LABORATOIRE D'INFORMATIQUE
GASPARD MONGE

Sous la co-tutelle de :
CNRS
ÉCOLE DES PONTS PARISTECH
ESIEE PARIS
UPEM • UNIVERSITÉ PARIS-EST MARNE-LA-VALLÉE

École Doctorale Paris-Est
Mathématiques & Sciences et Technologies
de l'Information et de la Communication

Thèse de doctorat
de l'UNIVERSITÉ PARIS EST

Domaine : Traitement du Signal et des Images

présentée par **Alexandre Boulch**

pour obtenir le grade de
Docteur de l'UNIVERSITÉ PARIS EST

Reconstruction automatique de maquettes numériques.

Soutenue publiquement le 19 décembre 2014 devant le jury composé de :

Pierre Alliez	INRIA Sophia-Antipolis – Méditerranée	Rapporteur
Jan Boehm	University College London (UCL)	Examinateur
Martin De La Gorce	École des Ponts ParisTech	Examinateur
Reinhard Klein	Universität Bonn	Examinateur
Bruno Levy	INRIA Nancy Grand-Est	Rapporteur
Renaud Marlet	École des Ponts ParisTech	Directeur de Thèse
Jean-Philippe Pons	Acute3D	Examinateur

École des Ponts ParisTech
LIGM-IMAGINE
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77455 Marne-la-Vallée cedex 2
France

Université Paris-Est Marne-la-Vallée
École Doctorale Paris-Est MSTIC
Département Études Doctorales
6, Av Blaise Pascal - Cité Descartes
Champs-sur-Marne
77454 Marne-la-Vallée cedex 2
France

Remerciements

En premier lieu, je souhaite remercier Renaud Marlet pour m'avoir dirigé et fait confiance pendant ces trois années. Ses conseils et ses connaissances m'ont permis de découvrir, d'apprendre et de progresser sur le plan scientifique et humain. Merci pour le temps passé à discuter, me guider, à imaginer et à travailler sur les articles, quelle que soit l'heure du jour (et parfois de la nuit).

Je voudrais ensuite remercier mon jury de thèse. Merci à Pierre Alliez et Bruno Lévy pour avoir accepté d'être mes rapporteurs. Merci à Jan Boehm, Reinhard Klein, Jean-Philippe Pons et Martin De La Gorce pour leur participation à mon jury en tant qu'examinateurs. Merci à eux pour leur enthousiasme et leurs questions et remarques constructives et instructives lors de la soutenance.

Merci à toute l'équipe d'Imagine qui m'a accueilli pendant mes travaux de thèse. Merci à Pascal Monasse pour m'avoir fait confiance et m'avoir donné la possibilité de donner des cours durant trois ans à l'École des Ponts. Merci à Arnak Dalalyan pour son aide précieuse en statistique. Merci à Olivier Tournaire, Jean-Yves Audibert, Bertrand Neveu, Guillaume Obozinski, Nikos Komodakis pour les discussions et les conseils prodigués lors de pauses cafés ou d'intrusions inopinées dans leur bureau. Ensuite, je tiens à remercier l'ensemble des doctorants de l'équipe : ceux qui ont soutenu avant moi, Olivier, David, Victoria et Pierre (sans qui la veille technologique et la distribution de jeux gratuits au sein de l'équipe n'auraient pas été les mêmes) ; un grand merci aussi à ceux qui me suivront sur les planches : Zhe (google code, wii et drone), Marina (top chef), Raghudeep (cinéma et presque adversaire au tennis), Mateusz (ça a été un plaisir de partager le bureau 413), Loïc (les maths, un jour je comprendrai), Yohann (successeur de Pierre sur la diffusion de jeux chronophages), Francisco (premier spot de conseils sur plein de domaines), et je n'oublie pas Amine, Spyros, Laura, Sergey et Maria sans qui l'ambiance dans les couloirs du laboratoire, à la cafétéria et dans les cinémas de Paris ne serait pas la même.

Toujours à l'École des Ponts, je remercie le laboratoire Navier pour mon adoption à visée sportive et caféinée. Merci Max (breizh power), François (guitare et basse), Maged (spoilers et téléphone) pour les séances de basket puis de tennis. Merci Patrick, Denis, Camille et Ghazi pour m'avoir presque donné envie de passer à la mécanique.

Toujours à Navier, je remercie Mathilde (gardienne des clés) et Jérémy (science infuse), amis indéfectibles aux Ponts et en dehors. Merci pour les aventures vécues pendant trois ans, ski (grumff sera a tout jamais une demande de pâte à tartiner à la noisette), panne sur le périph (c'est gratuit), soirées nombreuses et variées...

Merci aussi à Anaïs (soirées dessin), Sophie (jeux et théâtre), Alexis (nouvelles technologies), Adrien (soirée et sport tout en cuisse), Marion (animaux et bio), Jean

(Papy et physique quantique), Thomas (Boz et bugs), Virto (rognon et vodka violette) et Jean-Noël (fromage et judo).

Merci à Mathieu (basket et MMORPG) et Marie (arts, culture, cuisine...) pour les repas, les soirées (jusqu'au bout de la nuit).

Un grand merci à la coloc originelle de Glacière : Vincent (cf la suite), Janus et Janette (power ball et judo) et Xavier (frites et bières) : trois années mémorables qui resteront pour moi une expérience unique. Merci Vincent, pour la musique (Sexy sushi, fatal bazooka...), pour les soirées dont je ne me rappelle plus très bien le contenu, les footings où j'essayais de m'accrocher et je m'arrête là, avant d'en dire trop.

Merci Aude (cocktails et dynamisme), Tom (vin et style) et Sylvain (écologie et fun), branche parisienne de la bande finistérienne, pour les moments passés ensemble. Merci à toute la bande, Pouich (il a un prénom, mais on ne le connais pas), Mamie (cf remarque précédente), Nath (fête de la musique, oh yeah), Adrien (grand), Cycy (mes bras s'en souviennent encore), Jérôme et Marion (fruits et petits bébés), Flo et Becca (sorry my thesis is not written in English), Soiz (sports de montagne), Kris (en France, en Amérique du Sud...)... Merci Didi (handball, rando et mégane) pour les vacances et les journées dans les parcs d'attraction.

Je remercie maintenant ma famille, mon frère Maxime (force et honneur) et mes parents, Danielle (peinture et course à pied) et Hervé (moteurs et informatique ?) pour m'avoir soutenu et encouragé depuis le début. Je n'oublie pas non plus ma marraine qui a donné de son temps pour l'organisation du pot de thèse.

Enfin la personne sans qui cette aventure aurait été différente voire impossible : Hélène, qui connaît ma thèse sur le bout des doigts, bien que contre son plein gré ; Hélène que je ne remercierai jamais assez pour m'avoir supporté (dans tous les sens du terme) pendant ces trois ans. Merci pour tout.

Abstract

The interest for digital models in the building industry is growing rapidly. These centralize all the information concerning the building and facilitate communication between the players of construction: cost evaluation, physical simulations, virtual presentations, building lifecycle management, site supervision, etc. Although building models now tend to be used for large projects of new constructions, there is no such models for existing building. In particular, old buildings do not enjoy digital 3D model and information whereas they would benefit the most from them, e.g., to plan cost-effective renovation that achieves good thermal performance. Such 3D models are reconstructed from the real building.

Lately a number of automatic reconstruction methods have been developed either from laser or photogrammetric data. Lasers are precise and produce dense point clouds. Their price have greatly reduced in the past few years, making them affordable for industries. Photogrammetry, often less precise and failing in uniform regions (e.g. bare walls), is a lot cheaper than the lasers. However most approaches only reconstruct a surface from point clouds, not a semantically rich building model. A building information model is the alliance of a geometry and a semantics for the scene elements.

The main objective of this thesis is to define a framework for digital model production regarding both geometry and semantic, using point clouds as an entry.

The reconstruction process is divided in four parts, gradually enriching information, from the points to the final digital mockup.

First, we define a normal estimator for unstructured point clouds based on a robust Hough transform. It allows to estimate accurate normals, even near sharp edges and corners, and deals with the anisotropy inherent to laser scans.

Then, primitives such as planes are extracted from the point cloud. To avoid oversegmentation issues, we develop a general and robust statistical criterion for shape merging. It only requires a distance function from points to shapes.

A piecewise-planar surface is then reconstructed. Planes hypothesis for visible and hidden parts of the scene are inserted in a 3D plane arrangement. Cells of the arrangement are labelled full or empty using a new regularization on corner count and edge length. A linear formulation allow us to efficiently solve this labelling problem with a continuous relaxation.

Finally, we propose an approach based on constrained attribute grammars for 3D model semantization. This method is entirely bottom-up. We prevent the possible combinatorial explosion by introducing maximal operators and an order on variable instantiation.

Résumé

La maquette numérique de bâtiment est un outil nouveau et en plein essor dans les métiers de la construction. Elle centralise les informations et facilite la communication entre les acteurs : évaluation des coûts, simulations physiques, présentations virtuelles, suivis de travaux, etc. Si une maquette numérique est désormais utilisée pour les grands chantiers de bâtiments nouveaux, il n'en existe pas en revanche pour la plupart des bâtiments déjà construits. Or, avec le vieillissement du parc immobilier et le développement du marché de la rénovation, la maquette numérique serait une aide considérable pour des bâtiments anciens.

Des techniques de reconstruction plus ou moins automatique ont été développées ces dernières années, à base de mesures laser ou de photogrammétrie. Les lasers, précis et denses, sont chers mais restent abordables pour les industriels, tandis que la photogrammétrie, souvent moins précise et moins fiable dans les zones uniformes (p.ex. les murs), est beaucoup plus bon marché. Mais la plupart des approches s'arrêtent à la reconstruction de surfaces, sans produire de maquettes numériques. À la géométrie doit cependant s'ajouter des informations sémantiques décrivant les éléments de la scène.

L'objectif de cette thèse est de fournir un cadre de reconstruction de maquettes numériques, à la fois en ce qui concerne la géométrie et la sémantique, à partir de nuages de points.

Pour cela, plusieurs étapes sont proposées. Elles s'inscrivent dans un processus d'enrichissement des données, depuis les points jusqu'à la maquette numérique finale.

Dans un premier temps, un estimateur de normales pour les nuages de points est défini. Basé sur une transformée de Hough robuste, il permet de retrouver correctement les normales, y compris dans les zones anguleuses et s'adapte à l'anisotropie des données.

Dans un second temps, des primitives géométriques sont extraites du nuage de points avec leur normales. Afin d'identifier les primitives identiques existantes en cas de sur-segmentation, nous développons un critère statistique robuste et général pour l'identification de formes, ne requérant qu'une fonction distance entre points et formes.

Ensuite, une surface planaire par morceaux est reconstruite. Des hypothèses de plans pour les zones visibles et les parties cachées sont générées et insérées dans un arrangement. La surface est extraite avec une nouvelle régularisation sur le nombre de coins et la longueur des arêtes. L'utilisation d'une formulation linéaire permet, après relaxation continue, d'extraire efficacement une surface proche de l'optimum.

Enfin, nous proposons une approche basée sur des grammaires attribuées avec contraintes pour l'enrichissement sémantique de modèles 3D. Cette méthode est *bottom-up* : nous partons des données pour construire des objets de complexité croissante. La possible explosion combinatoire est gérée efficacement via l'introduction d'opérateurs maximaux et d'un ordre pour l'instanciation des variables.

Table des matières

Introduction

Chapitre 1

Maquettes numériques de bâtiments

Sommaire

1.1 Maquettes numériques et enjeux du BIM

1.1.1 Le BTP

Le secteur du bâtiment et des travaux publics (BTP) est acteur majeur de l'économie française. En 2011, plus de 1.3 millions de personnes travaillent pour une entreprise du BTP [1]. Les postes sont très variés, des métiers de la maçonnerie à ceux de l'administration en passant par les différents bureaux d'études et d'architectes. Cette variété dans les métiers se retrouve aussi dans le profil des entreprises. Si des entreprises comme Bouygues ou Vinci sont mondialement connues, et emploient près de 300 000 personnes, la grande majorité des entreprises sont de petites entreprises : 93% des 498 000 entreprises de BTP françaises ont moins de 10 salariés.

Le BTP est un secteur où la concurrence est rude. Pour être en position favorable sur un marché, il faut être capable de fournir rapidement une estimation des coûts et des délais de réalisation. La précision de cette estimation est cruciale, une surévaluation se soldant par une perte de marché, et une sous-évaluation par un déficit sur le projet. La vitesse de réalisation des ouvrages est aussi un critère important pour les clients. Ceci implique la mobilisation d'une grande quantité de main d'œuvre, de différents métiers sur un même chantier. Dans ce cadre, la coordination de toutes les forces de production, de l'architecte au peintre, est d'une importance cruciale. C'est pour répondre à ce besoin que le BIM (*Building Information Model*), ou maquette numérique de bâtiment, a été développé.

1.1.2 Building Information Model

La construction ou la rénovation d'un bâtiment font appel à de nombreux corps de métiers, souvent d'entreprises différentes avec des modes de fonctionnement variés. Ces dernières années, le développement de l'informatique a transformé les modes de conception des bâtiments. Cependant les différents formats de données des logiciels dits métiers (logiciels spécialisés) tendent à rendre difficile la communication entre les acteurs du chantier. Pour faciliter les échanges, il est évident qu'un objet unique contenant toutes les informations du chantier est un avantage. En effet, si toutes les informations sont unifiées, les acteurs ont accès aux mêmes données. Les erreurs de retranscription sont alors plus rares et les malfaçons dues auparavant à un manque de communication sont traitées en amont lors d'étapes de conception plutôt que découvertes en cours de chantier.

