America* 109.2, pp. 780–796 (cit. on pp. 37, 87).

Jain, Anil K (1989). *Fundamentals of digital image processing*. Englewood Cliffs NJ: Prentice Hall (cit. on p. 63).

Javanmardi, Mahdi, Ehsan Javanmardi, Yanlei Gu, and Shunsuke Kamijo (2017). "Towards high-definition 3D urban mapping: Road feature-based registration of mobile mapping systems and aerial imagery". In: *Remote Sensing* 9.10, p. 975 (cit. on p. 87).

Jefferys, William H and James O Berger (1991). "Sharpening Ockham's razor on a Bayesian strop". In: *Tech nical Report* (cit. on p. 157).

Jeffreys, Harold (1939). *Theory of probability Clarendon Press* (cit. on p. 157).

Jelley, JV and BFC Cooper (1961). "An Operational Ruby Maser for Observations at 21 Centimeters with a 60-Foot Radio Telescope". In: *Review of Scientific Instruments* 32.2, pp. 166–175 (cit. on p. 77).

Jeong, Jongmin, Tae Sung Yoon, and Jin Bae Park (2018). "Multimodal sensor-based semantic 3D mapping for a large-scale environment". In: *Expert Systems with Applications* 105, pp. 1–10 (cit. on p. 113).

Johnson, Katharine M and William B Ouimet (2014). "Rediscovering the lost archaeological landscape of southern New England using airborne light detection and ranging (LiDAR)". In: *Journal of Archaeological Science* 43, pp. 9–20 (cit. on p. 87).

Johnson, William TK (1991). "Magellan imaging radar mission to Venus". In: *Proceedings of the IEEE* 79.6, pp. 777–790 (cit. on p. 66).

Jung, Jaewook, Yoonseok Jwa, and Gunho Sohn (2017). "Implicit regularization for reconstructing 3D building rooftop models using airborne LiDAR data". In: *Sensors* 17.3, p. 621 (cit. on p. 157).

Jutzi, B and U Stilla (2003). "Laser pulse analysis for reconstruction and classification of urban objects". In: *International archives of photogrammetry remote sensing and spatial information sciences* 34.3/W8, pp. 151–156 (cit. on p. 79).

— (2005). "Measuring and processing the waveform of laser pulses". In: *Optical* 3, pp. 194–203 (cit. on p. 79).

Jutzi, Boris and Uwe Stilla (2006). "Range determination with waveform recording laser systems using a Wiener Filter". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 61.2, pp. 95–107 (cit. on p. 79).

Kaick, Oliver Matias van and Hélio Pedrini (2006). "A comparative evaluation of metrics for fast mesh simplification". In: *Computer Graphics Forum*. Vol. 25. 2. Wiley Online Library, pp. 197–210 (cit. on p. 155).

Katz, Sagi, Ayellet Tal, and Ronen Basri (2007). "Direct visibility of point sets". In: *ACM SIGGRAPH 2007 papers*, 24–es (cit. on p. 197).

Kavaya, Michael J, Robert T Menzies, David A Haner, Uri P Oppenheim, and Pierre H Flamant (1983). "Target reflectance measurements for calibration of lidar atmospheric backscatter data". In: *Applied optics* 22.17, pp. 2619–2628 (cit. on p. 143).

Kazhdan, Michael and Hugues Hoppe (2013). "Screened poisson surface reconstruction". In: *ACM Transactions on Graphics (TOG)* 32.3, p. 29 (cit. on pp. 51, 160, 197).

Keele, Kenneth David (1955). *Leonardo da Vinci on vision* (cit. on p. 61).

Kelly, Tom, John Femiani, Peter Wonka, and Niloy J Mitra (2017). "BigSUR: large-scale structured urban reconstruction". In: *ACM Transactions on Graphics (TOG)* 36.6, p. 204 (cit. on p. 117).

Kereszturi, Gabor, Lauren N Schaefer, William K Schleiffarth, Jonathan Procter, Rajasheker R Pullanagari, Stuart Mead, and Ben Kennedy (2018). "Integrating airborne hyperspectral imagery and LiDAR for volcano mapping and monitoring through image classification". In: *International Journal of Applied Earth Observation and Geoinformation* 73, pp. 323–339 (cit. on p. 86).

Khan, Rafaat H and Desmond Power (1995). "Aircraft detection and tracking with high frequency radar". In: *Proceedings International Radar Conference*. IEEE, pp. 44–48 (cit. on p. 66).

Khan, S, L Aragão, and J Iriarte (2017). "A UAV–lidar system to map Amazonian rainforest and its ancient landscape transformations". In: *International journal of remote sensing* 38.8-10, pp. 2313–2330 (cit. on p. 84).

Kidono, Kiyosumi, Takeo Miyasaka, Akihiro Watanabe, Takashi Naito, and Jun Miura (2011). "Pedestrian recognition using high-definition LIDAR". In: *2011 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 405–410 (cit. on p. 88).

Kim, Hyunsuk, Gardy H Tuell, Joong Yong Park, Eric Brown, and Gwangjae We (2019). "Overview of SEAHAWK: a Bathymetric LiDAR airborne mapping system for Localization in Korea". In: *Journal of Coastal Research* 91.sp1, pp. 376–380 (cit. on p. 84).

Kim, Minsu, Yuri Kopilevich, Viktor Feygels, Joong Yong Park, and Jennifer Wozencraft (2016). "Modeling of airborne bathymetric lidar waveforms". In: *Journal of Coastal Research* 76.sp1, pp. 18–30 (cit. on p. 83).

Kim, Pileun, Jingdao Chen, and Yong K Cho (2018). "Autonomous Mobile Robot Localization and Mapping for Unknown Construction Environments". In: *ASCE Construction Research Congress*, pp. 147–156 (cit. on p. 83).

Kirscht, Martin and Carsten Rinke (1998). "3D Reconstruction of Buildings and Vegetation from Synthetic Aperture Radar (SAR) Images." In: *MVA*, pp. 228–231 (cit. on p. 65).

Klasing, Klaas, Dirk Wollherr, and Martin Buss (2008). "A clustering method for efficient segmentation of 3D laser data". In: *Robotics and Automation, 2008. ICRA 2008. IEEE International Conference on*. IEEE, pp. 4043–4048 (cit. on p. 120).

Klein, Reinhard, Gunther Liebich, and Wolfgang Straßer (1996). "Mesh reduction with error control". In: *Proceedings of Seventh Annual IEEE Visualization'96*. IEEE, pp. 311–318 (cit. on p. 158).

Knight, Joseph, Lian Rampi, Trevor Host, et al. (2017). "2015 Twin Cities Metropolitan Area Urban Tree Canopy Assessment". In: (cit. on p. 87).

Kobbelt, Leif, Swen Campagna, and Hans-Peter Seidel (1998). "A general framework for mesh decimation". In: *Graphics interface*. Vol. 98, pp. 43–50 (cit. on p. 152).

Kobbelt, Leif, Swen Campagna, Jens Vorsatz, and Hans-Peter Seidel (1998). "Interactive multi-resolution modeling on arbitrary meshes". In: *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pp. 105–114 (cit. on p. 155).

Koganov, Gennady A, Reuben Shuker, and Evgueni P Gordov (2005). "Multimirror autodyne lidar for local detection of hostile gases". In: *Applied optics* 44.15, pp. 3105–3109 (cit. on p. 77).

Kolbe, Thomas H, Gerhard Gröger, and Lutz Plümer (2005). "CityGML: Interoperable access to 3D city models". In: *Geo-information for disaster management*. Springer, pp. 883–899 (cit. on p. 68).

Kostenko, Alexei A, Alexander I Nosich, and Irina A Tishchenko (2001). "Radar prehistory, Soviet Side: three-coordinate L-Band pulse radar developed in Ukraine in the late 30's". In: *IEEE Antennas and Propagation Society International Symposium. 2001 Digest. Held in conjunction with: USNC/URSI National Radio Science Meeting (Cat. No. 01CH37229)*. Vol. 4. IEEE, pp. 44–47 (cit. on p. 65).

Kukko, Antero, Harri Kaartinen, Juha Hyyppä, and Yuwei Chen (2012). "Multiplatform mobile laser scanning: Usability and performance". In: *Sensors* 12.9, pp. 11712–11733 (cit. on p. 83).

Lafarge, Florent and Pierre Alliez (2013). "Surface reconstruction through point set structuring". In: *Computer Graphics Forum*. Vol. 32. 2pt2. Wiley Online Library, pp. 225–234 (cit. on p. 192).

Lafarge, Florent, Xavier Descombes, Josiane Zerubia, and Marc Pierrot-Deseilligny (2008). "Structural approach for building reconstruction from a single DSM". In: *IEEE PAMI* 32.1, pp. 135–147 (cit. on p. 116).