De ce constat est né le Building Information Model (BIM). Le BIM d'un bâtiment est un concept numérique représentant le bâtiment. L'idée est de concentrer un maximum d'informations sur le bâtiment à l'intérieur d'un unique document, de sa structure (poutres, murs...) aux réseaux (électriques, hydrauliques, téléphoniques...) qui le composent. Le BIM est aussi appelé maquette numérique ou simplement Maquette.

Outre les formats des logiciels propriétaires, les IFC (*Industry Foundation Classes*) sont aujourd'hui le format de référence pour le BIM. A l'initiative de Building Smart [2], les IFC sont un format libre. Ils font l'objet d'un développement actif, notamment de la

[1] Données chiffrées de l'Observatoire des métiers du BTP, http://www.metiers-btp.fr/

[2] Building Smart, précédemment International Alliance for Interoperability, http://www.buildingsmart.org

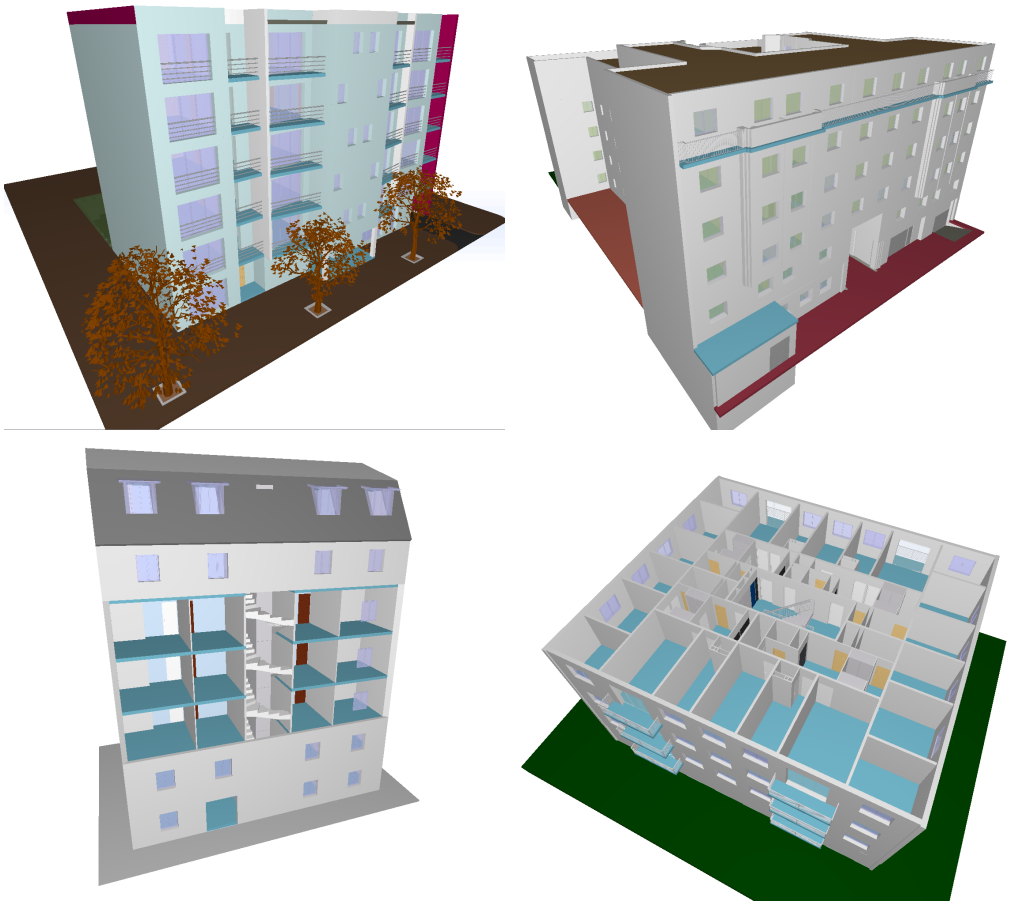
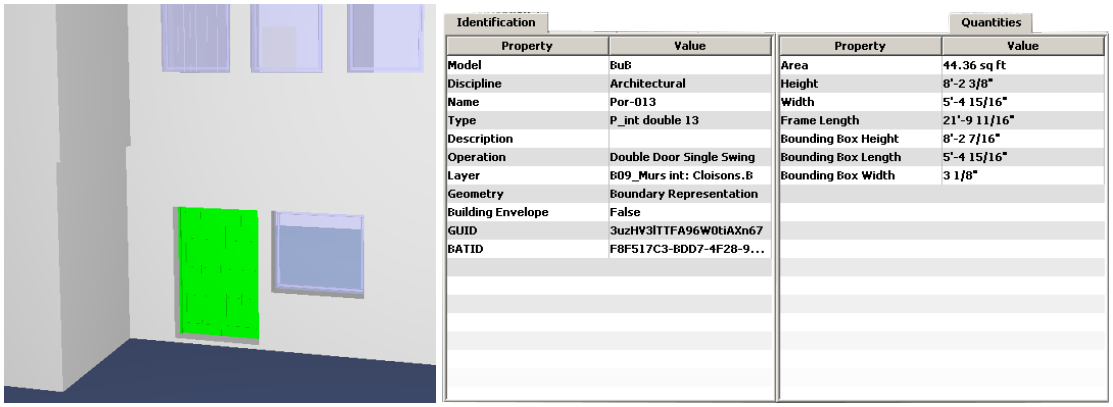


FIGURE 1.1 – Exemples de maquettes numériques de bâtiments.

part d'institutions telles que le Centre Scientifique et Technique du Bâtiment (CSTB) [3], en vue d'étendre leurs applications et la richesse de leur représentation.

Bien qu'une grande majorité des chantiers soit encore traitée de manière conventionnelle, avec des plans papiers, le BIM se développe notamment sur les projets d'envergure, où la somme des plans rend la gestion difficile et sujette à erreurs. Les avantages du BIM sont multiples. Son format numérique en fait un objet modifiable et facilement diffusable. De plus comme le concept de BIM ne fixe pas a priori de limites aux types de données qu'il peut contenir, tous les corps de métiers peuvent y incorporer leurs données. Les simulations thermiques, acoustiques et les calculs de structures peuvent être effectués directement à partir du modèle. Il est ainsi plus facile de détecter les zones de conflits (canalisations traversant des conduits de ventilation...). Enfin, si on ajoute des informations temporelles, il est possible de coordonner les tâches de chantier voire de maintenance directement sur le BIM, et d'optimiser le cycle de vie du bâtiment. La figure 1.1 montre quatre exemples de BIM pour différents bâtiments. Sur les images, certaines parties ont été cachées afin de rendre compte de la construction intérieure du BIM. La figure 1.2, présente le détail d'une porte issue d'une maquette avec les informations qui lui sont rattachées.

[3] CSTB, http://www.cstb.fr/



Identification

Property	Value
Model	BuB
Discipline	Architectural
Name	Por-013
Type	P_int double 13
Description	
Operation	Double Door Single Swing
Layer	B09_Murs int: Cloisons.B
Geometry	Boundary Representation
Building Envelope	False
GUID	3uzHV3ITTFA96W0tiAXn67
BATID	F8F517C3-BDD7-4F28-9...

Quantities

Property	Value
Area	44.36 sq ft
Height	8'-2 3/8"
Width	5'-4 15/16"
Frame Length	21'-9 11/16"
Bounding Box Height	8'-2 7/16"
Bounding Box Length	5'-4 15/16"
Bounding Box Width	3 1/8"

FIGURE 1.2 – Une porte dans un bâtiment, et le détail de ses attributs.

1.1.3 BIM = géométrie + sémantique

Comme introduit dans la section précédente, le BIM contient toutes les informations nécessaires à la conception d'un bâtiment et à son exploitation.

Dans le présent document, on séparera généralement les informations sous deux catégories :

- **La géométrie**, décrivant la forme d'un objet : dimensions, angles, position...
- **La sémantique**, c'est à dire les informations complémentaires : fonction (mur, fenêtre, table...), matière, propriétés mécaniques ou thermiques, éventuelle référence produit...

La figure 1.3 illustre la décomposition d'une maquette numérique sur une géométrie simple et une sémantique limitée.

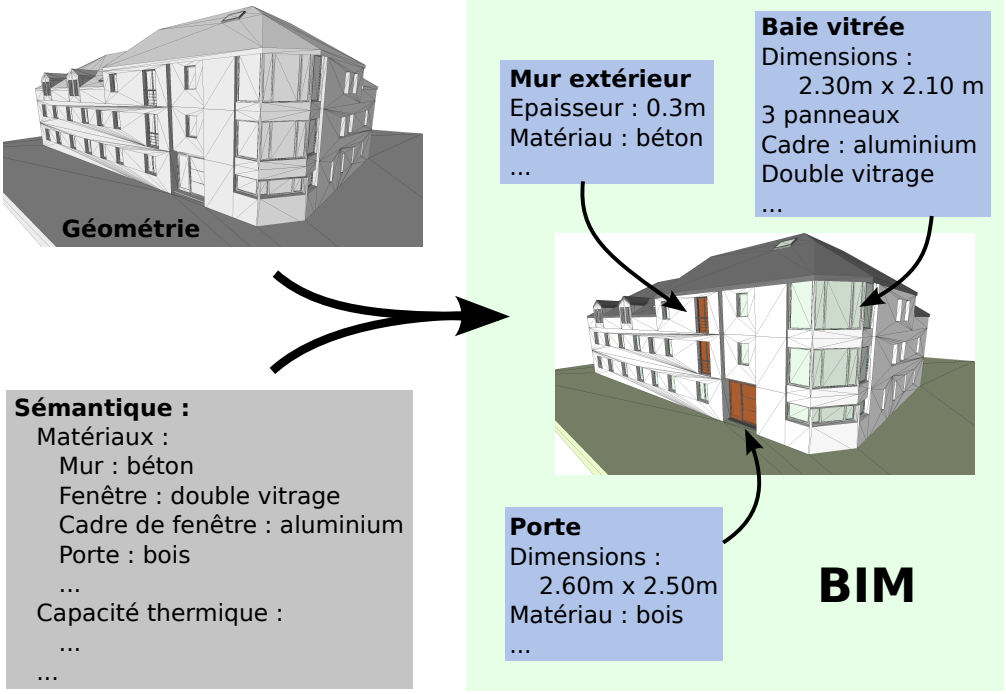


FIGURE 1.3 – Une maquette numérique est composée d'une géométrie et d'une sémantique.

1.2 Cas particulier de l'existant

Le marché de la rénovation est en pleine expansion. Les nouvelles normes, notamment thermiques, le vieillissement du parc immobilier ainsi que les incitations fiscales aux travaux d'amélioration sont favorables au développement de ce marché.

Si pour les nouveaux bâtiments, l'utilisation du BIM peut se faire depuis la phase de conception jusqu'à la réalisation et l'exploitation, la plupart des bâtiments existants ne disposent pas d'une maquette numérique et les seules données disponibles sont les plans de construction (lorsqu'ils existent et avec des degrés de fiabilité très variables) et le bâtiment lui-même. De plus, même lorsque les plans existent, ils ne reflètent pas toujours les travaux effectués dans le bâtiment au cours de sa vie.

Lors d'une rénovation, comme lors d'une construction, une maquette s'avère un outil important pour un gain de temps et de performance. Il faut donc être capable de construire une maquette à partir de l'existant, à peu de frais. Cette capacité est aussi un atout pour la construction neuve, où la création de la maquette en cours de construction permet de suivre le déroulement des travaux et la conformité entre la réalité et la maquette initiale.

Vouloir reconstruire à partir de l'existant implique de capturer les positions des différents objets dans l'espace. Pour ce faire, plusieurs systèmes sont aujourd'hui à notre disposition :

- **La photogrammétrie** : technique basée sur de multiples prises de photos. Les appareils photos étant aujourd'hui relativement abordables, la photogrammétrie est une technique peu coûteuse, mais qui demande une certaine maîtrise de la photographie et un très grand nombre de prises de vues.
- **Les capteurs de mouvement** : le plus connu est sans doute le Kinect de Microsoft. Le système est peu coûteux et simple d'utilisation, mais aujourd'hui peu précis et sensible aux conditions de prise de vue car le capteur infrarouge peut être aveuglé par la lumière solaire.
- **Le laser** : les lasers sont des outils de grande précision qui permettent de scanner de grandes surfaces très rapidement et avec une très grande précision. Leur coût est beaucoup plus élevé que les appareils précédents, mais les récentes baisses de prix les rendent accessibles aux industriels.

Le chapitre 2 développe ces différents modes d'acquisition de données, leurs points communs et leurs spécificités.

En pratique, pour des opérations de reconstruction, c'est aujourd'hui le laser qui a la préférence des industriels. Des entreprises proposent aujourd'hui un service de création de maquettes numériques pour l'existant à partir de relevés laser.

1.3 Reconstruction automatique

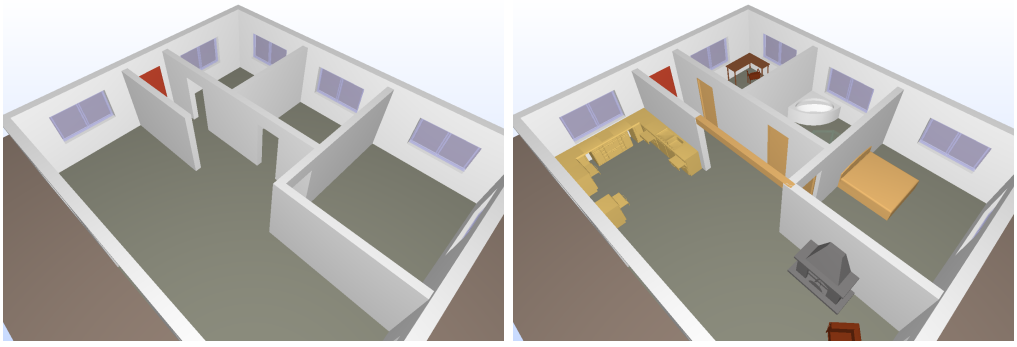
Actuellement les entreprises proposant des services de reconstruction de maquettes numériques pour de l'existant utilisent généralement des données issues de laser.

Une visite chez l'un de ces prestataires a permis d'apprécier le niveau de détail et la précision des bâtiments reconstruits. Il apparaît que le processus de création de la maquette sémantisée est manuel, long et fastidieux. En effet, reconstruire le bâtiment revient à le redessiner, dans ses moindres détails, en accord avec les données acquises sur le terrain. Les outils utilisés sont manuels ou semi-automatiques, et laissent une grande latitude au reconstructeur.

1.3.1 Niveau de détail

Les prestataires pour la reconstruction de maquette recréent un objet proche du BIM pour les construction neuves : géométrie très fiable et sémantique poussée. Cependant le niveau de détail de la maquette fournie n'est pas toujours en adéquation avec les applications pour lesquelles elle a été créée. Le niveau de précision requis varie du tout au tout. Même si *"qui peut le plus peut le moins"*, créer une maquette extrêmement détaillée uniquement pour calculer un diagnostic de performance énergétique qui ne prend principalement en compte que les surfaces vitrées, les surfaces extérieures et leurs matériaux et le mode de chauffage, est une perte de temps. De même, la volonté de faire un rendu graphique (pré-projet, jeux vidéos...) ne nécessite souvent pas un niveau de détail très haut, l'impression de réalisme étant donnée principalement par la lumière et les textures plaquées sur le modèle. D'un autre côté, des simulations acoustiques ou d'accessibilité aux personnes handicapées vont nécessiter de connaître avec précision l'ensemble des pièces, la largeur des portes voire même la hauteur du mobilier ou des serrures.

Il est évident que le niveau de détail de la maquette doit correspondre à celui de la tâche la plus exigeante. La figure 1.4 illustre les différents niveaux de précision nécessaires suivant l'application.



À gauche, uniquement la structure générale, les fenêtres et les portes, pour une simulation thermique. À droite, le mobilier est modélisé pour simuler, par exemple, l'accessibilité.

FIGURE 1.4 – Deux niveaux de détail pour une même maquette [KIT, site].

1.3.2 Automatisation

De nombreuses constructions répondent à des standards aisément identifiables : immeubles de bureaux, logements collectifs ou individuels. Leur époque de construction comme les matériaux utilisés imposent des codes qui se retrouvent dans l'architecture de ces bâtiments. Reconstruire une à une toutes les pièces d'une barre d'immeubles des années 1980 est répétitif et coûteux en temps, surtout lorsque celles-ci sont des formes géométriques simples et qu'elles sont toutes identiques.

Par exemple, une grande quantité de bâtiments de bureaux répondent à ce que l'on appelle l'hypothèse du monde Manhattan. Sous cette hypothèse, on considère qu'un bâtiment est constitué de pièces dont tous les murs sont planaires et orientés uniquement selon deux directions orthogonales. La figure 1.5 présente un ensemble de pièces suivant les règles du monde Manhattan.

Plus les tâches sont répétitives et les géométries sont simples, plus il est naturel de vouloir les automatiser. Cela vaut pour la géométrie mais aussi pour la sémantique. Si

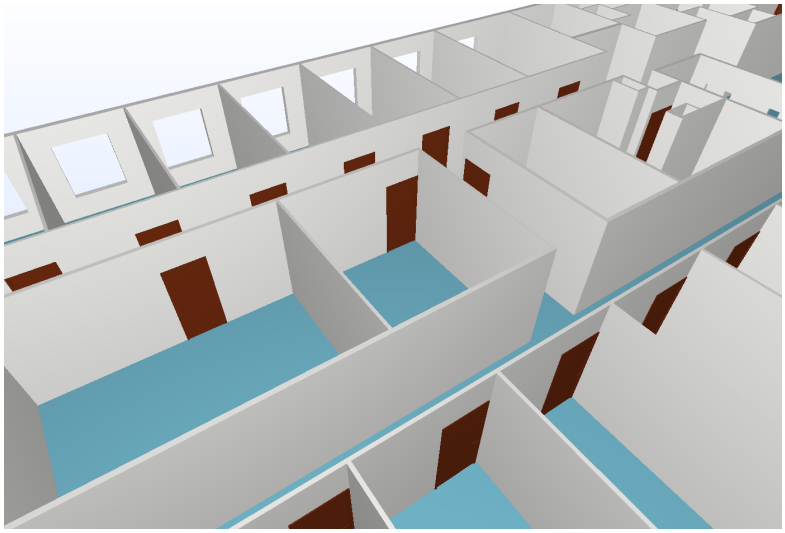


FIGURE 1.5 – Vue d'une maquette répondant aux règles du monde Manhattan.

discerner les propriétés physiques ou le matériau demande des connaissances supplémentaires, la détection des classes d'objets comme les murs ou les sols doit pouvoir être réalisée automatiquement.

De même, il est concevable que pour certaines tâches une main experte puisse être nécessaire. Ce serait particulièrement le cas pour des reconstructions d'une précision extrême, ne répondant pas à des règles particulières, statues, frises gravées, éléments très déformés... Mais les cas nécessitant cette précision sont rares et la reconstruction automatique peut donc répondre à une grande quantité de besoins.

Chapitre 2

Le matériel d'acquisition / Les données

Sommaire

La reconstruction de maquettes numériques se base sur des nuages de points 3D. Il existe différents dispositifs et techniques pour acquérir des points dans l'espace. L'objectif de ce chapitre est de présenter les dispositifs et méthodes d'acquisition de nuages de points ainsi que les particularités des nuages qui en sont issus.

2.1 Photogrammétrie

La photogrammétrie est une technique utilisant des photographies pour retrouver la position 3D des points du modèle. Peu coûteuse, elle a fait l'objet d'une étude poussée dans la littérature. Ses résultats, dépendants de la qualité des photographies, peuvent être arbitrairement précis.

La figure 2.1 montre le principe de la reconstruction 3D par stéréo-photogrammétrie, reconstruction à l'aide de deux images.

Soit p (respectivement p') un point de l'image I (resp. I'). p est aussi un point 3D dans le plan image de la caméra de centre optique C. Alors le point P de l'objet réel, dont p est la projection sur l'image, est sur le rayon $\overrightarrow{Cp}$.

Si maintenant p' est aussi l'image de P, mais cette fois-ci dans l'image I', alors P est à l'intersection de $[C, p)$ et de $[C', p')$.

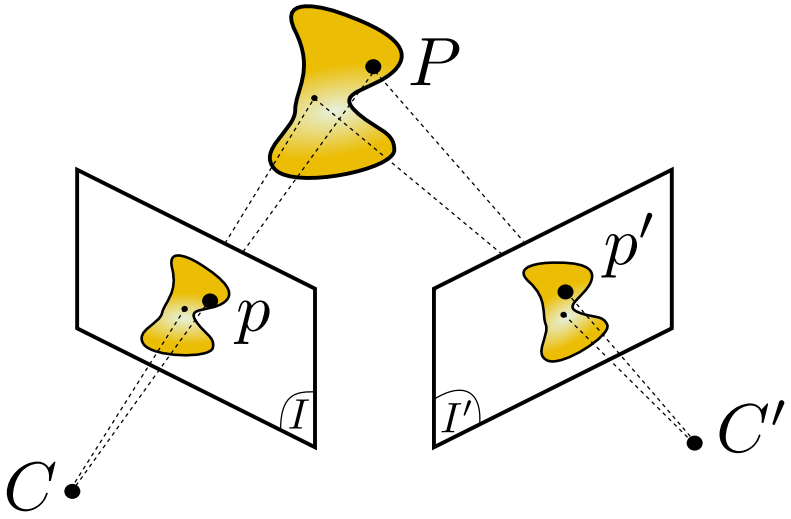


FIGURE 2.1 – Principe de reconstruction par stéréo photogrammétrie.

Les difficultés de la reconstruction multi-vues sont nombreuses. Parmi elles, on compte :

- la recherche de correspondances fiables entre points saillants détectés dans l'image,
- l'estimation des paramètres internes des caméras,
- l'estimation robuste et précise des transformations liant les placements relatifs des caméras.

La technique se base sur la recherche de points d'intérêt dans les images, ce sont ceux-ci qui vont être mis en correspondance. La qualité de la reconstruction est donc liée à la capacité à trouver des points d'intérêt dans les images et à identifier les points identiques d'une image à l'autre. Ces points servent à calibrer le système, ils sont en général peu nombreux. Le nuage de points final est un nuage densifié. La transformation

entre les caméras étant connue, de nouvelles correspondances entre points dans les images sont recherchées. Cependant l'identification de points dans les zones uniformes est très difficile. Il en résulte un nuage de densité très disparate, manquant crucialement de points dans les zones uniformes telles que les murs dans les scènes d'intérieur. C'est, par exemple le cas sur la figure 2.2 où très peu de points sont présents sur les zones peu texturées, comme le tableau ou le plafond.

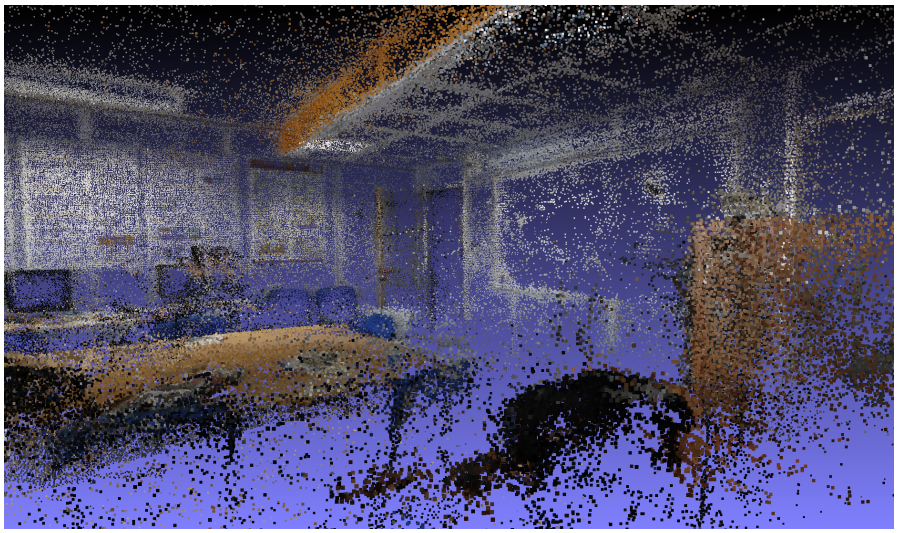


FIGURE 2.2 – Exemple de nuage de points photogrammétrique pour une salle de réunion.

2.2 Capteurs de profondeur

Les capteurs de profondeur, dont le plus connu est sans doute le Kinect de Microsoft, sont des appareils qui détectent la distance du capteur aux objets placés devant lui. Cette donnée de profondeur est en général associée à une information de couleur.

Le Kinect utilise une lumière structurée infrarouge pour estimer une image de profondeur. Un motif connu est projeté sur la scène, ensuite c'est l'observation des déformations de ce motif qui permet de récupérer l'information de profondeur. Cette projection est faite dans l'infrarouge pour ne pas être dépendante des conditions lumineuses et colorimétriques de la scène. La figure 2.3 illustre ce fonctionnement.

Le format est une image RGBD (Red Green Blue Depth, soit couleur + profondeur). L'information de couleur est en fait ajoutée à l'information de profondeur via une caméra supplémentaire placée sur l'appareil.

Contrairement à la photogrammétrie, un tel capteur est efficace même dans les zones uniformes, ce qui est en fait un outil utilisable pour la reconstruction de scènes d'intérieur. La figure 2.4 présente un nuage de points acquis via Kinect, c'est l'une des scènes du dépôt NYU,[Silberman et al., 2012].

Alors que la précision de la photogrammétrie dépend de la qualité des images, les données issues de capteurs de profondeur sont à précision fixe. Pour le Kinect, l'erreur est de l'ordre de quelques centimètres. Cette erreur est accentuée par la discrétisation interne des valeurs de profondeur. La figure 2.5 montre cet effet de discrétisation. Pour

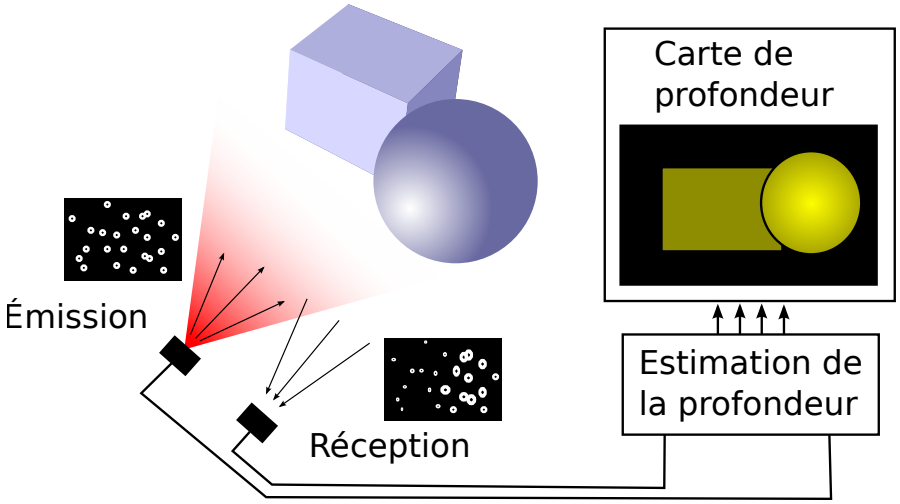


FIGURE 2.3 – Principe de fonctionnement du Kinect.