Lague, Dimitri, Patrick Launeau, Cyril Michon, Emmanuel Gouraud, Cyril Juge, William Gentile, Laurence Hubert-Moy, and Alain Crave (2016). "A geomorphologist's dream come true: synoptic high resolution river bathymetry with the latest generation of airborne dual wavelength lidar". In: *EGU General Assembly Conference Abstracts*. Vol. 18 (cit. on p. 84).

Landrieu, Loic and Mohamed Boussaha (2019). "Point cloud oversegmentation with graph-structured deep metric learning". In: *arXiv preprint arXiv:1904.02113* (cit. on pp. 125, 188).

Landrieu, Loic and Guillaume Obozinski (2016). "Cut Pursuit: fast algorithms to learn piecewise constant functions on general weighted graphs". In: (cit. on p. 178).

— (2017). "Cut pursuit: Fast algorithms to learn piecewise constant functions on general weighted graphs". In: *SIAM Journal on Imaging Sciences* 10.4, pp. 1724–1766 (cit. on pp. 46, 54, 120, 122 sqq., 144).

Landrieu, Loic and Martin Simonovsky (2018). "Large-scale point cloud semantic segmentation with superpoint graphs". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4558–4567 (cit. on pp. 82, 88, 125, 170, 188).

Landrieu, Loic, Martin Weinmann, and Clément Mallet (2017). "Comparison of belief propagation and graph-cut approaches for contextual classification of 3D LIDAR point cloud data". In: (cit. on pp. 170, 180).

Landrieu, Loic, Hugo Raguet, Bruno Vallet, Clément Mallet, and Martin Weinmann (2017). "A structured regularization framework for spatially smoothing semantic labelings of 3D point clouds". In: (cit. on pp. 120, 123, 125, 168).

Lang, T (1969). "Rules for the robot draughtsmen". In: *The Geographical Magazine* 42.1, pp. 50–51 (cit. on p. 147).

Larive, Mathieu and Veronique Gaildrat (2006). "Wall grammar for building generation". In: *Proceedings of the 4th international conference on Computer graphics and interactive techniques in Australasia and Southeast Asia*. ACM, pp. 429–437 (cit. on p. 116).

Lau, Alvaro, Kim Calders, Harm Bartholomeus, Christopher Martius, Pasi Raumonen, Martin Herold, Matheus Vicari, Hansrajie Sukhdeo, Jeremy Singh, and Rosa C Goodman (2019). "Tree Biomass Equations from Terrestrial LiDAR: A Case Study in Guyana". In: *Forests* 10.6, p. 527 (cit. on p. 86).

Leberl, F and J Thurgood (2004). "The promise of softcopy photogrammetry revisited". In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 35.Part B3, pp. 759–763 (cit. on p. 62).

Ledig, Christian, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. (2017). "Photo-realistic single image super-resolution using a generative adversarial network". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4681–4690 (cit. on p. 63).

Lee, Hyunho and Min-Ho Kyung (2016). "Parallel mesh simplification using embedded tree collapsing". In: *The Visual Computer* 32.6-8, pp. 967–976 (cit. on p. 155).

Lee, Jiyeong and Sisi Zlatanova (2009). "Solar radiation over the urban texture: LIDAR data and image processing techniques for environmental analysis at city scale". In: *3D Geo-information sciences*. Springer, pp. 319–340 (cit. on p. 88).

Lejemble, Thibault, Claudio Mura, Loïc Barthe, and Nicolas Mellado (2018). "Multiscale Planar Segments Extraction from Point Clouds". In: *Journées Françaises d'Informatique Graphique* (cit. on p. 120).

Levy, GS, RP Linfield, JS Ulvestad, CD Edwards, JF Jordan, SJ Di Nardo, CS Christensen, RA Preston, LJ Skjerve, LR Stavert, et al. (1986). "Very long baseline interferometric observations made with an orbiting radio telescope". In: *Science* 234.4773, pp. 187–189 (cit. on p. 77).

Li, Dan, Disheng Hu, Yuke Sun, and Yingsong Hu (2018). "3D scene reconstruction using a texture probabilistic grammar". In: *Multimedia Tools and Applications* 77.21, pp. 28417–28440 (cit. on p. 116).

Li, Duanshun and Chen Feng (2019). "Primitive fitting using deep geometric segmentation". In: *ISARC. Proceedings of the International Symposium on Automation and Robotics in Construction*. Vol. 36. IAARC Publications, pp. 780–787 (cit. on p. 144).

Li, Lingxiao, Minhyuk Sung, Anastasia Dubrovina, Li Yi, and Leonidas J Guibas (2019). "Supervised fitting of geometric primitives to 3d point clouds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2652–2660 (cit. on p. 118).

Li, Minglei, Peter Wonka, and Liangliang Nan (2016). "Manhattan-world urban reconstruction from point clouds". In: *ECCV*, pp. 54–69 (cit. on p. 150).

Li, Qinghua, John Degnan, Terence Barrett, and Jie Shan (2016). "First evaluation on single photon-sensitive lidar data". In: *Photogramm. Eng. Remote Sens* 82, pp. 455–463 (cit. on p. 82).

Lim, Ee Hui and David Suter (2009). "3D terrestrial LIDAR classifications with super-voxels and multi-scale Conditional Random Fields". In: *Computer-Aided Design* 41.10, pp. 701–710 (cit. on pp. 168, 170).

Lim, Kevin, Paul Treitz, Michael Wulder, Benoît St-Onge, and Martin Flood (2003). "LiDAR remote sensing of forest structure". In: *Progress in physical geography* 27.1, pp. 88–106 (cit. on p. 86).

Lin, Hui, Jizhou Gao, Yu Zhou, Guiliang Lu, Mao Ye, Chenxi Zhang, Ligang Liu, and Ruigang Yang (2013). "Semantic decomposition and reconstruction of residential scenes from LiDAR data". In: *ACM Transactions on Graphics (TOG)* 32.4, pp. 1–10 (cit. on p. 88).

Lin, Yi, Juha Hyyppa, and Anttoni Jaakkola (2010). "Mini-UAV-borne LIDAR for fine-scale mapping". In: *IEEE Geoscience and Remote Sensing Letters* 8.3, pp. 426–430 (cit. on p. 84).

Lindstrom, Peter and Greg Turk (1999). "Evaluation of memoryless simplification". In: *IEEE Transactions on Visualization and Computer Graphics* 5.2, pp. 98–115 (cit. on p. 154).

— (1998). "Fast and memory efficient polygonal simplification". In: *Proceedings Visualization'98 (Cat. No. 98CB36276)*. IEEE, pp. 279–286 (cit. on pp. 51, 147, 153 sqq., 165).

Liu, Kun, Xin Shen, Lin Cao, Guibin Wang, and Fuliang Cao (2018). "Estimating forest structural attributes using UAV-LiDAR data in Ginkgo plantations". In: *ISPRS journal of photogrammetry and remote sensing* 146, pp. 465–482 (cit. on p. 86).

Liu, Luxia, Nicholas C Coops, Neal W Aven, and Yong Pang (2017). "Mapping urban tree species using integrated airborne hyperspectral and LiDAR remote sensing data". In: *Remote Sensing of Environment* 200, pp. 170–182 (cit. on pp. 38, 88).

Liu, Miaomiao, Xuming He, and Mathieu Salzmann (2016). "Building scene models by completing and hallucinating depth and semantics". In: *European Conference on Computer Vision*. Springer, pp. 258–274 (cit. on p. 192).

Löwner, Marc-O, Joachim Benner, Gerhard Gröger, and Karl-Heinz Häfele (2013). "New concepts for structuring 3D city models–an extended level of detail concept for CityGML buildings". In: *International Conference on Computational Science and Its Applications*. Springer, pp. 466–480 (cit. on p. 68).

Luo, Shezhou, Cheng Wang, Xiaohuan Xi, Feifei Pan, Dailiang Peng, Jie Zou, Sheng Nie, and Haiming Qin (2017). "Fusion of airborne LiDAR data and hyperspectral imagery for aboveground and belowground forest biomass estimation". In: *Ecological Indicators* 73, pp. 378–387 (cit. on pp. 37, 86).

Ma, Lingni, Raphael Favier, Luat Do, Egor Bondarev, and Peter HN de With (2013). "Plane segmentation and decimation of point clouds for 3d environment reconstruction". In: *2013 IEEE 10th Consumer Communications and Networking Conference (CCNC)*. IEEE, pp. 43–49 (cit. on p. 118).

Maggiori, Emmanuel, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez (2017). "Polygonization of remote sensing classification maps by mesh approximation". In: *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, pp. 560–564 (cit. on p. 155).

Maglo, Adrien, Guillaume Lavoué, Florent Dupont, and Céline Hudelot (2015). "3d mesh compression: Survey, comparisons, and emerging trends". In: *ACM Computing Surveys (CSUR)* 47.3, pp. 1–41 (cit. on p. 160).