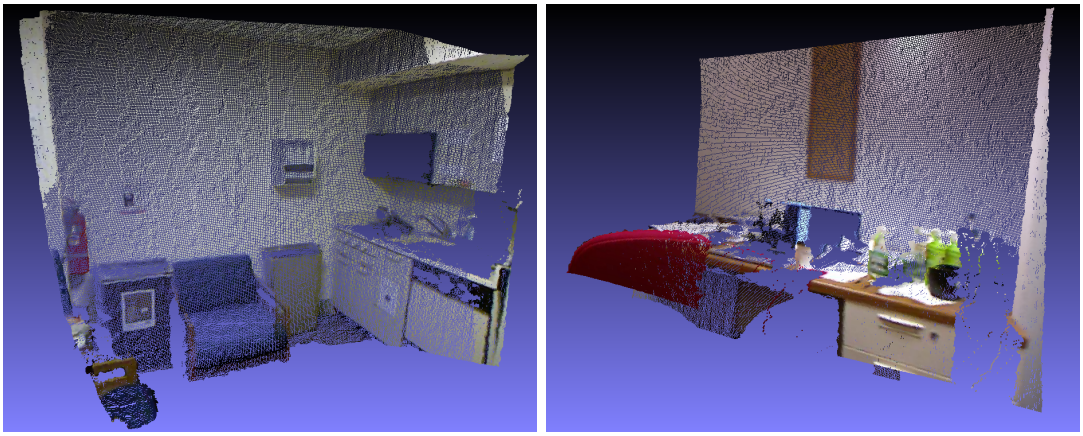


FIGURE 2.4 – Exemple de nuages de points acquis via Kinect.

pallier cet inconvénient, il est nécessaire de multiplier les acquisitions de différents points de vue afin d'obtenir une image moyenne.

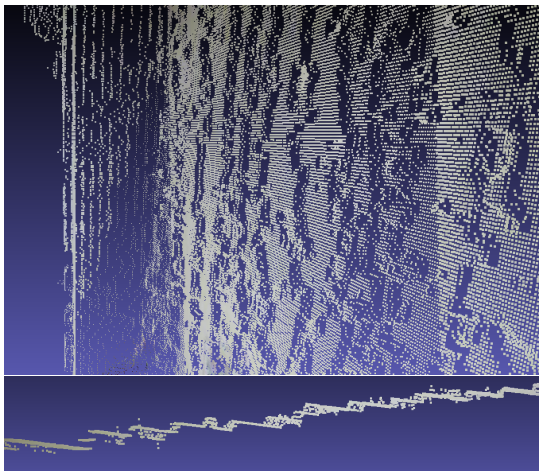


FIGURE 2.5 – Effet de discrétisation du Kinect : exemple d'un mur. En bas, vue en coupe.

Enfin, à une précision modeste s'ajoutent des décalages entre les informations colori-

métriques et les informations de positions. (figure 2.6)

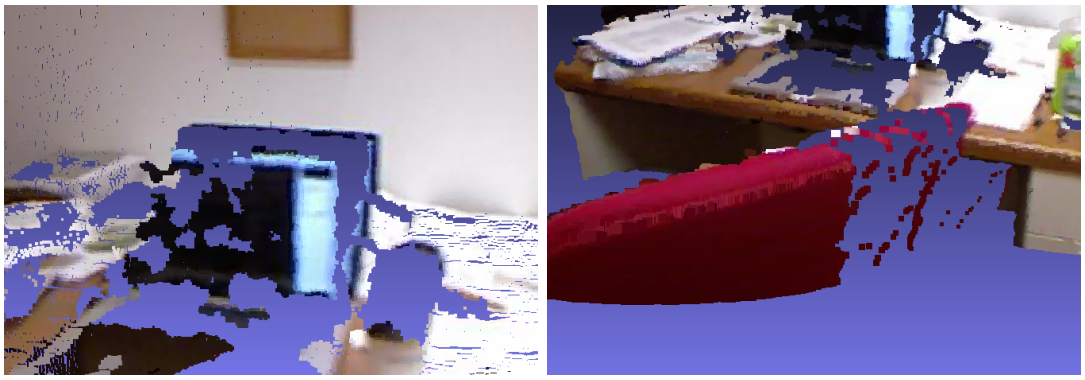


FIGURE 2.6 – Exemple de décalage entre couleurs et géométrie. Les couleurs de l'écran se projettent sur le mur, celles de la chaise sur la table.

2.3 Laser

Le laser est l'appareil le plus précis mais aussi le plus onéreux. Cependant, ces dernières années ont vu le prix des scanners laser décroître fortement, ils sont désormais accessibles à des industriels et des entreprises dont le corps de métiers n'est pas directement la création de maquettes numériques.

2.3.1 Principe du laser

Les lasers utilisés dans le cadre de la télémétrie utilisent principalement deux types de technologie, le temps de vol et le décalage de phase.

Temps de vol

Un laser mesurant par temps de vol envoie une pulsation lumineuse vers l'objet à mesurer. La vitesse de la lumière dans l'air étant connue, la distance est calculée à partir du temps mis par la lumière pour faire l'aller retour entre le laser et l'objet. La figure 2.7 illustre ce principe.

Décalage de phase

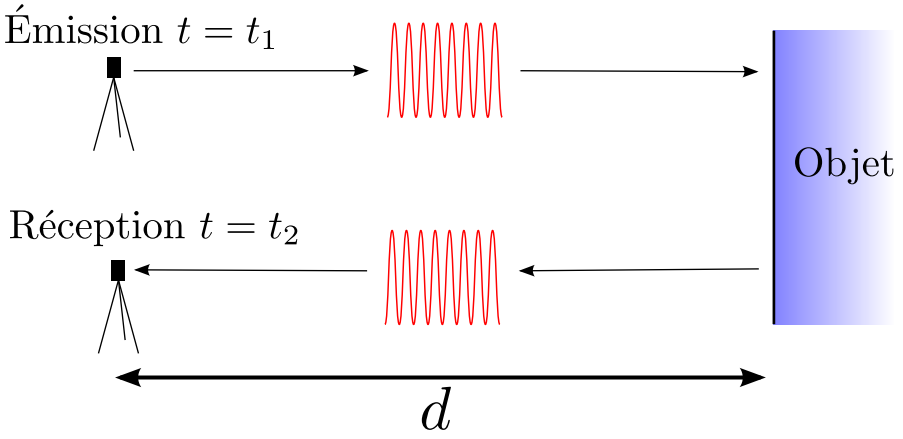
Un laser à décalage de phase émet une onde lumineuse dont le flux lumineux est modulé sinusoïdalement. Celle-ci se réfléchit sur l'objet et on mesure le décalage entre les phases des deux signaux (figure 2.8).

Supposant que le signal d'émission est de la forme $A_e \sin(\omega\, t)$ alors le signal de retour est : $A_r \sin(\omega\, (t + dt))$. En multipliant les signaux, on a un nouveau signal :

$$\frac{A_e A_r}{2} \left(\cos(\omega\, dt) - cos(2\omega\, t + \omega\, dt)\right)$$

En utilisant un filtre passe bas, on récupère la composante constante $\cos(\omega\, dt)$ dont on extrait le temps de vol dt puis la distance entre l'émetteur et l'objet.

Les lasers à décalage de phase sont en général plus précis que ceux qui utilisent le temps de vol mais leur portée est moindre.



La vitesse de propagation de l'onde étant connue, la distance d est déduite du temps $t_2 - t_1$.

FIGURE 2.7 – Principe de fonctionnement de la mesure par temps de vol.

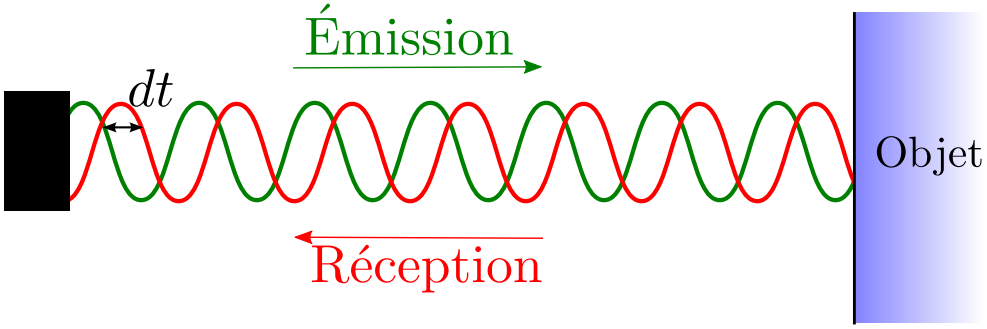


FIGURE 2.8 – Principe de fonctionnement de la mesure par décalage de phase.

2.3.2 Les données

Les données laser sont très précises, de l'ordre du millimètre pour une acquisition à 10 mètres. Les lasers récents sont capables d'acquérir plusieurs millions de points en seulement quelques minutes. Ils sont généralement placés sur des trépieds et l'acquisition est effectuée depuis une position ou un ensemble de positions statiques. Contrairement à la photogrammétrie et aux capteurs de profondeur, il est difficile de déplacer le laser dans de multiples positions pour capturer tous les recoins d'une scène.

Acquisition sphérique

Les données laser utilisées dans cette thèse sont issues de laser à décalage de phase, dont l'acquisition est sphérique. En coordonnées sphériques, le laser effectue une rotation autour de l'axe vertical avec un pas constant Δ_θ. Pour chaque longitude, il effectue une rotation en colatitude à pas constant Δ_ϕ, de la verticale jusqu'à un angle donné, définissant un cône d'invisibilité sous le scanner. La présence de ce cône d'invisibilité sous le scanner est due à des contraintes physiques, d'encombrement de l'appareil. La figure 2.9 illustre ce fonctionnement.

Les pas constants en longitude et en colatitude produisent une géométrie d'acquisition spécifique, une image laser. Dans cette image, chaque pixel représente une direction, sa valeur étant les coordonnées du point touché. Ce format est aisément transformable en une image de profondeur, en calculant la norme du vecteur liant l'origine du scanner au

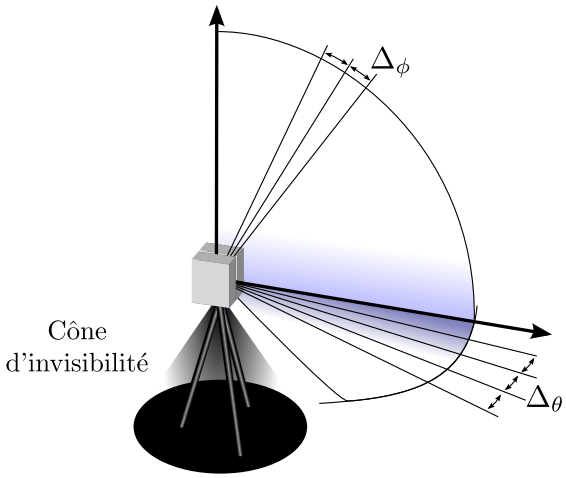
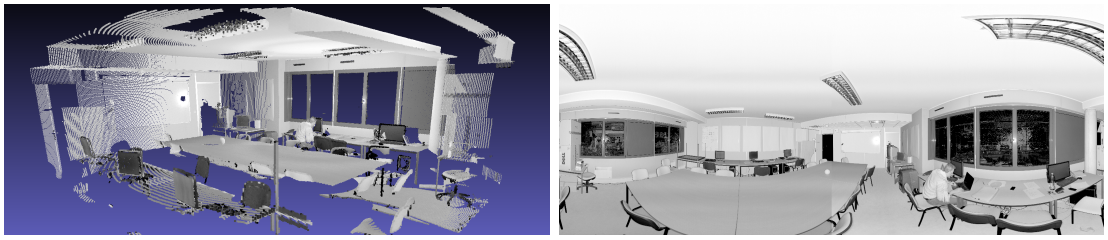


FIGURE 2.9 – Mode d'acquisition d'un laser fixe sur trépied.

point scanné. Cette image est une projection de la sphère sur un plan 2D, les lignes droites y sont déformées. La figure 2.10a montre un nuage de points laser et la figure 2.10b, l'image laser correspondante. Dans l'image, le niveau de gris représente l'absorption de la surface. Ce format structuré est un avantage dans le traitement des données, il est aisé de récupérer les voisins de chaque point dans le nuage de points en utilisant la connectivité des pixels. C'est un des avantages sur les nuages issus de photogrammétrie, où l'accès aux voisins est effectué via des structures de données particulières telles que les kd-trees.



(a) Nuage de points. (b) Image laser.

FIGURE 2.10 – Acquisition laser d'une salle de réunion.

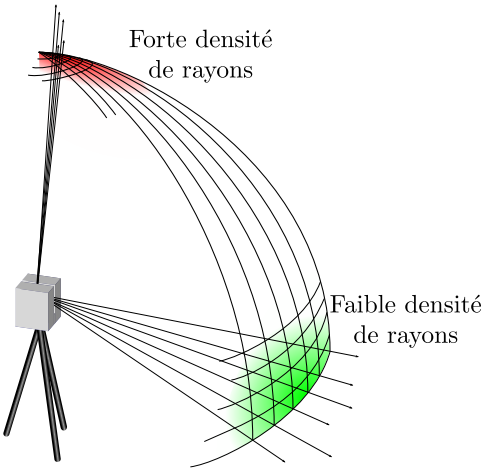
Anisotropie

Le mode d'acquisition sphérique par balayage présente une grande disparité de densités de rayons. Au niveau du pôle, la densité de rayons lancés est de plusieurs ordres de grandeur plus grande que celle à l'équateur. La figure 2.11 montre cette variation de densité, avec un schéma et des zones colorées en fonction de leur densité (figure 2.11a) et les zones correspondantes sur une scène laser(figure2.11b).

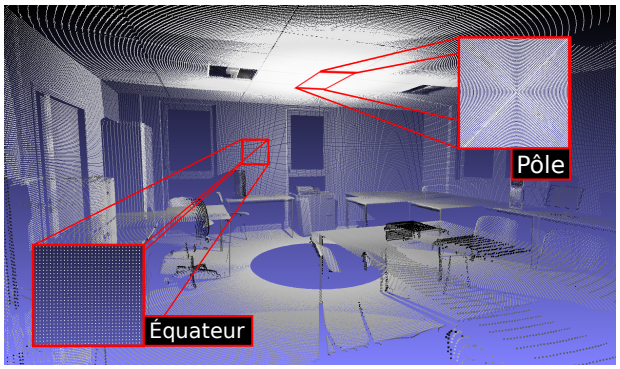
Pour des pas de discrétisation Δ_θ et Δ_ϕ constants, la densité de points par stéradian (unité de surface sur la sphère unité), $w_{sphérique}$ est donnée par la relation suivante :

$$w_{sphérique}(\mathbf{x}) = \Delta_\theta \Delta_\phi \sin(\phi) \tag{2.1}$$

où x est le point de coordonnées sphériques (r, θ, ϕ).



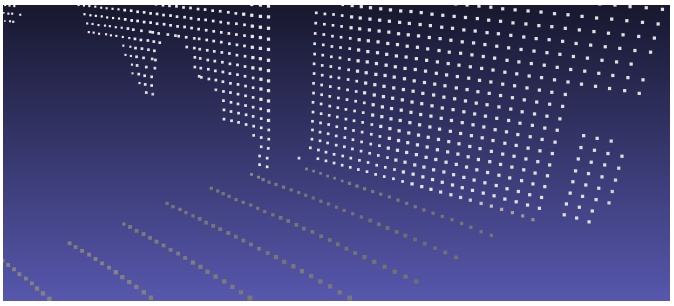
(a) Différence de densité entre pôle et équateur.



(b) Scène laser.

FIGURE 2.11 – Variations de densité entre le pôle et l'équateur.

Cette variation de densité dans l'acquisition ne suffit pas à expliquer la forte anisotropie dans les nuages de points laser. À cette anisotropie globale (variation d'échantillonnage) s'ajoute une anisotropie locale : dans le voisinage d'un point la densité de points varie suivant la direction d'observation. La figure 2.12 montre ce phénomène sur un nuage laser. L'angle d'incidence $\psi \in [0, \pi/2]$, du laser sur la surface, joue un rôle important. Lorsque ψ est faible, la surface associée au point est faible (figure 2.13 à gauche). A l'inverse lorsque ψ est grand, l'aire de la surface associée se déforme et est plus importante (figure 2.13 à droite). De plus, l'aire est d'autant plus importante que la surface touchée est loin de l'origine O du scanner.



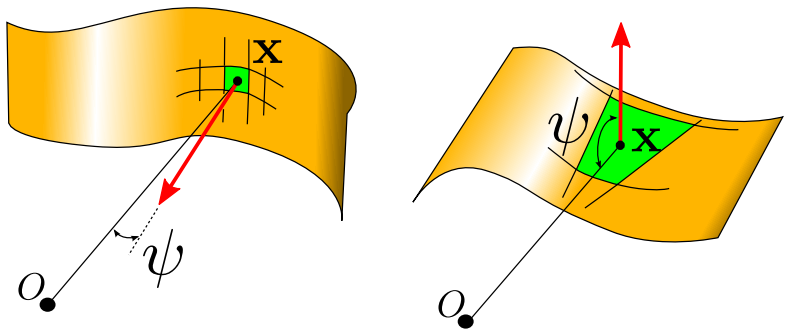
Le laser frappe le mur (en blanc) quasi-orthogonalement tandis qu'il touche le sol (en gris) tangentiellement.

FIGURE 2.12 – Illustration sur un nuage de point laser de l'anisotropie due à l'incidence du laser.

L'angle ψ est accessible via le calcul de la normale à la surface (partie I). L'aire associée au point $\mathbf{x}$, notée $w_{aniso}(\mathbf{x})$, est donnée par la relation suivante :

$$w_{aniso}(\mathbf{x}) = d^2 \Delta_\theta \Delta_\phi \frac{\sin(\phi)}{\cos\psi} \tag{2.2}$$

où d est la distance de $\mathbf{x}$ à l'origine du scanner et ψ, l'angle d'incidence en $\mathbf{x}$.



L'angle ψ est l'angle entre la surface au point touché par le rayon et ce rayon. La surface verte représente la surface associée à ce point.

FIGURE 2.13 – Surface associée à un point en fonction de l'orientation locale de la surface.

Chapitre 3

Processus de reconstruction

Sommaire

Les travaux de cette thèse se focalisent sur la reconstruction automatique de maquettes numériques de bâtiments existants. Les données en entrées sont des nuages de points recalés, c'est-à-dire positionnés les uns par rapport aux autres. Des informations supplémentaires telles que les plans des bâtiments ne sont pas considérées ici.

3.1 Vers l'abstraction

Un ensemble de points 3D, aussi précis soit-il, ne constitue pas une maquette. Les points sont des informations de très bas niveau : trois coordonnées dans l'espace (avec potentiellement un ou plusieurs points de vue). Ils ne contiennent pas d'information sémantique et l'information géométrique qu'ils représentent est très limitée. Un point est un échantillon sur une surface. Par conséquent, en faisant abstraction du bruit dans les données, la seule chose que l'on sait est que la surface voulue passe par ce point.

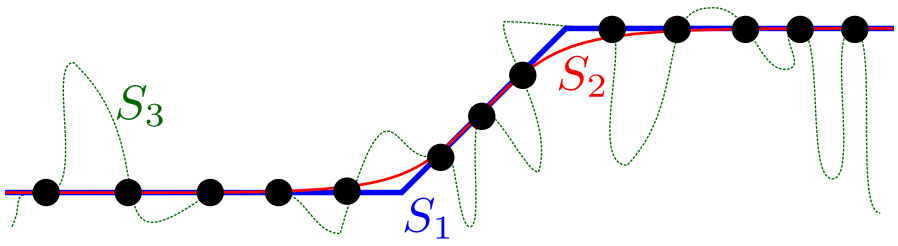


FIGURE 3.1 – Echantillonnage

En ce qui concerne la géométrie, notre but est de reconstruire une surface à partir de points. Il n'existe pas une unique surface correspondant à un ensemble de points. Il faut donc faire des hypothèses sur la surface, pour restreindre l'ensemble des surfaces possibles et être à même de choisir la surface la plus probable.

Par exemple, la figure 3.1 présente trois surfaces S_1, S_2 et S_3 correspondant au même nuage de points. Si on décide de faire une reconstruction planaire par morceaux, le choix logique sera la surface S_1. Imposer une surface régulière lisse (sans angles) pourra mener au choix de la surface S_2.

Et ici encore, une surface n'est pas une maquette, il faut la comprendre. Comme précédemment, en utilisant des a priori forts sur la surface, comme le fait qu'elle représente un bâtiment, il est possible d'enrichir la géométrie avec des données sémantiques.

L'idée générale de cette thèse est de partir des points (information bas niveau), de créer une surface et de l'enrichir avec de nouvelles informations afin de produire une maquette numérique complète (informations haut niveau).

3.2 Pipeline de reconstruction

Les travaux présentés dans ce document s'inscrivent dans une logique d'enrichissement progressif des données de bas niveau, les points. La figure 3.2 présente les étapes de la reconstruction :

- l'estimation de normales,
- l'extraction de primitives géométriques,

- la recherche de la surface,
- l'enrichissement de la surface avec des données sémantiques.

Les quatre blocs se situent à quatre niveaux d'abstraction différents. La recherche de normales s'effectue au niveau des points. La normale est une information locale, propre à chaque point. Les primitives sont quant à elles un regroupement de points en un ensemble cohérent. La surface, au niveau d'abstraction supérieur, est un maillage basé sur les primitives. Enfin la maquette numérique, au dessus de la géométrie, est la géométrie associée à des données sémantiques.

Les quatre parties de cette thèse suivent les quatre blocs de la figure 3.2. Chaque partie est voulue indépendante des autres (hormis ses entrées et ses sorties), et les références d'une partie à l'autre sont limitées.

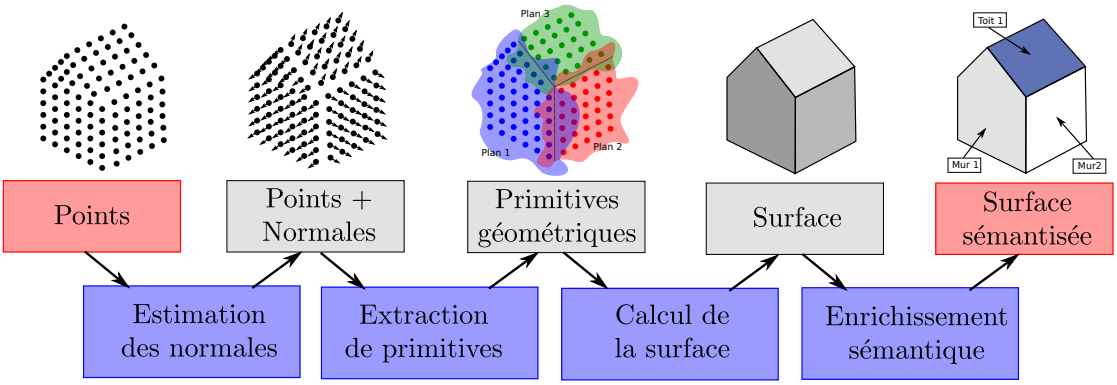


FIGURE 3.2 – Étapes du processus de reconstruction.

3.3 Présentation des parties

Note : Cette section présente succinctement les parties du manuscrit. Pour une meilleure lisibilité, l'état de l'art sera développé au sein de chaque partie.

3.3.1 Estimation de normales

La partie I est consacrée à l'estimation de normales pour des nuages de points.

L'estimation de normales est une opération de base pour la compréhension des nuages de points. Les normales donnent l'orientation locale de la surface sous-jacente. Cette tâche est un problème très étudié et de nombreuses méthodes ont été développées. Le bruit dans les données et les variations dans l'échantillonnage des surfaces sont des obstacles à une estimation précise. Évaluer correctement la normale implique une estimation continue sur les surfaces lisses tout en se prémunissant contre les effets de lissage au niveau des angles. De plus, cette estimation doit être effectuée en un temps de calcul minimal afin de passer à l'échelle des nuages de plusieurs millions, voire milliards, de points.

Les méthodes basées sur la régression [Cazals and Pouget, 2003; Hoppe et al., 1992], le diagramme de Voronoï [Dey and Goswami, 2006] ou le RANSAC [Li et al., 2010], sont des estimateurs de qualité sans pour autant remplir toutes ces exigences.

La méthode que nous proposons est une nouvelle approche. Elle prend pour base une transformée de Hough robuste (Chapitre 4). La distribution des normales possibles est estimée en remplissant un accumulateur (tableau), la normale retenue est le maximum de cette distribution (chapitre 5). L'exploration des normales possibles est régie par des bornes statistiques robustes, et les effets liés à la discrétisation sont gérés en ajoutant de l'aléatoire sur l'orientation des accumulateurs. L'algorithme est capable de retrouver correctement la normale des points dans les zones anguleuses et s'accommode de l'anisotropie locale et du bruit pouvant être présents dans les données.

La performance de cette méthode a été mesurée par différentes expériences, qui montrent qu'elle est au moins aussi précise et robuste au bruit que l'état de l'art tout en étant plus rapide. Ce compromis peut être modifié, des variantes sont proposées. Au prix d'un temps de calcul un peu plus long, l'anisotropie des voisinages des points peut aussi être gérée.

La figure 3.3 montre l'estimation des normales au niveau d'une zone anguleuse pour deux estimateurs différents. A gauche, une régression linéaire tend à produire une normale lissée. A droite, la méthode proposée ici retrouve les normales attendues, y compris pour les points proches de l'angle.

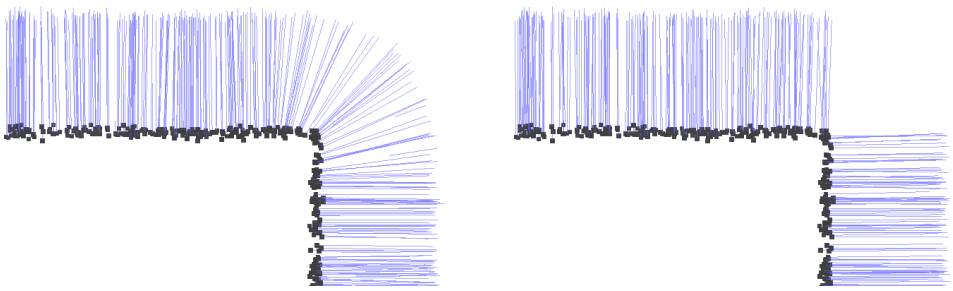


FIGURE 3.3 – Deux comportements possibles d'estimateurs de normales au niveau d'une zone anguleuse : à gauche, régression linéaire, à droite, notre méthode.

3.3.2 Recherche de primitives géométriques

Les nuages de points sont parfois très volumineux et il est nécessaire de regrouper ceux-ci en ensembles cohérents qui vont aussi servir de base à la constitution de surfaces complexes. Des primitives géométriques simples offrent l'avantage d'être faciles à manipuler. La recherche de primitives dans les nuages de points est un problème classique en géométrie algorithmique. Une grande quantité de méthodes ont été développées pour extraire diverses primitives (lignes, cercles, ellipses... en 2D ; plans, cylindres, sphères ... en 3D). Cette partie est décomposée en deux chapitres : l'un consacré à la reconnaissance de primitives, l'autre à la fusion de primitives identiques (figure 3.4).