Maguya, Almasi, Virpi Junttila, and Tuomo Kauranne (2014). "Algorithm for extracting digital terrain models under forest canopy from airborne LiDAR data". In: *Remote Sensing* 6.7, pp. 6524–6548 (cit. on p. 86).

Maiman, Theodore H et al. (1960). "Stimulated optical radiation in ruby". In: (cit. on p. 76).

Maiwald, F, D Schneider, F Henze, S Münster, and F Niebling (2018). "FEATURE MATCHING OF HISTORICAL IMAGES BASED ON GEOMETRY OF QUADRILATERALS." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 42.2 (cit. on p. 62).

Malambo, L and M Hahn (2010). "Lidar assisted citygml creation". In: *AGSE 2010* 13 (cit. on p. 70).

Malinowski, Radosław, Bernhard Höfle, Kristina Koenig, Geoff Groom, Wolfgang Schwanghart, and Goswin Heckrath (2016). "Local-scale flood mapping on vegetated floodplains from radiometrically calibrated airborne LiDAR data". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 119, pp. 267–279 (cit. on p. 86).

Mallet, Clément and Frédéric Bretar (2009). "Full-waveform topographic lidar: State-of-the-art". In: *ISPRS Journal of photogrammetry and remote sensing* 64.1, pp. 1–16 (cit. on pp. 79 sqq.).

Mallet, Clément, Frédéric Bretar, Michel Roux, Uwe Soergel, and Christian Heipke (2011). "Relevance assessment of full-waveform lidar data for urban area classification". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 66.6, S71–S84 (cit. on p. 81).

Mandlburger, G, H Lehner, and N Pfeifer (2019). "A COMPARISON OF SINGLE PHOTON AND FULL WAVEFORM LIDAR." In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 4 (cit. on p. 82).

Mandlburger, Gottfried and Boris Jutzi (2019). "On the Feasibility of Water Surface Mapping with Single Photon LiDAR". In: *ISPRS International Journal of Geo-Information* 8.4, p. 188 (cit. on p. 82).

Marseille, GJ and A Stoffelen (2003). "Simulation of wind profiles from a space-borne Doppler wind lidar". In: *Quarterly Journal of the Royal Meteorological Society: A journal of the atmospheric sciences, applied meteorology and physical oceanography* 129.594, pp. 3079–3098 (cit. on p. 86).

Marthaler, James G and John E Heighway (1979). "Radar image processing of real aperture SLAR data for the detection and identification of iceberg and ship targets". In: (cit. on p. 66).

Martinez, Manuel, Alina Roitberg, Daniel Koester, Rainer Stiefelhagen, and Boris Schauerte (2017). "Using technology developed for autonomous cars to help navigate blind people". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 1424–1432 (cit. on p. 63).

Martinovic, Andelo and Luc Van Gool (2013). "Bayesian grammar learning for inverse procedural modeling". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 201–208 (cit. on p. 116).

Martinovic, Andelo, Jan Knopp, Hayko Riemenschneider, and Luc Van Gool (2015). "3d all the way: Semantic segmentation of urban scenes from start to end in 3d". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4456–4465 (cit. on p. 117).

Matei, Bogdan C, Harpreet S Sawhney, Supun Samarasekera, Janet Kim, and Rakesh Kumar (2008). "Building segmentation for densely built urban regions using aerial lidar data". In: *2008 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 1–8 (cit. on p. 157).

Matti, Damien, Hazım Kemal Ekenel, and Jean-Philippe Thiran (2017). "Combining lidar space clustering and convolutional neural networks for pedestrian detection".

In: *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. IEEE, pp. 1–6 (cit. on pp. 38, 88).

Maturana, Daniel and Sebastian Scherer (2015a). "3d convolutional neural networks for landing zone detection from lidar". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3471–3478 (cit. on p. 119).

— (2015b). "Voxnet: A 3d convolutional neural network for real-time object recognition". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 922–928 (cit. on p. 119).

McCarley, T Ryan, Crystal A Kolden, Nicole M Vaillant, Andrew T Hudak, Alistair MS Smith, Brian M Wing, Bryce S Kellogg, and Jason Kreitler (2017). "Multi-temporal LiDAR and Landsat quantification of fire-induced changes to forest structure". In: *Remote Sensing of Environment* 191, pp. 419–432 (cit. on pp. 37, 86).

McCormick, MP (1977). "The use of lidar for stratospheric measurements". In: (cit. on p. 76).

McCormick, MP, DM Winker, EV Browell, JA Coakley, CS Gardner, RM Hoff, GS Kent, SH Melfi, RT Menzies, CMR Piatt, et al. (1993). "Scientific investigations planned for the Lidar In-space Technology Experiment (LITE)". In: *Bulletin of the American Meteorological Society* 74.2, pp. 205–214 (cit. on p. 85).

McKeown, David M, Ted Bulwinkle, Steven Cochran, Wilson Harvey, Chris McGlone, and Jefferey A Shufelt (2000). "Performance evaluation for automatic feature extraction". In: *International Archives of Photogrammetry and Remote Sensing* 33.B2; PART 2, pp. 379–394 (cit. on p. 120).

Meigs, Andrew (2013). "Active tectonics and the LiDAR revolution". In: *Lithosphere* 5.2, pp. 226–229 (cit. on pp. 37, 86).

Meinhardt-Llopis, Enric and Marie d'Autume (2019). "3D Texturing From Multi-Date Satellite Images". In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pp. 0–0 (cit. on p. 87).

Meng, Xiaoli, Heng Wang, and Bingbing Liu (2017). "A robust vehicle localization approach based on gnss/imu/dmi/lidar sensor fusion for autonomous vehicles". In: *Sensors* 17.9, p. 2140 (cit. on p. 77).

Middleton, Wek and AF Spilhaus (1953). "The measurement of atmosferic humidity". In: *Meteorological Instruments. Toronto: University of Toronto*, pp. 105–111 (cit. on p. 76).

Mikhail, Edward M, James S Bethel, and J Chris McGlone (2001). "Introduction to modern photogrammetry". In: *New York*, p. 19 (cit. on p. 63).

Milde, J, Y Zhang, C Brenner, L Plümer, and M Sester (2008). "Building reconstruction using a structural description based on a formal grammar". In: *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 37, p. 47 (cit. on p. 116).

Miller, Russ and Quentin F. Stout (1988). "Efficient parallel convex hull algorithms". In: *IEEE transactions on Computers* 37.12, pp. 1605–1618 (cit. on p. 158).

Millward-Hopkins, JT, AS Tomlin, L Ma, DB Ingham, and M Pourkashanian (2013). "Mapping the wind resource over UK cities". In: *Renewable energy* 55, pp. 202–211 (cit. on p. 87).

Mirzaei, Faraz M, Dimitrios G Kottas, and Stergios I Roumeliotis (2012). "3D LIDAR–camera intrinsic and extrinsic calibration: Identifiability and analytical least-squares-based initialization". In: *The International Journal of Robotics Research* 31.4, pp. 452–467 (cit. on p. 77).

Mölg, Nico and Tobias Bolch (2017). "Structure-from-motion using historical aerial images to analyse changes in glacier surface elevation". In: *Remote Sensing* 9.10, p. 1021 (cit. on p. 62).

Monica, Riccardo and Jacopo Aleotti (2018). "Contour-based next-best view planning from point cloud segmentation of unknown objects". In: *Autonomous Robots* 42.2, pp. 443–458 (cit. on p. 94).

Monnier, Fabrice, Bruno Vallet, and Bahman Soheilian (2012). "Trees detection from laser point clouds acquired in dense urban areas by a mobile mapping system". In: *ISPRS Ann. Photogramm. Remote Sens. Spat. Inf. Sci* 3, pp. 245–250 (cit. on p. 83).

Monszpart, Aron, Nicolas Mellado, Gabriel J Brostow, and Niloy J Mitra (2015). "RAPter: rebuilding man-made scenes with regular arrangements of planes." In: *ACM Trans. Graph.* 34.4, pp. 103–1 (cit. on p. 119).

Morrison, Charles B, Andreas P Glinz, Zheng Zhu, James H Bechtel, Steven M Frimel, and Kenneth P Roenker (1998). "Wide-area thin film metal-semiconductor-metal photodetectors for lidar applications". In: *Photodetectors: Materials and Devices III*. Vol. 3287. International Society for Optics and Photonics, pp. 40–47 (cit. on p. 77).

Muhammad, Naveed and Simon Lacroix (2010). "Calibration of a rotating multi-beam lidar". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 5648–5653 (cit. on p. 77).

Munoz, Daniel, J Andrew Bagnell, Nicolas Vandapel, and Martial Hebert (2009). "Contextual classification with functional max-margin markov networks". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 975–982 (cit. on pp. 54, 168, 181).

Murayama, Toshiyuki, Detlef Müller, Katsuya Wada, Atsushi Shimizu, Miho Sekiguchi, and Tatsuro Tsukamoto (2004). "Characterization of Asian dust and Siberian smoke with multi-wavelength Raman lidar over Tokyo, Japan in spring 2003". In: *Geophysical Research Letters* 31.23 (cit. on p. 81).