Le chapitre 6 présente le principe de deux méthodes parmi les plus utilisées pour résoudre ce problème :

- RANdom SAmple Consensus (RANSAC) [Schnabel et al., 2007]
- La croissance de régions [Bughin et al., 2010; Chauve et al., 2010]

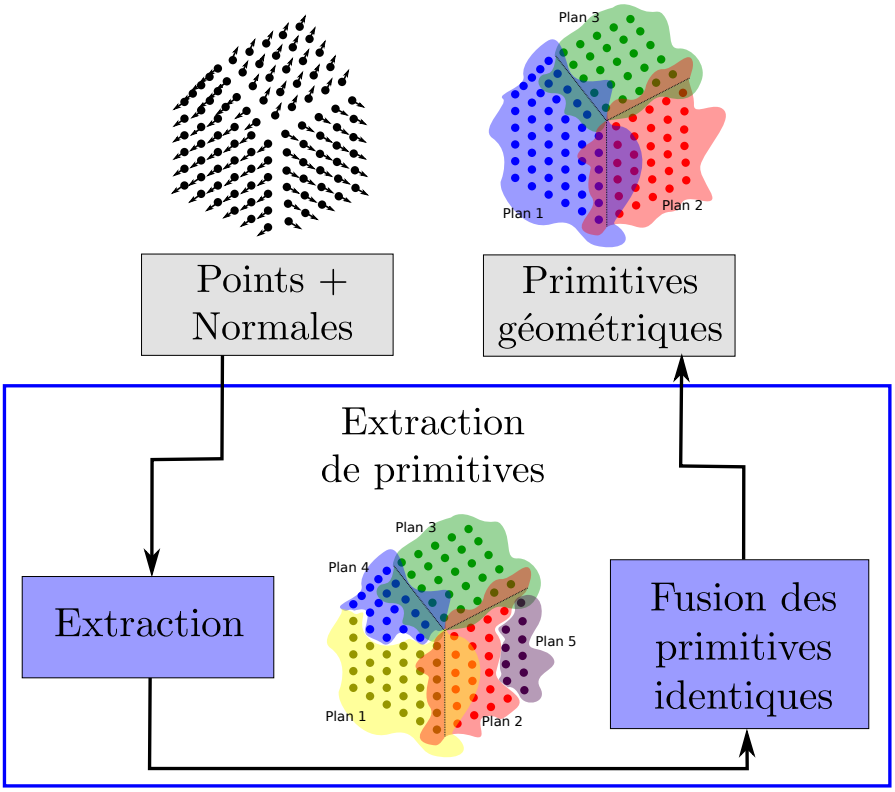


FIGURE 3.4 – extraction de primitives, fusion de primitives

Dans un second temps, le chapitre 7 présente une méthode originale pour la fusion de primitives géométriques. En effet, les algorithmes d'extraction de primitives sont sujets à des erreurs telles que la sur-segmentation (trop de primitives extraites). Deux critères statistiques sont proposés afin de décider si deux primitives peuvent être considérées comme identiques (figure 3.5). L'un est basé sur le test de Kolmogorov-Smirnov pour la comparaison de deux distributions, l'autre sur le test de Mann-Whitney. L'inégalité de Dvoretzky-Keifer-Wolfowitz permet de calculer une borne inférieure sur la taille minimale de l'échantillon nécessaire pour obtenir une précision donnée, ce qui améliore drastiquement le temps de calcul.

Une des forces de cette méthode est qu'elle ne nécessite pas de connaissance approfondie des surfaces à comparer, uniquement une capacité à calculer la distance d'un point à la surface. L'espace des surfaces à comparer ne se limite pas à des primitives géométriques simples, mais à n'importe quelles formes ou maillages pour lesquels le calcul des distances est accessible.

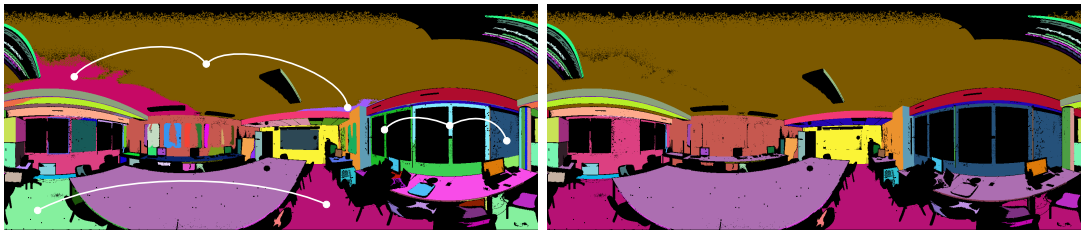


FIGURE 3.5 – Exemple de fusion de primitives sur une scène laser. À gauche, la scène sur-segmentée après croissance de régions et à droite, la scène après fusion des plans identiques.

3.3.3 Reconstruction de surface

La partie III est consacrée à la reconstruction de la surface planaire par morceaux, basée sur les plans préalablement détectés dans le nuage. Du point de vue de la maquette numérique, ce choix s'explique par le fait que les environnements créés par l'homme sont majoritairement planaires par morceaux, ou constitués de formes géométriques pouvant être approximées par un ensemble de plans.

De nombreux algorithmes de reconstruction ont été développés pour estimer une surface à partir d'un ensemble de points. La reconstruction de Poisson [Kazhdan et al., 2006] fait partie des algorithmes visant à créer une surface sans simplification. Ici notre objectif est l'idéalisation de la surface : nous recherchons une surface simple, planaire par morceaux. Le but et la méthode sont proches de ceux proposés par Chauve et al. [2010].

L'algorithme que nous proposons fournit une surface étanche, orientable et sans bords qui correspond aux observations. C'est un maillage polygonal reconstruit à partir des plans visibles de la scène. Avec pour objectif de reconstruire une surface réaliste, y compris dans les parties invisibles de la scène, des hypothèses de plans supplémentaires sont générées. L'extraction de la surface est ensuite posée sous la forme d'un problème d'optimisation discrète. Trois critères de régularisation sont proposés : une minimisation de l'aire de la surface reconstruite, hypothèse déjà utilisée dans la littérature, mais aussi la minimisation de la longueur des arêtes et celle du nombre de coins. Ces deux derniers critères se traduisent par des potentiels d'ordre supérieur à deux. Nous introduisons une formulation linéaire pour ce problème. La résolution du problème est rendue possible grâce à une relaxation continue, afin de le traiter comme un problème linéaire à variables réelles. La solution calculée est alors une solution approchée, mais qui expérimentalement est de bonne qualité. Au travers des expérimentations effectuées sur des intérieurs de bâtiments, la longueur des arêtes et le nombre de coins s'avèrent de meilleurs critères de minimisation que l'aire.

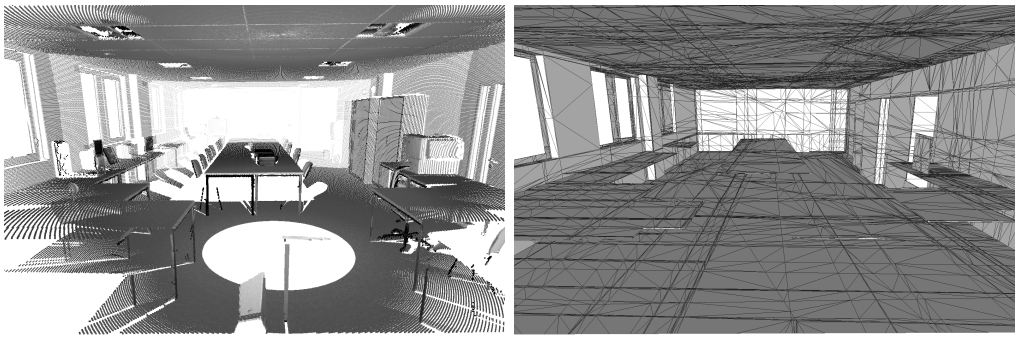


FIGURE 3.6 – Exemple de reconstruction de surface. À gauche le nuage de points, à droite la géométrie reconstruite.

3.3.4 Création de la sémantique

Pour l'ajout de données sémantiques sur un modèle géométrique, la partie IV propose une nouvelle approche basée sur les grammaires attribuées avec contraintes.

La reconnaissance d'éléments dans une scène complexe est en plein essor depuis quelques années. Les grammaires sont des outils utiles pour décrire les scènes avec une structure

hiérarchique, comme les façades en 2D [Teboul et al., 2010] ou les temples doriques en stéréo multi-vues [Mathias et al., 2011].

Ces grammaires allient une description hiérarchique de la scène avec des contraintes d'assemblage. L'objectif est de partir d'un certain nombre de règles simples, dictées par le contexte (bâtiment de type Haussmanien, bureaux...) et de reconnaître dans le modèle les instances de chaque élément décrit par ces règles. Les résultats sont donnés sous la forme d'un arbre décrivant la hiérarchie constructive du bâtiment. Pour ne pas s'exposer à une explosion combinatoire, les opérateurs maximaux sont définis et utilisés pour réduire l'espace d'exploration. Il est possible d'envisager plusieurs interprétations pour une même scène, toutes les interprétations sont calculées de concert, sous la forme d'une forêt (ensemble d'arbres). Les expériences montrent que des structures complexes comme des escaliers de formes variables sont reconnues.

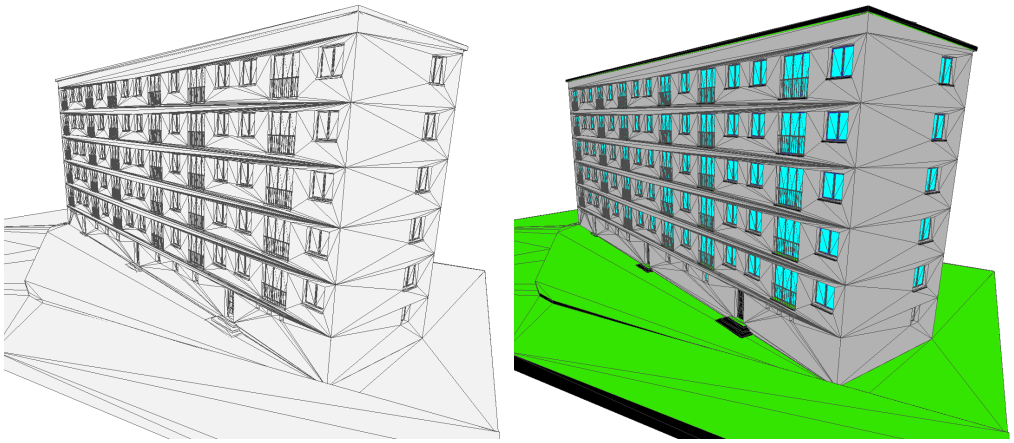


FIGURE 3.7 – Exemple d'enrichissement sémantique sur un modèle géométrique de bâtiment. À gauche, la géométrie non sémantisée (soupe de polygones) et à droite, le modèle sémantisé automatiquement, avec une couleur par type d'élément.

Bibliographie

Bughin, E., Almansa, A., Grompone von Gioi, R., and Tendero, Y. (2010). Fast plane detection in disparity maps. In *ICIP*, pages 2961–2964.

Cazals, F. and Pouget, M. (2003). Estimating Differential Quantities using Polynomial fitting of Osculating Jets. In *Symposium on Geometry Processing*, pages 177–187.

Chauve, A.-L., Labatut, P., and Pons, J.-P. (2010). Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *CVPR*, pages 1261–1268.

Dey, T. K. and Goswami, S. (2006). Provable surface reconstruction from noisy samples. *Comput. Geom.*, 35(1-2):124–141.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. A., and Stuetzle, W. (1992). Surface reconstruction from unorganized points. In *SIGGRAPH*, pages 71–78.

Kazhdan, M. M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Symposium on Geometry Processing*, pages 61–70.

KIT (s.i.t.e.). Karlsruhe Institute of Technology, FZK-House. http://www.iai.fzk.de/www-extern/index.php?id=1177&L=1.

Li, B., Schnabel, R., Klein, R., Cheng, Z.-Q., Dang, G., and Jin, S. (2010). Robust normal estimation for point clouds with sharp features. *Computers & Graphics*, 34(2):94–106.

Mathias, M., Martinovic, A., Weissenberg, J., and Gool, L. V. (2011). Procedural 3D Building Reconstruction Using Shape Grammars and Detectors. In *IEEE International Conference 3DIMPVT*, pages 304–311.

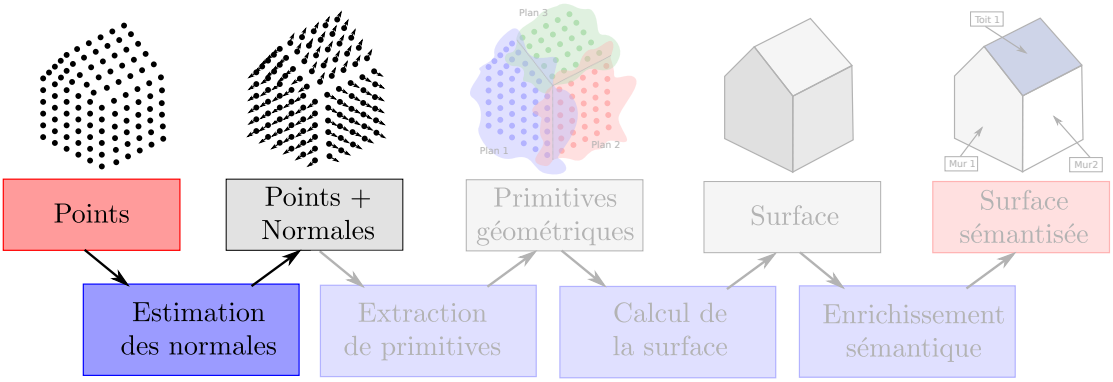
Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum*, 26(2):214–226.

Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor Segmentation and Support Inference from RGBD Images. In *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part V*, pages 746–760.

Teboul, O., Simon, L., Koutsourakis, P., and Paragios, N. (2010). Segmentation of building facades using procedural shape priors. In *CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pages 3105–3112.

Première partie

Transformée de Hough robuste et estimation de normales



Chapitre 4

Transformée de Hough aléatoire et robuste

4.1 Introduction

La reconnaissance de formes est un problème classique en vision par ordinateur. En 2D (images) comme en 3D (nuages de points), la compréhension d'une scène à haut niveau (objets, structures...) impose souvent l'extraction de primitives géométriques simples, par exemple, des lignes, des cercles ou des ellipses dans les images, des plans, des cylindres ou des sphères dans les nuages de points. Par exemple, pour rechercher un mur dans un nuage de points, une approche est d'en extraire en premier lieu les zones planes puis de choisir parmi celles-ci les bonnes candidates (taille, verticalité...). De même, en chimie, pour vérifier la présence ou non de bulles dans une solution transparente, l'extraction de cercles et d'ellipses dans des photos est une possibilité.

Avec le développement des moyens d'acquisition 3D (laser, Kinect...) et l'utilisation de techniques de photogrammétrie toujours plus poussées, les images et les nuages de points à traiter sont toujours plus volumineux. Aujourd'hui, traiter plusieurs millions de points est une tâche commune. Pour être efficace, il est nécessaire de condenser ces données. Une manière d'y parvenir est d'extraire des formes géométriques simples et de remplacer les points concernés par ces formes, codées par un faible nombre de paramètres. De plus la représentation des nuages de points par un ensemble de primitives géométriques facilite souvent un travail ultérieur visant à retrouver des objets ou des structures particulières.

Dans un premier temps, ce chapitre présente brièvement les différentes méthodes de reconnaissance de primitives géométriques avant de se focaliser plus particulièrement sur la transformée de Hough [Hough, 1962]. La partie II revient sur le cas plus particulier de la recherche de primitives dans un nuage de points en trois dimensions.

La suite de ce chapitre est une extension de la transformée de Hough aléatoire (Randomized Hough Transform, RHT) présentant une borne sur le nombre d'échantillons à utiliser pour la détection ainsi qu'un critère d'arrêt lorsqu'on recherche un nombre précis d'occurrences de la forme.

4.2 Méthodes de reconnaissance de formes

La détection de formes a beaucoup été étudiée ces dernières années. D'abord étudiée dans les images, elle s'est ensuite généralisée dans des espaces de plus grandes dimensions.

Certaines des méthodes évoquées ici seront détaillées dans le cas de l'extraction de surfaces en trois dimensions dans la partie II.

RANSAC L'une des méthodes les plus utilisées est sans-doute le Random SAmple Consensus (RANSAC), introduite par Fischler and Bolles [1981]. Le principe est de sélectionner des hypothèses de formes et de les vérifier avec les données si l'hypothèse est justifiée. Si celle-ci est retenue, alors les éléments associés à cette forme sont mis de côté et on réitère l'algorithme sur les données restantes. Le RANSAC a, depuis son introduction, bénéficié de multiples améliorations et spécialisations. Il a au départ été utilisé pour extraire des cylindres dans des images de profondeurs [Fischler and Bolles, 1981], et plus tard dans des nuages de points [Chaperon and Goulette, 2001]. En 1993, Roth and Levine [1993] ont proposé un algorithme pour des formes variées. Le RANSAC de Schnabel et al. [2007] permet l'extraction robuste et rapide de primitives géométriques dans des nuages de points non structurés en trois dimensions. Des améliorations ont aussi été proposées pour améliorer la robustesse du RANSAC, au travers de méthodes comme MSAC [Torr and Zisserman, 1998], MLESAC [Torr and Zisserman, 2000], ou encore la vitesse d'exécution [Nistér, 2005].

Méthodes a contrario Les méthodes a contrario ont aussi été utilisées pour la détection de segments dans les images [von Gioi et al., 2008, 2010] ou, plus récemment [Akinlar and Topal, 2011]. Bughin [2011] a proposé une détection de plans dans des images de profondeur basée sur un critère a contrario. L'idée des méthodes a contrario est d'estimer l'hypothèse selon laquelle la configuration des données a été générée par du bruit. Si la configuration a peu de chances d'être due au bruit, on considère alors avoir découvert une structure digne d'intérêt.

J-linkage Toldo and Fusiello [2008] ont proposé une méthode appelée J-linkage. Elle est construite sur une combinaison entre le RANSAC et la transformée de Hough (décrite pas la suite). Le J-linkage se base sur le tirage d'hypothèses à la manière d'un RANSAC et sur un changement d'espace (comme dans une transformée de Hough) afin de ne retenir que celles souhaitées. Au contraire d'un RANSAC qui agit de manière gloutonne (repérer un modèle, retirer les points lui correspondant et recommencer), le J-linkage permet de détecter plusieurs modèles à la fois.

Transformée de Hough La transformée de Hough, introduite par Hough [1962], a été d'abord utilisée pour détecter des lignes et des arcs dans les images. Dès [Duda and Hart, 1972], les accumulateurs sont utilisés pour représenter un espace de recherche discrétisé. La transformée de Hough s'accompagne d'un changement d'espace pour la représentation des données, l'espace d'arrivée étant plus propice à la détection des formes souhaitées. L'objectif est de créer un espace tel que les formes proches s'accumulent en un point de l'espace [Illingworth and Kittler, 1988]. En deux dimensions, d'autres primitives telles que les cercles et les ellipses ont ainsi pu être détectées [Tsuji and Matsumoto, 1978] de même que des coins [Davies, 1988; Shen and Wang, 2002]. Avec la transformée de Hough généralisée, [Ballard, 1981] étend la détection à des formes non paramétriques dans les images. La transformée de Hough a depuis aussi été utilisée pour la détection et la classification [Barinova et al., 2010, 2012; Gall and Lempitsky, 2009; Lowe, 2004; Okada, 2009].

La transformée de Hough a aussi été appliquée pour des dimensions supérieures à deux. Elle a été particulièrement utilisée en trois dimensions pour la segmentation et la reconnaissance de formes [Knopp et al., 2010; Pham et al., 2011]. L'algorithme original

est en pratique peu utilisable car très gourmand en ressources. Kiryati et al.[Kiryati et al., 1991] ont proposé une version probabiliste, ils n'observent qu'une partie des données pour réduire le temps de calcul. La transformée de Hough aléatoire a été définie par Xu et al. [Xu and Oja, 2009; Xu et al., 1990]. Dans la version originale, on tire un point puis on vote dans l'accumulateur pour l'ensemble des formes auxquelles il pourrait appartenir. Ici, on génère un modèle possible à partir des données (par exemple, on tire trois points 3D pour générer un plan) puis on vote dans la case correspondante de l'accumulateur.

Que ce soit pour la transformée de Hough ou sa version aléatoire, le critère pour arrêter de tirer des primitives est un paramètre fixe et arbitraire défini par l'utilisateur.

4.3 Principe de la transformée de Hough

Cette section présente les principes de la transformée de Hough et de sa variante aléatoire (Randomized Hough Transform, RHT). Les explications sont données par l'étude de l'exemple de l'extraction de lignes dans une image en 2D.

4.3.1 Transformée de Hough

Soit $P = p_1, p_2, \ldots, p_n$ un ensemble de n points en deux dimensions. Le but est de rechercher les droites représentant ces points. Une droite en 2D est donnée par une équation du type:

$$L(a,b,c) = \left\{(x,y) \in \mathbb{R}^2,\ ax + by + c = 0\right\} \tag{4.1}$$

avec $(a,b,c) \in \mathbb{R}^3$. L'idée principale de la transformée de Hough est de trouver un espace où les lignes vont être représentées par un point. Ainsi, une accumulation d'occurrences de la même ligne se traduira par un amas de points localisés dans la même zone de l'espace.

Choix de l'espace

L'utilisation du vecteur $(a,b,c) \in \mathbb{R}^3$ pour décrire la droite n'est pas une bonne solution car une droite n'est pas définie de manière unique. Elle est définie à multiplication des coordonnées par un réel près. Les trois coordonnées n'étant pas indépendantes, on souhaite représenter la droite par deux coordonnées. Si on impose $||(a,b)||_2 = 1$, (a,b) est sur le cercle unité, et peut être représenté par un angle $\theta \in [0,\pi]$, angle entre la normale orientée et l'axe des abscisses. Le deuxième paramètre est alors $\rho \in \mathbb{R}$, la distance signée de l'origine à la droite.

Une droite est représentée de façon unique par un couple $(\rho,\theta) \in [0,\pi] \times \mathbb{R}$ (figure 4.1).

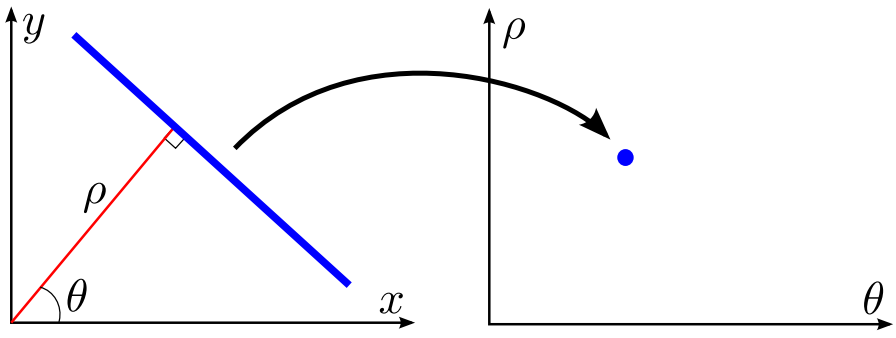


FIGURE 4.1 – Espace (ρ,θ)

Représentation des points

Soit $p = (x, y),\ (x, y) \in \mathbb{R}^2$ un point de l'espace usuel (espace de départ). L'ensemble des droites passant par ce point est :

$$\left\{(a, b, c) \in \mathbb{R}^3,\ ax + by + c = 0\right\} \tag{4.2}$$

ou encore en utilisant la description de la droite dans l'espace (ρ, θ) :

$$\{(\rho, \theta) \in [0, \pi] \times \mathbb{R},\ \cos(\theta)x + \sin(\theta)y - \rho = 0\} \tag{4.3}$$

Ainsi un point de $\mathbb{R}^2$ est représenté par une courbe dans l'espace (ρ, θ) (figure 4.2):

$$\rho = \cos(\theta)x + \sin(\theta)y \tag{4.4}$$

La courbe décrit l'ensemble des droites passant par (x, y).

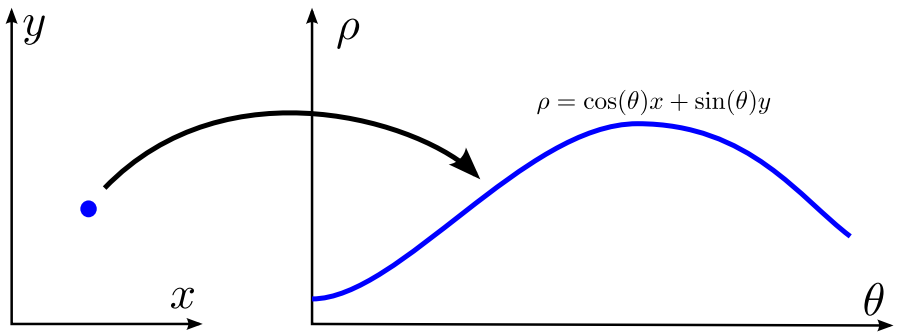


FIGURE 4.2 – Représentation des points

Algorithme

Pour chaque point de l'espace initial, il suffit de tracer la courbe correspondante dans l'espace (ρ, θ). Lorsque plusieurs points sont alignés les courbes se coupent au même endroit. Les points correspondant à l'intersection de nombreuses courbes sont alors les droites principales du nuages de points initial. En pratique, l'espace d'arrivée est discrétisé pour faciliter le décompte du nombre de courbes.

L'algorithme 1 résume la démarche pour l'extraction de droites dans une image en deux dimensions. En fin d'algorithme, la procédure "est un maximum local" pour le choix des lignes à retenir. Cette procédure varie suivant les algorithmes. Par exemple, il est possible de regarder directement le voisinage d'une case dans l'accumulateur, ou encore de faire une régression sur les cases voisines de la case considérée.

La figure 4.3 est un exemple d'extraction de lignes. Les points considérés sont les pixels noirs de l'image de départ. Pour remplir le tableau, tous les points ont été utilisés. L'image de droite donne la forme de l'accumulateur une fois rempli. Plus la couleur est foncée, plus la zone a reçu un nombre important de votes. On observe quatre zones de densité très élevée correspondant aux quatre segments de l'image.