Namouchi, Slim, Bruno Vallet, Imed Riadh Farah, and Haythem Ismail (2019). "Piecewise Horizontal 3D Roof Reconstruction from Aerial Lidar". In: *IGARSS 2019-2019 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, pp. 8992–8995 (cit. on p. 125).

Nan, Liangliang and Peter Wonka (2017). "Polyfit: Polygonal surface reconstruction from point clouds". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Venice, Italy*, pp. 2353–2361 (cit. on pp. 51, 65, 113, 118, 120, 150, 160, 165).

Nguyen, Xuan Truong, Van Luan Dinh, Hyuk-Jae Lee, and Hyun Kim (2017). "A high-definition LIDAR system based on two-mirror deflection scanners". In: *IEEE Sensors Journal* 18.2, pp. 559–568 (cit. on p. 77).

Niclass, Cristiano, Daisuke Inoue, Hiroyuki Matsubara, Tadashi Ichikawa, and Mineki Soga (2015). "Development of Automotive LIDAR". In: *Electronics and Communications in Japan* 98.5, pp. 28–33 (cit. on p. 77).

Nicodemus, Fred Edwin, JC Richmond, JJ Hsia, JW Ginsberg, and T Limperis (1977). "Geometrical considerations and nomenclature for reflectance." In: (cit. on p. 143).

Nie, Sheng, Cheng Wang, Hongcheng Zeng, Xiaohuan Xi, and Guicai Li (2017). "Above-ground biomass estimation using airborne discrete-return and full-waveform LiDAR data in a coniferous forest". In: *Ecological Indicators* 78, pp. 221–228 (cit. on p. 81).

Niemeyer, Joachim, Franz Rottensteiner, and Uwe Soergel (2014). "Contextual classification of lidar data and building object detection in urban areas". In: *ISPRS journal of photogrammetry and remote sensing* 87, pp. 152–165 (cit. on pp. 55, 168, 170, 179).

Niemeyer, Joachim, Jan Dirk Wegner, Clément Mallet, Franz Rottensteiner, and Uwe Soergel (2011). "Conditional random fields for urban scene classification with full waveform LiDAR data". In: *Photogrammetric Image Analysis*. Springer, pp. 233–244 (cit. on p. 176).

Niemeyer, Joachim, Franz Rottensteiner, Uwe Soergel, and Christian Heipke (2016). "Hierarchical higher order crf for the classification of airborne lidar point clouds in urban areas". In: *23rd International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences Congress, ISPRS 2016, 12–19 July 2016, Prague, Czech Republic*. Göttingen: Copernicus GmbH (cit. on pp. 168, 170).

Nishida, Gen, Ignacio Garcia-Dorado, Daniel G Aliaga, Bedrich Benes, and Adrien Bousseau (2016). "Interactive sketching of urban procedural models". In: *ACM Transactions on Graphics (TOG)* 35.4, p. 130 (cit. on p. 117).

Nissen, Edwin, Sri Raghu Malireddi, Kate Clark, Ian Hamling, Robert Langridge, William Ries, Andrea Tagliasacchi, KJ Clark, P Upton, K Kelly, et al. (2017). "Three-dimesnional coastal deformation in the Mw 7.8 Kaikōura earthquake from differntial airborne lidar". In: *Proc. of the 8th International INQUA Meeting on Paleoseismology, Active Tectonics and Archeoseismology: Handbook and Programme*, pp. 13–16 (cit. on p. 87).

Northend, CA, RC Honey, and WE Evans (1966). "Laser radar (lidar) for meteorological observations". In: *Review of Scientific Instruments* 37.4, pp. 393–400 (cit. on p. 76).

Ochmann, Sebastian, Richard Vock, Raoul Wessel, and Reinhard Klein (2016). "Automatic reconstruction of parametric building models from indoor point clouds". In: *Computers & Graphics* 54, pp. 94–103 (cit. on p. 144).

Odaker, Thomas, Dieter Kranzlmueller, and Jens Volkert (2015). "View-dependent simplification using parallel half edge collapses". In: (cit. on p. 155).

Oesau, Sven, Florent Lafarge, and Pierre Alliez (2016). "Planar shape detection and regularization in tandem". In: *Computer Graphics Forum*. Vol. 35. 1. Wiley Online Library, pp. 203–215 (cit. on p. 119).

O'Handley, Douglas A and William B Green (1972). "Recent developments in digital image processing at the Image Processing Laboratory at the Jet Propulsion Laboratory". In: *Proceedings of the IEEE* 60.7, pp. 821–828 (cit. on p. 62).

Ohori, Ken Arroyo, Hugo Ledoux, Filip Biljecki, and Jantien Stoter (2015). "Modeling a 3D city model and its levels of detail as a true 4D model". In: *ISPRS International Journal of Geo-Information* 4.3, pp. 1055–1075 (cit. on p. 70).

Olariu, Stephan, James L. Schwing, and Jingyuan Zhang (1993). "Optimal convex hull algorithms on enhanced meshes". In: *BIT Numerical Mathematics* 33.3, pp. 396–410 (cit. on p. 158).

Olsen, Michael J and Robert Kayen (2013). "Post-earthquake and tsunami 3D laser scanning forensic investigations". In: *Forensic Engineering 2012: Gateway to a Safer Tomorrow*, pp. 477–486 (cit. on p. 193).

Opitz, David and Richard Maclin (1999). "Popular ensemble methods: An empirical study". In: *Journal of Artificial Intelligence Research* 11, pp. 169–198 (cit. on p. 169).

Oren, Michael and Shree K Nayar (1994). "Generalization of Lambert's reflectance model". In: *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pp. 239–246 (cit. on p. 143).

Otepka, G and J Loitsch (1979). "Digitally Controlled Production of Orthophotos and". In: *Photogrammetric Engineering and Remote Sensing* 45.10, pp. 1353–1362 (cit. on p. 61).

Pan, Zhigeng, Kun Zhou, and Jiaoying Shi (2001). "A new mesh simplification algorithm based on triangle collapses". In: *Journal of computer science and technology* 16.1, pp. 57–63 (cit. on p. 155).

Paparoditis, Nicolas, Jean-Pierre Papelard, Bertrand Cannelle, Alexandre Devaux, Bahman Soheilian, Nicolas David, and Erwann Houzay (2012). "Stereopolis II: A multi-purpose and multi-sensor 3D mobile mapping system for street visualisation and 3D metrology". In: *Revue française de photogrammétrie et de télédétection* 200.1, pp. 69–79 (cit. on pp. 43, 62, 83, 104, 128, 180).

Pauly, Mark, Markus Gross, and Leif P Kobbelt (2002). "Efficient simplification of point-sampled surfaces". In: *IEEE Visualization, 2002. VIS 2002.* IEEE, pp. 163–170 (cit. on p. 146).

Phillips, Cynthia B and Robert T Pappalardo (2014). "Europa Clipper mission concept: exploring Jupiter's ocean moon". In: *Eos, Transactions American Geophysical Union* 95.20, pp. 165–167 (cit. on p. 66).

Piepereit, R, A Beuster, M von der Gruen, U Voß, M Pries, and U Wagner (2019). "TOWARDS WIND-SIMULATION OF VIRTUAL 3D CITY MODELS IN A COLLABORATIVE VR ENVIRONMENT." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* (cit. on pp. 58, 192).

Plaza-Leiva, Victoria, Jose Gomez-Ruiz, Anthony Mandow, and Alfonso García-Cerezo (2017). "Voxel-based neighborhood for spatial shape pattern classification of lidar point clouds with supervised learning". In: *Sensors* 17.3, p. 594 (cit. on p. 119).

Pokorny, Florian T, Majd Hawasly, and Subramanian Ramamoorthy (2016). "Topological trajectory classification with filtrations of simplicial complexes and persistent homology". In: *The International Journal of Robotics Research* 35.1-3, pp. 204–223 (cit. on p. 93).

Pomerleau, François, Francis Colas, Roland Siegwart, et al. (2015). "A review of point cloud registration algorithms for mobile robotics". In: *Foundations and Trends® in Robotics* 4.1, pp. 1–104 (cit. on p. 83).

Ponciano, Jean-Jacques, Alain Trémeau, and Frank Boochs (2019). "Automatic Detection of Objects in 3D Point Clouds Based on Exclusively Semantic Guided Processes". In: *ISPRS International Journal of Geo-Information* 8.10, p. 442 (cit. on p. 116).

Popović, Jovan and Hugues Hoppe (1997). "Progressive simplicial complexes". In: *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., pp. 217–224 (cit. on pp. 93, 113).

Poux, Florent and Roland Billen (2019). "Voxel-based 3D point cloud semantic segmentation: unsupervised geometric and relationship featuring vs deep learning methods". In: *ISPRS International Journal of Geo-Information* 8.5, p. 213 (cit. on p. 119).

Premebida, Cristiano, Joao Carreira, Jorge Batista, and Urbano Nunes (2014). "Pedestrian detection combining rgb and dense lidar data". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, pp. 4112–4117 (cit. on p. 78).

Priedhorsky, William C, R Clayton Smith, and Cheng Ho (1996). "Laser ranging and mapping with a photon-counting detector". In: *Applied optics* 35.3, pp. 441–452 (cit. on p. 82).

Putkinen, Niko, Nick Eyles, Satu Putkinen, Antti EK Ojala, Jukka-Pekka Palmu, Pertti Sarala, Tapio Väänänen, Jukka Räisänen, Jouko Saarelainen, Niina Ahtonen, et al. (2017). "High-resolution LiDAR mapping of glacial landforms and ice stream lobes in Finland." In: *Bulletin of the Geological Society of Finland* 89.2 (cit. on pp. 37, 86).

Qi, Charles R, Hao Su, Kaichun Mo, and Leonidas J Guibas (2017). "Pointnet: Deep learning on point sets for 3d classification and segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660 (cit. on pp. 82, 170).

Quadros, ND, PA Collier, and CS Fraser (2008). "Integration of bathymetric and topographic LiDAR: a preliminary investigation". In: *The International Archives of the Pho-*

togrammetry, Remote Sensing and Spatial Information Sciences 36, pp. 1299–1304 (cit. on p. 83).

Raczko, Edwin and Bogdan Zagajewski (2017). "Comparison of support vector machine, random forest and neural network classifiers for tree species classification on airborne hyperspectral APEX images". In: *European Journal of Remote Sensing* 50.1, pp. 144–154 (cit. on p. 170).

Radovic, Matija, Offei Adarkwa, and Qiaosong Wang (2017). "Object recognition in aerial images using convolutional neural networks". In: *Journal of Imaging* 3.2, p. 21 (cit. on p. 63).

Ramakrishnan, Shivakumar, Vincent Demarcus, Jerome Le Ny, Neal Patwari, and Joel Gussy (2002). "Synthetic aperture radar imaging using spectral estimation techniques". In: *Advanced Signal Processing* (cit. on p. 65).

Ramsey, Norman F (1983). "History of atomic clocks". In: *Journal of research of the National Bureau of Standards* 88.5, p. 301 (cit. on p. 77).

Reitberger, Josef, Cl Schnörr, Peter Krzystek, and Uwe Stilla (2009). "3D segmentation of single trees exploiting full waveform LIDAR data". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 64.6, pp. 561–574 (cit. on p. 81).

Ribeiro, SCL, M Jarzabek-Rychard, JP Cintra, and H-G Maas (2019). "DESCRIBING THE VERTICAL STRUCTURE OF INFORMAL SETTLEMENTS ON THE BASIS OF LIDAR DATA–A CASE STUDY FOR FAVELAS (SLUMS) IN SAO PAULO CITY." In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 4 (cit. on p. 88).

Richardson, Jeffrey J and L Monika Moskal (2011). "Strengths and limitations of assessing forest density and spatial configuration with aerial LiDAR". In: *Remote Sensing of Environment* 115.10, pp. 2640–2651 (cit. on p. 86).

Ripperda, Nora and Claus Brenner (2009). "Application of a formal grammar to facade reconstruction in semiautomatic and automatic environments". In: *Proc. of the 12th AGILE Conference on GIScience*, pp. 1–12 (cit. on p. 117).

Rissanen, Jorma (1978). "Modeling by shortest data description". In: *Automatica* 14.5, pp. 465–471 (cit. on p. 157).

Rodríguez-Gonzálvez, Pablo, Belén Jiménez Fernández-Palacios, Ángel Muñoz-Nieto, Pedro Arias-Sanchez, and Diego Gonzalez-Aguilera (2017). "Mobile LiDAR system: New possibilities for the documentation and dissemination of large cultural heritage sites". In: *Remote Sensing* 9.3, p. 189 (cit. on pp. 38, 87).

Romanoni, Andrea, Amaël Delaunoy, Marc Pollefeys, and Matteo Matteucci (2016). "Automatic 3d reconstruction of manifold meshes via delaunay triangulation and mesh sweeping". In: *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE, pp. 1–8 (cit. on pp. 39, 96).

Rosenfeld, Azriel (1969). "Picture processing by computer". In: *ACM Computing Surveys (CSUR)* 1.3, pp. 147–176 (cit. on p. 63).

Rossignac, Jarek and Paul Borrel (1993). "Multi-resolution 3D approximations for rendering complex scenes". In: *Modeling in computer graphics*. Springer, pp. 455–465 (cit. on p. 152).

Ruiz, Reuben M, Denis A Elliott, Gary M Yagi, Richard B Pomphrey, Margaret A Power, K Winslow Farrell Jr, Jean J Lorre, William D Benton, Robert E Dewar, and Louise E Cullen (1977). "IPL processing of the Viking Orbiter images of Mars". In: *Journal of Geophysical Research* 82.28, pp. 4189–4202 (cit. on p. 62).

Rupnik, Ewelina, Mehdi Daakir, and Marc Pierrot Deseilligny (2017). "MicMac–a free, open-source solution for photogrammetry". In: *Open Geospatial Data, Software and Standards* 2.1, pp. 1–9 (cit. on p. 62).

Russell, Bryan C, Josef Sivic, Jean Ponce, and Helene Dessales (2011). "Automatic alignment of paintings and photographs depicting a 3D scene". In: *2011 IEEE international conference on computer vision workshops (ICCV workshops)*. IEEE, pp. 545–552 (cit. on p. 94).

Rusu, Radu Bogdan, Nico Blodow, Zoltan Csaba Marton, and Michael Beetz (2009). "Close-range scene segmentation and reconstruction of 3D point cloud maps for mobile manipulation in domestic environments". In: *International Conference on Intelligent Robots and Systems* (cit. on p. 113).

Rutzinger, Martin, Bernhard Höfle, Markus Hollaus, and Norbert Pfeifer (2008). "Object-based point cloud analysis of full-waveform airborne laser scanning data for urban vegetation classification". In: *Sensors* 8.8, pp. 4505–4528 (cit. on p. 168).

Salberg, Arnt-Børre, Øivind Due Trier, and Michael Kampffmeyer (2017). "Large-scale mapping of small roads in lidar images using deep convolutional neural networks". In: *Scandinavian Conference on Image Analysis*. Springer, pp. 193–204 (cit. on p. 176).

Salinas, David, Florent Lafarge, and Pierre Alliez (2015). "Structure-aware mesh decimation". In: *Computer Graphics Forum*. Vol. 34. 6. Wiley Online Library, pp. 211–227 (cit. on p. 155).

Sankey, Temuulen, Jonathon Donager, Jason McVay, and Joel B Sankey (2017). "UAV lidar and hyperspectral fusion for forest monitoring in the southwestern USA". In: *Remote Sensing of Environment* 195, pp. 30–43 (cit. on p. 84).

Sankey, Temuulen T, Jason McVay, Tyson L Swetnam, Mitchel P McClaran, Philip Heilman, and Mary Nichols (2018). "UAV hyperspectral and lidar data and their fusion for arid and semi-arid land vegetation monitoring". In: *Remote Sensing in Ecology and Conservation* 4.1, pp. 20–33 (cit. on p. 84).

Sasano, Yasuhiro and Edward V Browell (1989). "Light scattering characteristics of various aerosol types derived from multiple wavelength lidar observations". In: *Applied optics* 28.9, pp. 1670–1679 (cit. on p. 81).

Sato, Gentei (1991). "A secret story about the Yagi antenna". In: *IEEE Antennas and Propagation Magazine* 33.3, pp. 7–18 (cit. on p. 65).

Schaefer, Scott and Joe Warren (2003). "Adaptive vertex clustering using octrees". In: *SIAM geometric design and computing* 2.6 (cit. on p. 152).

Schawlow, Arthur L and Charles H Townes (1958). "Infrared and optical masers". In: *Physical Review* 112.6, p. 1940 (cit. on p. 76).

Schellekens, Maurice (2015). "Self-driving cars and the chilling effect of liability law". In: *Computer Law & Security Review* 31.4, pp. 506–517 (cit. on p. 88).

Schmidt, Mark (2007). *UGM: A Matlab toolbox for probabilistic undirected graphical models* (cit. on p. 180).

Schnabel, Ruwen, Roland Wahl, and Reinhard Klein (2007). "Efficient RANSAC for point-cloud shape detection". In: *Computer graphics forum*. Vol. 26. 2. Wiley Online Library, pp. 214–226 (cit. on pp. 118, 197).

Schramm, Matthias (1961). *Anthemius of Tralles. A Study in Later Greek Geometry* (cit. on p. 61).

Schroeder, William J, Jonathan A Zarge, and William E Lorensen (1992). "Decimation of triangle meshes". In: *Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pp. 65–70 (cit. on p. 152).

Schuster, Hanns-Florian and Uwe Weidner (2003). "A new approach towards quantitative quality evaluation of 3D building models". In: *Workshop of the ISPRS* (cit. on p. 120).

Shafer, Glenn (1992). "Dempster-shafer theory". In: *Encyclopedia of artificial intelligence* 1, pp. 330–331 (cit. on p. 69).

Shapovalov, Roman, Er Velizhev, and Olga Barinova (2010). "Non-associative markov networks for 3d point cloud classification". In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVIII, Part 3A*. Citeseer (cit. on pp. 168, 170).

Sharma, Gopal, Rishabh Goyal, Difan Liu, Evangelos Kalogerakis, and Subhransu Maji (2018). "Csgnet: Neural shape parser for constructive solid geometry". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5515–5523 (cit. on pp. 117 sq.).

Shekhar, Raj, Elias Fayyad, Roni Yagel, and J Fredrick Cornhill (1996). "Octree-based decimation of marching cubes surfaces". In: *Proceedings of Seventh Annual IEEE Visualization'96*. IEEE, pp. 335–342 (cit. on p. 146).

Shen, Xin, Lin Cao, Dong Chen, Yuan Sun, Guibin Wang, and Honghua Ruan (2018). "Prediction of forest structural parameters using airborne full-waveform LiDAR and hyperspectral data in subtropical forests". In: *Remote Sensing* 10.11, p. 1729 (cit. on p. 81).

Shi, Bao-Quan, Jin Liang, and Qing Liu (2011). "Adaptive simplification of point cloud using k-means clustering". In: *Computer-Aided Design* 43.8, pp. 910–922 (cit. on p. 146).

Shi, Jianbo and Jitendra Malik (2000). "Normalized cuts and image segmentation". In: *IEEE Transactions on pattern analysis and machine intelligence* 22.8, pp. 888–905 (cit. on p. 120).

Shotton, Jamie, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon (2013). "Scene coordinate regression forests for camera relocalization in RGB-D images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2930–2937 (cit. on p. 94).

Si, Hang (2015). "TetGen, a Delaunay-based quality tetrahedral mesh generator". In: *ACM Transactions on Mathematical Software (TOMS)* 41.2, p. 11 (cit. on p. 96).

Simard, Marc, Naiara Pinto, Joshua B Fisher, and Alessandro Baccini (2011). "Mapping forest canopy height globally with spaceborne lidar". In: *Journal of Geophysical Research: Biogeosciences* 116.G4 (cit. on p. 86).

Sohn, Gunho and Ian Dowman (2007). "Data fusion of high-resolution satellite imagery and LiDAR data for automatic building extraction". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 62.1, pp. 43–63 (cit. on p. 157).

Soilán Rodríguez, Mario, R Lindenbergh, Belén Riveiro Rodríguez, Ana Sánchez Rodríguez, et al. (2019). "Pointnet for the automatic classification of aerial point clouds". In: (cit. on p. 170).

Soucy, Marc and Denis Laurendeau (1996). "Multiresolution surface modeling based on hierarchical triangulation". In: *Computer vision and image understanding* 63.1, pp. 1–14 (cit. on p. 152).

Soussen, Charles, Jérôme Idier, David Brie, and Junbo Duan (2011). "From Bernoulli–Gaussian deconvolution to sparse signal restoration". In: *IEEE Transactions on Signal Processing* 59.10, pp. 4572–4584 (cit. on p. 123).

Steinbrucker, Frank, Christian Kerl, and Daniel Cremers (2013). "Large-scale multi-resolution surface reconstruction from RGB-D sequences". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 3264–3271 (cit. on p. 94).

Stitch, Malcolm L (1961). "Power output characteristics of a ruby laser". In: *Journal of Applied Physics* 32.10, pp. 1994–1999 (cit. on p. 76).

Strom, Johannes, Andrew Richardson, and Edwin Olson (2010). "Graph-based segmentation for colored 3D laser point clouds". In: *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, pp. 2131–2136 (cit. on p. 120).

Suomi, Irene, Sven-Erik Gryning, Ewan J O'Connor, and Timo Vihma (2017). "Methodology for obtaining wind gusts using Doppler lidar". In: *Quarterly Journal of the Royal Meteorological Society* 143.706, pp. 2061–2072 (cit. on pp. 37, 85).

Swatantran, Anu, Hao Tang, Terence Barrett, Phil DeCola, and Ralph Dubayah (2016). "Rapid, high-resolution forest structure and terrain mapping over large areas using single photon lidar". In: *Scientific reports* 6, p. 28277 (cit. on p. 82).

Takeda, Yasushi (1986). "Velocity profile measurement by ultrasound Doppler shift method". In: *International journal of heat and fluid flow* 7.4, pp. 313–318 (cit. on p. 85).

Talton, Jerry O, Yu Lou, Steve Lesser, Jared Duke, Radomír Měch, and Vladlen Koltun (2011). "Metropolis procedural modeling". In: *ACM Transactions on Graphics (TOG)* 30.2, p. 11 (cit. on p. 117).

Tang, Hao, Anu Swatantran, Terence Barrett, Phil DeCola, and Ralph Dubayah (2016). "Voxel-based spatial filtering method for canopy height retrieval from airborne single-photon LiDAR". In: *Remote Sensing* 8.9, p. 771 (cit. on p. 82).

Tao, C Vincent and Jonathan Li (2007). *Advances in mobile mapping technology*. Vol. 4. CRC Press (cit. on p. 83).

Tatavarti, Aparna, John Papadakis, and Andrew R Willis (2017). "Towards real-time segmentation of 3D point cloud data into local planar regions". In: *SoutheastCon, 2017*. IEEE, pp. 1–6 (cit. on p. 117).

Taylor, Albert H, Leo C Young, and Lawrence A Hyland (1934). *System for detecting objects by radio*. US Patent 1,981,884 (cit. on p. 65).

Taylor, John W and GUNTIS Brunins (1985). "Design of a new airport surveillance radar (ASR-9)". In: *Proceedings of the IEEE* 73.2, pp. 284–289 (cit. on p. 66).

Tchapmi, Lyne, Christopher Choy, Iro Armeni, JunYoung Gwak, and Silvio Savarese (2017). "Segcloud: Semantic segmentation of 3d point clouds". In: *2017 International Conference on 3D Vision (3DV)*. IEEE, pp. 537–547 (cit. on pp. 88, 119, 170).

Teboul, Olivier, Iasonas Kokkinos, Loic Simon, Panagiotis Koutsourakis, and Nikos Paragios (2011). "Shape grammar parsing via reinforcement learning". In: *CVPR 2011*. IEEE, pp. 2273–2280 (cit. on p. 117).

Tenbrinck, Daniel, Fjedor Gaede, and Martin Burger (2019). "Variational Graph Methods for Efficient Point Cloud Sparsification". In: *arXiv preprint arXiv:1903.02858* (cit. on p. 125).

Teo, Tee-Ann and Shih-Han Huang (2014). "Surface-based registration of airborne and terrestrial mobile LiDAR point clouds". In: *Remote Sensing* 6.12, pp. 12686–12707 (cit. on p. 77).

Thompson, F Vivian (1908). "Stereo-photo surveying". In: *The Geographical Journal* 31.5, pp. 534–549 (cit. on p. 63).

Thompson, Morris M, Robert C Eller, William A Radlinski, and Juliun L Speert (1966). *Manual of photogrammetry*. Vol. 1. American Society of Photogrammetry Falls Church, Va. (cit. on p. 61).

Toffoli, Tommaso and Norman Margolus (1987). *Cellular automata machines: a new environment for modeling*. MIT press (cit. on pp. 39, 63, 94).

Toshev, Alexander, Philippos Mordohai, and Ben Taskar (2010). "Detecting and parsing architecture at city scale from range data". In: *CVPR*, pp. 398–405 (cit. on p. 116).

Toth, Charles K (2009). "R&D of mobile LiDAR mapping and future trends". In: *Proc. ASPRS Annu. Conf*, pp. 9–13 (cit. on p. 82).

Tournaire, O, B Soheilian, and N Paparoditis (2006). "Towards a sub-decimetric georeferencing of groundbased mobile mapping systems in urban areas: Matching ground-based and aerial-based imagery using roadmarks". In: *International Archives of Photogrammetry and Remote Sensing. Proceedings..., Paris* (cit. on p. 83).

Tran, H and K Khoshelham (2019). "a Stochastic Approach to Automated Reconstruction of 3d Models of Interior Spaces from Point Clouds". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 299–306 (cit. on p. 117).

Tsochantaridis, Ioannis, Thomas Hofmann, Thorsten Joachims, and Yasemin Altun (2004). "Support vector machine learning for interdependent and structured output spaces". In: *Proceedings of the twenty-first international conference on Machine learning*. ACM, p. 104 (cit. on p. 117).

Tulsiani, Shubham, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik (2017). "Learning shape abstractions by assembling volumetric primitives". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2635–2643 (cit. on p. 118).

Tutzauer, P and N Haala (2015). "Façade reconstruction using geometric and radiometric point cloud information". In: *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences* 40.3, p. 247 (cit. on p. 116).

Unwin, RS (1992). "The development of radar in New Zealand in World War II". In: *IEEE Antennas and Propagation Magazine* 34.3, pp. 31–39 (cit. on p. 65).

Upadhyay, Aakriti, Weifu Wang, and Chinwe Ekenna (2019). "Approximating Cfree space topology by constructing Vietoris-Rips complex". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 2517–2523 (cit. on p. 72).

Vallet, Bruno, Mathieu Brédif, Andrés Serna, Beatriz Marcotegui, and Nicolas Paparoditis (2015). "TerraMobilita/iQmulus urban point cloud analysis benchmark". In: *Computers & Graphics* 49, pp. 126–133 (cit. on pp. 39, 97).

Van Sinh, Nguyen, Tran Manh Ha, and Nguyen Tien Thanh (2016). "Filling holes on the surface of 3D point clouds based on tangent plane of hole boundary points". In: *Proceedings of the Seventh Symposium on Information and Communication Technology*. ACM, pp. 331–338 (cit. on p. 113).

Vanegas, Carlos A, Daniel G Aliaga, and Bedřich Beneš (2010). "Building reconstruction using manhattan-world grammars". In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, pp. 358–365 (cit. on p. 116).

Vauhkonen, J (2015). "Reconstruction, quantification, and visualization of forest canopy based on 3D triangulations of airborne laser scanning point data". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2.3, p. 255 (cit. on p. 93).

Vauhkonen, Jari, Teemu Hakala, Juha Suomalainen, Sanna Kaasalainen, Olli Nevalainen, Mikko Vastaranta, Markus Holopainen, and Juha Hyyppä (2013). "Classification of spruce and pine trees using active hyperspectral LiDAR". In: *IEEE Geoscience and Remote Sensing Letters* 10.5, pp. 1138–1141 (cit. on p. 81).

Verdie, Yannick, Florent Lafarge, and Pierre Alliez (2015). "Lod generation for urban scenes". In: *ACM Transactions on Graphics* 34.ARTICLE, p. 30 (cit. on pp. 57, 192).

Vicari, Matheus B, Mathias Disney, Phil Wilkes, Andrew Burt, Kim Calders, and William Woodgate (2019). "Leaf and wood classification framework for terrestrial LiDAR point clouds". In: *Methods in Ecology and Evolution* 10.5, pp. 680–694 (cit. on p. 169).

Vidal, Vincent, Christian Wolf, and Florent Dupont (2014). "Mechanical Mesh Segmentation and Global 3D Shape Extraction". PhD thesis. Université Lyon 1-Claude Bernard; INSA Lyon (cit. on p. 144).

Vincent, Matthew L, Mariano Flores Gutierrez, Chance Coughenour, Victor Manuel, Lopez-Menchero Bendicho, Fabio Remondino, and Dieter Fritsch (2015). "Crowd-

sourcing the 3D digital reconstructions of lost cultural heritage". In: *2015 Digital Heritage*. Vol. 1. IEEE, pp. 171–172 (cit. on p. 87).

Visvalingam, Mahes and J Duncan Whyatt (1990). "The Douglas-Peucker Algorithm for Line Simplification: Re-evaluation through Visualization". In: *Computer Graphics Forum*. Vol. 9. 3. Wiley Online Library, pp. 213–225 (cit. on p. 147).

Visvalingam, Maheswari and James D Whyatt (1993). "Line generalisation by repeated elimination of points". In: *The cartographic journal* 30.1, pp. 46–51 (cit. on p. 147).

Vlaminck, Michiel, Hiep Quang Luong, Werner Goeman, Peter Veelaert, and Wilfried Philips (2016). "Towards online mobile mapping using inhomogeneous lidar data". In: *2016 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 845–850 (cit. on p. 82).

Vo, Anh-Vu, Linh Truong-Hong, Debra F Laefer, and Michela Bertolotto (2015). "Octree-based region growing for point cloud segmentation". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 104, pp. 88–100 (cit. on p. 119).

Vosselman, George, Maximilian Coenen, and Franz Rottensteiner (2017). "Contextual segment-based classification of airborne laser scanner data". In: *ISPRS journal of photogrammetry and remote sensing* 128, pp. 354–371 (cit. on p. 168).

Wagner, Wolfgang, Andreas Ullrich, Thomas Melzer, Christian Briese, and Karl Kraus (2004). *From single-pulse to full-waveform airborne laser scanners: potential and practical challenges*. na (cit. on p. 79).

Wallace, Luke, Arko Lucieer, and Christopher S Watson (2014). "Evaluating tree detection and segmentation routines on very high resolution UAV LiDAR data". In: *IEEE Transactions on Geoscience and Remote Sensing* 52.12, pp. 7619–7628 (cit. on p. 84).

Wallace, Luke, Arko Lucieer, Christopher Watson, and Darren Turner (2012). "Development of a UAV-LiDAR system with application to forest inventory". In: *Remote Sensing* 4.6, pp. 1519–1543 (cit. on pp. 84, 86).

Wan, Guowei and Andrei Sharf (2012). "Grammar-based 3D facade segmentation and reconstruction". In: *Computers & Graphics* 36.4, pp. 216–223 (cit. on p. 116).

Wang, Chi-Kuei and William D Philpot (2007). "Using airborne bathymetric lidar to detect bottom type variation in shallow waters". In: *Remote Sensing of Environment* 106.1, pp. 123–135 (cit. on p. 84).

Wang, Yanjun, Qi Chen, Qing Zhu, Lin Liu, Chaokui Li, and Dunyong Zheng (2019). "A Survey of Mobile Laser Scanning Applications and Key Techniques over Urban Areas". In: *Remote Sensing* 11.13, p. 1540 (cit. on pp. 38, 87).

Wang, Yanmin and Hongbin Shi (2014). "A Segmentation Method for Point Cloud Based on Local Sample and Statistic Inference". In: *International Conference on Geo-Informatics in Resource Management and Sustainable Ecosystem*. Springer, pp. 274–282 (cit. on p. 191).

Wang, Yuan, Tianyue Shi, Peng Yun, Lei Tai, and Ming Liu (2018). "Pointseg: Real-time semantic segmentation based on 3d lidar point cloud". In: *arXiv preprint arXiv:1807.06288* (cit. on p. 193).

Wasil, BA and DC Merchant (1964). "Plate-deflection measurement by photogrammetric methods". In: *Experimental Mechanics* 4.3, pp. 77–83 (cit. on p. 61).

Wästlund, André, Johan Holmgren, Eva Lindberg, and Håkan Olsson (2018). "Forest variable estimation using a high altitude single photon lidar system". In: *Remote Sensing* 10.9, p. 1422 (cit. on p. 82).

Weber, Joseph (1953). "Amplification of microwave radiation by substances not in thermal equilibrium". In: *Transactions of the IRE Professional Group on Electron Devices* 3, pp. 1–4 (cit. on p. 76).

Wei, Gong, Song Shalei, Zhu Bo, Shi Shuo, Li Faquan, and Cheng Xuewu (2012). "Multi-wavelength canopy LiDAR for remote sensing of vegetation: Design and system

performance". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 69, pp. 1–9 (cit. on p. 143).

Wei, Junqing, Jarrod M Snider, Junsung Kim, John M Dolan, Raj Rajkumar, and Bakhtiar Litkouhi (2013). "Towards a viable autonomous driving research platform". In: *2013 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, pp. 763–770 (cit. on p. 88).

Wei, Shen et al. (2008). "Building boundary extraction based on lidar point clouds data". In: *Proceedings of the International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 37, pp. 157–161 (cit. on p. 88).

Weinmann, M, Steffen Urban, Stefan Hinz, Boris Jutzi, and Clément Mallet (2015a). "Distinctive 2D and 3D features for automated large-scale scene analysis in urban areas". In: *Computers & Graphics* 49, pp. 47–57 (cit. on pp. 142, 168 sq., 171, 182).

Weinmann, Martin, Boris Jutzi, and Clément Mallet (2014). "Semantic 3D scene interpretation: A framework combining optimal neighborhood size selection with relevant features". In: *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2.3, p. 181 (cit. on p. 95).

Weinmann, Martin, Boris Jutzi, Stefan Hinz, and Clément Mallet (2015b). "Semantic point cloud interpretation based on optimal neighborhoods, relevant features and efficient classifiers". In: *ISPRS Journal of Photogrammetry and Remote Sensing* 105, pp. 286–304 (cit. on pp. 39, 53, 95).

Whelan, Thomas, Lingni Ma, Egor Bondarev, PHN de With, and John McDonald (2015). "Incremental and batch planar simplification of dense point cloud maps". In: *Robotics and Autonomous Systems* 69, pp. 3–14 (cit. on p. 119).

Winkelbach, Simon, Markus Rilk, Christoph Schönfelder, and Friedrich M Wahl (2004). "Fast random sample matching of 3d fragments". In: *Joint Pattern Recognition Symposium*. Springer, pp. 129–136 (cit. on p. 146).

Winker, David M, Richard H Couch, and MPatrick McCormick (1996). "An overview of LITE: NASA's lidar in-space technology experiment". In: *Proceedings of the IEEE* 84.2, pp. 164–180 (cit. on p. 85).

Winker, David M, Mark A Vaughan, Ali Omar, Yongxiang Hu, Kathleen A Powell, Zhaoyan Liu, William H Hunt, and Stuart A Young (2009). "Overview of the CALIPSO mission and CALIOP data processing algorithms". In: *Journal of Atmospheric and Oceanic Technology* 26.11, pp. 2310–2323 (cit. on p. 85).

Wirth, Martin, Andreas Fix, Peter Mahnke, Horst Schwarzer, Friedrich Schrandt, and Gerhard Ehret (2009). "The airborne multi-wavelength water vapor differential absorption lidar WALES: system design and performance". In: *Applied Physics B* 96.1, p. 201 (cit. on p. 81).

Wonka, Peter, Michael Wimmer, François Sillion, and William Ribarsky (2003). *Instant architecture*. Vol. 22. 3. ACM (cit. on p. 116).

Woodhouse, Iain H, Caroline Nichol, Peter Sinclair, Jim Jack, Felix Morsdorf, Tim J Malthus, and Genevieve Patenaude (2011). "A multispectral canopy LiDAR demonstrator project". In: *IEEE Geoscience and Remote Sensing Letters* 8.5, pp. 839–843 (cit. on p. 81).

Wu, Bichen, Alvin Wan, Xiangyu Yue, and Kurt Keutzer (2018). "Squeezeseg: Convolutional neural nets with recurrent crf for real-time road-object segmentation from 3d lidar point cloud". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1887–1893 (cit. on p. 193).

Wu, S-T and MERCEDES ROCio GONZALES Marquez (2003). "A non-self-intersection Douglas-Peucker algorithm". In: *16th Brazilian symposium on computer graphics and Image Processing (SIBGRAPI 2003)*. IEEE, pp. 60–66 (cit. on p. 147).

Wulfmeyer, Volker and Jens Bösenberg (1998). "Ground-based differential absorption lidar for water-vapor profiling: assessment of accuracy, resolution, and meteorological applications". In: *Applied Optics* 37.18, pp. 3825–3844 (cit. on p. 85).

Xiao, Wen, Bruno Vallet, and Nicolas Paparoditis (2013). "Change detection in 3D point clouds acquired by a mobile mapping system". In: *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences* 1.2, pp. 331–336 (cit. on pp. 39, 97).

Xiao, Zhaoxia and Wenming Huang (2009). "Kd-tree based nonuniform simplification of 3D point cloud". In: *2009 Third International Conference on Genetic and Evolutionary Computing*. IEEE, pp. 339–342 (cit. on p. 146).

Xie, Hui, Jianning Wang, Jing Hua, Hong Qin, and Arie Kaufman (2003). "Piecewise C1 continuous surface reconstruction of noisy point clouds via local implicit quadric regression". In: *Proceedings of the 14th IEEE Visualization 2003 (VIS'03)*. IEEE Computer Society, p. 13 (cit. on p. 117).

Xing, X-F, Mir Abolfazl Mostafavi, Geoffrey Edwards, and Nouri Sabo (2019). "AN IMPROVED AUTOMATIC POINTWISE SEMANTIC SEGMENTATION OF A 3D URBAN SCENE FROM MOBILE TERRESTRIAL AND AIRBORNE LIDAR POINT CLOUDS: A MACHINE LEARNING APPROACH." In: *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 4 (cit. on p. 169).

Xu, Bo, Wanshou Jiang, Jie Shan, Jing Zhang, and Lelin Li (2015). "Investigation on the weighted ransac approaches for building roof plane segmentation from lidar point clouds". In: *Remote Sensing* 8.1, p. 5 (cit. on p. 118).

Xu, Katie, Yasuhiro Yao, Kazuhiko Murasaki, Shingo Ando, and Atsushi Sagata (2019). "Semantic Segmentation of Sparsely Annotated 3D Point Clouds by Pseudo-Labelling". In: *2019 International Conference on 3D Vision (3DV)*. IEEE, pp. 463–471 (cit. on p. 168).

Xu, Yusheng, Wei Yao, Ludwig Hoegner, and Uwe Stilla (2017). "Segmentation of building roofs from airborne LiDAR point clouds using robust voxel-based region growing". In: *Remote Sensing Letters* 8.11, pp. 1062–1071 (cit. on p. 119).

Yang, J and JF Coughlin (2014). "In-vehicle technology for self-driving cars: Advantages and challenges for aging drivers". In: *International Journal of Automotive Technology* 15.2, pp. 333–340 (cit. on p. 88).

Yastikli, Naci (2007). "Documentation of cultural heritage using digital photogrammetry and laser scanning". In: *Journal of Cultural Heritage* 8.4, pp. 423–427 (cit. on p. 62).

Yilmaz, Erdal and Gürcan Buyuksalih (2015). "3D Lidar Mapping of Underground Metro System". In: (cit. on p. 193).

Yu, Qian, Petra Helmholz, and David Belton (2016). "EVALUATION OF MODEL RECOGNITION FOR GRAMMAR-BASED AUTOMATIC 3D BUILDING MODEL RECONSTRUCTION." In: *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences* 41 (cit. on p. 120).

Yvinec, Mariette (2020). "2D Triangulation". In: *CGAL User and Reference Manual*. 5.0.2. CGAL Editorial Board (cit. on p. 160).

Zeng, Huayi, Jiaye Wu, and Yasutaka Furukawa (2018). "Neural procedural reconstruction for residential buildings". In: *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 737–753 (cit. on p. 117).

Zhang, Feihu, Daniel Clarke, and Alois Knoll (2014). "Vehicle detection based on LiDAR and camera fusion". In: *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, pp. 1620–1625 (cit. on p. 78).

Zhang, Wende (2010). "Lidar-based road and road-edge detection". In: *2010 IEEE Intelligent Vehicles Symposium*. IEEE, pp. 845–848 (cit. on p. 88).

Zhang, Zhengyou (2012). "Microsoft kinect sensor and its effect". In: *IEEE multimedia* 19.2, pp. 4–10 (cit. on p. 63).

Zheng, Linwei, Yilong Zhu, Bohuan Xue, Ming Liu, and Rui Fan (2019). "Low-cost GPS-aided lidar state estimation and map building". In: *arXiv preprint arXiv:1910.12731* (cit. on p. 77).

Zhou, Guoqing, C Song, John Simmers, and Penggen Cheng (2004). "Urban 3D GIS from LiDAR and digital aerial images". In: *Computers & Geosciences* 30.4, pp. 345–353 (cit. on p. 157).

Zhou, Qian-Yi (2012). *3D urban modeling from city-scale aerial LiDAR data*. University of Southern California (cit. on p. 87).

Zhou, Yin and Oncel Tuzel (2018). "Voxelnet: End-to-end learning for point cloud based 3d object detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4490–4499 (cit. on p. 119).

Zhu, Lingjie, Shuhan Shen, Xiang Gao, and Zhanyi Hu (2018). "Large Scale Urban Scene Modeling from MVS Meshes". In: *ECCV*, pp. 614–629 (cit. on p. 150).

Zhu, Lingjie, Shuhan Shen, Lihua Hu, and Zhanyi Hu (2017). "Variational building modeling from urban MVS meshes". In: *3DV*, pp. 318–326 (cit. on p. 150).

Zhu, Quanwen, Long Chen, Qingquan Li, Ming Li, Andreas Nüchter, and Jian Wang (2012). "3d lidar point cloud based intersection recognition for autonomous driving". In: *2012 IEEE Intelligent Vehicles Symposium*. IEEE, pp. 456–461 (cit. on p. 88).

Zolkos, SG, SJ Goetz, and R Dubayah (2013). "A meta-analysis of terrestrial aboveground biomass estimation using lidar remote sensing". In: *Remote Sensing of Environment* 128, pp. 289–298 (cit. on p. 86).

Zollhöfer, Michael, Matthias Nießner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, et al. (2014). "Real-time non-rigid reconstruction using an RGB-D camera". In: *ACM Transactions on Graphics (ToG)* 33.4, p. 156 (cit. on p. 94).

Zou, Chuhang, Ersin Yumer, Jimei Yang, Duygu Ceylan, and Derek Hoiem (2017). "3d-prnn: Generating shape primitives with recurrent neural networks". In: *Proceedings of the IEEE International Conference on Computer Vision*, pp. 900–909 (cit. on p. 118).