4.3.2 Transformée de Hough Aléatoire

La transformée de Hough aléatoire (RHT) a été introduite par [Xu and Oja, 2009; Xu et al., 1990]. Appliqué à la détection de droites, le principe n'est plus de prendre un point et d'envisager toutes les droites passant par ce point, mais d'utiliser des droites plausibles. Pour ce faire, il suffit de tirer deux points et d'incrémenter la case de l'accumulateur

Algorithme 1 Transformée de Hough

Paramètre: n_ρ # Discrétisation en ρ
Paramètre: n_θ # Discrétisation en θ
Paramètre: n_{min} # Nombre minimal de pixels pour une droite

```
L ← ∅
Θ = {iπ/n_θ, i ∈ ⟦1, n_θ⟧}
T ← Tableau vide de taille n_ρ × n_θ
pour tout p = (x, y) ∈ P faire
    pour tout θ, i_θ ∈ Θ faire
        ρ = cos(θ)x + sin(θ)y
        i_ρ = indice de la cellule correspondant à ρ dans T
        T[i_ρ, i_θ] ← T[i_ρ, i_θ] + 1
    fin pour
fin pour
pour tout (i_ρ, i_θ) faire
    si T[i_ρ, i_θ] > n_min et T[i_ρ, i_θ] est un maximum local alors
        L ← L ∪ {(ρ, θ)}
    fin si
fin pour
```

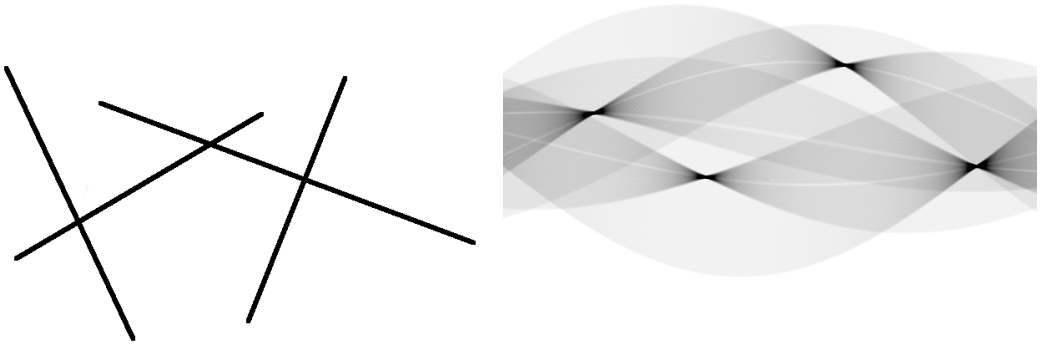


FIGURE 4.3 – Exemple avec quatre segments dans une image (à gauche). A droite, allure de l'accumulateur une fois rempli. Les points utilisés sont les pixels noirs de l'image de gauche.

correspondant à la droite passant par ces deux points. Le principal avantage est la réduction du temps de calcul, il n'y a plus l'ensemble de l'espace des droites passant par un point à balayer.

De plus, comme le montre la figure 4.4, les droites envisagées se limitent à des droites plausibles. Les maxima locaux sont mieux définis, l'accumulation est plus franche. Il est donc plus aisé de repérer les droites principales. La simulation a été effectuée avec 400000 tirages de couples de pixels noirs, nombre arbitrairement fixé.

4.4 Une borne robuste sur la taille des échantillons

Dans le cas d'une transformée de Hough aléatoire, le nombre de tirages possibles peut s'avérer très grand. Par exemple, lors d'une recherche de ligne dans une image, si N_p

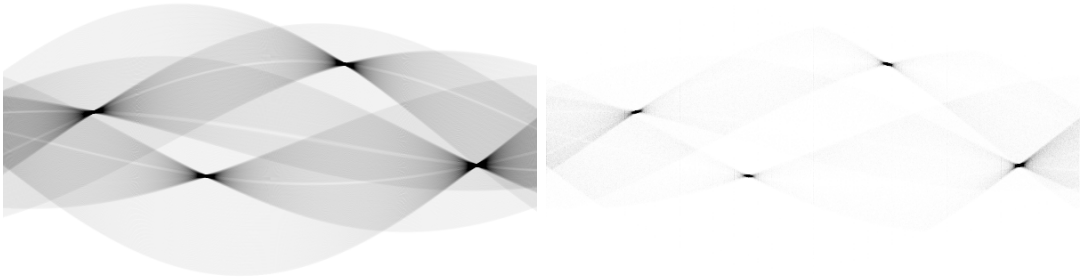


FIGURE 4.4 – Comparaison de la forme de l'accumulateur pour la transformée de Hough (à gauche) et la RHT (à droite).

est le nombre de pixels éligibles à l'appartenance à une ligne, le nombre de paires est $O(N^2)$; pour une recherche de plans définis par 3 points ce nombre devient $O(N^3)$. On conçoit donc aisément qu'envisager tous les tirages est une stratégie qui ne passe pas à l'échelle. Les méthodes présentées précédemment reposent sur le choix d'un nombre de tirages souvent renseigné par l'utilisateur. Le succès de la segmentation est dès lors lié à l'expérience de l'utilisateur sur le type de données qu'il tente de segmenter.

Réduire au maximum le nombre de tirages est nécessaire, mais une réduction trop drastique entraîne une mauvaise estimation des niveaux de votes dans chaque case de l'accumulateur. Il faut donc trouver un compromis entre le nombre de tirages et la précision de l'estimation. Nous voulons de plus que cette borne sur le nombre de tirages soit indépendante de la forme recherchée, pour rester le plus général possible.

4.4.1 Objectif

Plus formellement, soit M le nombre de cases de l'accumulateur. Chaque case de l'accumulateur suit une loi de Bernouilli de paramètre $p_m,\ p \in \{1, \dots, M\}$. Une bonne estimation de la distribution des tirages dans chaque cellule revient à approcher au mieux chacun des p_m, inconnus.

Soit T le nombre de tirages effectués, notons $\{X_{m,t},\ m \in \{1, \dots, M\},\ t \in \{1, \dots, T\}\}$ les variables aléatoires indépendantes et identiquement distribuées associées à chaque couple $(case, tirage)$. $X_{m,t} = 1$ si le tirage d'indice t vote dans la case m, 0 sinon.

Notons $\hat{p}_m$ la moyenne empirique obtenue pour chaque cellule :

$$\hat{p}_m = \sum_{t=1}^{T} X_{m,t} \tag{4.5}$$

ou encore :

$$\hat{p}_m = \frac{nombre\ de\ votes\ dans\ la\ case\ m}{nombre\ total\ de\ tirages} \tag{4.6}$$

L'objectif est d'estimer le nombre T^*, minimal, tel que $\hat{p}_m$ et p_m soient proches. Soient δ la distance maximale autorisée entre $\hat{p}_m$ et p_m, et $\alpha \in]0, 1[$ le niveau de confiance voulu sur l'estimation, alors l'objectif est de trouver T^* tel que :

$$\mathbb{P}(\max_{m \in \{1,\dots,M\}} |\hat{p}_m - p_m| < \delta) \geq \alpha \tag{4.7}$$

4.4.2 Borne inférieure pour le nombre de tirages

Théorème 1.

$$T^* \geq \frac{1}{2\delta^2} \log\left(\frac{2M}{1-\alpha}\right) \tag{4.8}$$

est une condition suffisante pour que l'équation 4.7 soit vraie.

Démonstration. Comme les $X_{m,t}$ sont iid, l'inégalité de [Hoeffding, 1963] s'applique :

$$\forall m \in \{1,\dots,M\}, \mathbb{P}(|\hat{p}_m - p_m| \geq \delta) \leq 2\exp\left(-\frac{2\delta^2 T^{*2}}{\sum_{t=1}^{T^*}(a_t - b_t)^2}\right)$$

où a_t et b_t sont les bornes d'un intervalle tel que $X_{m,t}$ soit à valeurs dans $[a_t, b_t]$. Ici on prend $a_t = 0$ et $b_t = 1$. La relation devient :

$$\forall m \in \{1,\dots,M\}, \mathbb{P}(|\hat{p}_m - p_m| \leq \delta) \leq 2\exp\left(-2\delta^2 T^*\right)$$

.

Cette relation est vraie quel que soit $m \in \{1,\dots,M\}$, ainsi :

$$\begin{aligned}
\mathbb{P}(\max_{m\in\{1,\dots,M\}} |\hat{p}_m - p_m| \geq \delta) &= \mathbb{P}(\exists m \in \{1,\dots,M\},\ |\hat{p}_m - p_m| \geq \delta) \\
&\leq \sum_{m=1}^{M} \mathbb{P}(|\hat{p}_m - p_m| \geq \delta) \\
&\leq 2M\exp\left(-2\delta^2 T^*\right)
\end{aligned}$$

il vient :

$$\begin{aligned}
\mathbb{P}(\max_{1,\dots,M} |\hat{p}_m - p_m| < \delta) &= 1 - \mathbb{P}(\max_{1,\dots,M} |\hat{p}_m - p_m| \geq \delta) \\
&\leq 1 - 2M\exp\left(-2\delta^2 T^*\right)
\end{aligned}$$

Considérant maintenant l'équation 4.7 :

$$\alpha \leq 1 - 2M\exp\left(-2\delta^2 T^*\right)$$

et en isolant T^* :

$$T^* \geq \frac{1}{2\delta^2} \log\left(\frac{2M}{1-\alpha}\right)$$

□

En pratique T^* est un entier, et il ne peut pas être plus grand que le nombre de tirages correspondant à l'estimation exacte de la distribution, c'est-à-dire, le nombre de tirages possibles, T_{max}. On choisira donc :

$$T^* = \min\left(\left\lceil \frac{1}{2\delta^2} \log\left(\frac{2M}{1-\alpha}\right)\right\rceil,\ T_{max}\right) \tag{4.9}$$

4.5 Recherche de la forme la plus probable

La section 4.4 donne une borne sur le nombre de tirages à effectuer pour que l'ensemble de la distribution sur l'accumulateur soit correctement estimée. Cependant, dans le cas où on ne cherche qu'une forme, la forme la plus probable, cette borne peut sembler être surestimée.

Considérons l'exemple suivant. Nous cherchons une ligne, dans un nuage de points 2D. Les points à notre disposition pour effectuer les tirages sont en fait alignés, hormis quelques points aberrants. Pour peu que nous ayons choisi un accumulateur fin (beaucoup de cases), un niveau de confiance fort et une distance faible, le nombre de paires à tirer peut être grand. Or, lorsque nous tirons les paires, nous votons presque tout le temps dans la même cellule. Il est légitime de vouloir arrêter les tirages lorsque nous "savons" quelle est la case la plus probable.

4.5.1 Objectif

Le but est de trouver, avec un niveau de confiance donné, la case de l'accumulateur correspondant au maximum de la distribution de probabilité :

$$m^* = \underset{m\in\{1,\dots,M\}}{\operatorname{argmax}}\ p_m \tag{4.10}$$

4.5.2 Un critère d'arrêt pour la recherche du maximum

Considérons les moyennes empiriques $\hat{p}_{m_i},\ i \in \{1,\dots,M\}$, classées par ordre décroissant. À chacune de ces moyennes est associé un intervalle de confiance. Plus le nombre de tirages effectués est grand, plus cet intervalle est petit. Pour déterminer le maximum de la distribution sur l'accumulateur, il suffit que les intervalles de confiance des deux cases ayant reçu le plus de votes ne s'intersectent pas.

Théorème 2. *Soit α un niveau de confiance. Notant m_1 et m_2 les indices des deux cases de l'accumulateur ayant reçu le plus de votes, T est le nombre de tirages effectués et $z(\alpha)$ est tel que l'intégrale de la densité gaussienne centrée réduite entre $-x$ et x est α :*

$$\hat{p}_{m_1} - \hat{p}_{m_2} > \frac{z(\alpha)}{\sqrt{T}} \tag{4.11}$$

est une condition suffisante, pour le niveau de confiance α, pour que les intervalles de confiance de m_1 et m_2 ne s'intersectent pas.

Démonstration. D'après le Théorème de la Limite Centrale :

$$\frac{\hat{p}_m - p_m}{\sqrt{p_m(1-p_m)/T}}$$

converge en distribution vers la loi normale centrée réduite $\mathcal{N}(0,1)$.
L'intervalle de confiance, au niveau α, associé à p_m est alors :

$$\hat{p}_m - z(\alpha)\sqrt{\frac{p_m(1-p_m)}{T}} \le p_m \le \hat{p}_m + z(\alpha)\sqrt{\frac{p_m(1-p_m)}{T}}$$

de plus comme $p_m \in [0,1]$, $p_m(1-p_m) \le \frac{1}{4}$, et :

$$\hat{p}_m - \frac{1}{2}\frac{z(\alpha)}{\sqrt{T}} \le p_m \le \hat{p}_m + \frac{1}{2}\frac{z(\alpha)}{\sqrt{T}}$$

Pour que les intervalles de confiance associés aux cases d'indice m_1 et m_2 ne s'intersectent pas, il faut que :

$$\hat{p}_{m_2} + \frac{1}{2}\frac{z(\alpha)}{\sqrt{T}} < \hat{p}_{m_1} - \frac{1}{2}\frac{z(\alpha)}{\sqrt{T}}$$

et donc :

$$\hat{p}_{m_1} - \hat{p}_{m_2} > \frac{z(\alpha)}{\sqrt{T}}$$

□

Pour un $\alpha = 95\%$, $z(\alpha) \simeq 2$. Ainsi, pour un niveau de confiance à 95%, l'équation (4.11) devient :

$$\hat{p}_{m_1} - \hat{p}_{m_2} > \frac{2}{\sqrt{T}} \tag{4.12}$$

La figure 4.5 illustre le critère d'arrêt pour la recherche de la case la plus probable de l'accumulateur, pour $\alpha = 95\%$. Dès que les intervalles de confiance à α des deux cases les plus probables ne s'intersectent plus, le maximum est déterminé avec une confiance α.

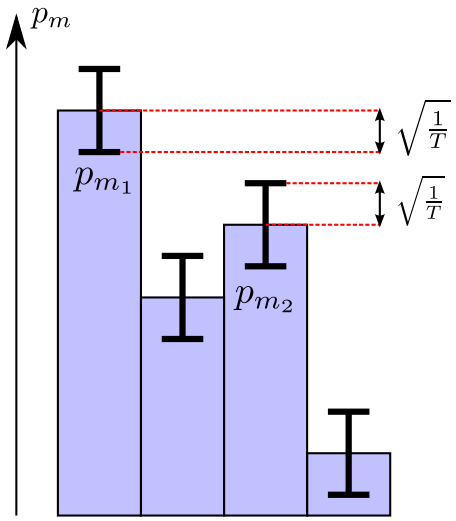


FIGURE 4.5 – Illustration des intervalles de confiance sur l'estimation de la probabilité pour chaque case de l'accumulateur.

4.5.3 Extension pour la recherche des k formes les plus probables

Le même raisonnement s'applique à la recherche des k formes les plus probables. Il suffit que les intervalles de confiances associés aux k cases de l'accumulateur avec les moyennes empiriques les plus élevées n'intersectent pas les intervalles des cases restantes.

En pratique, il suffit que :

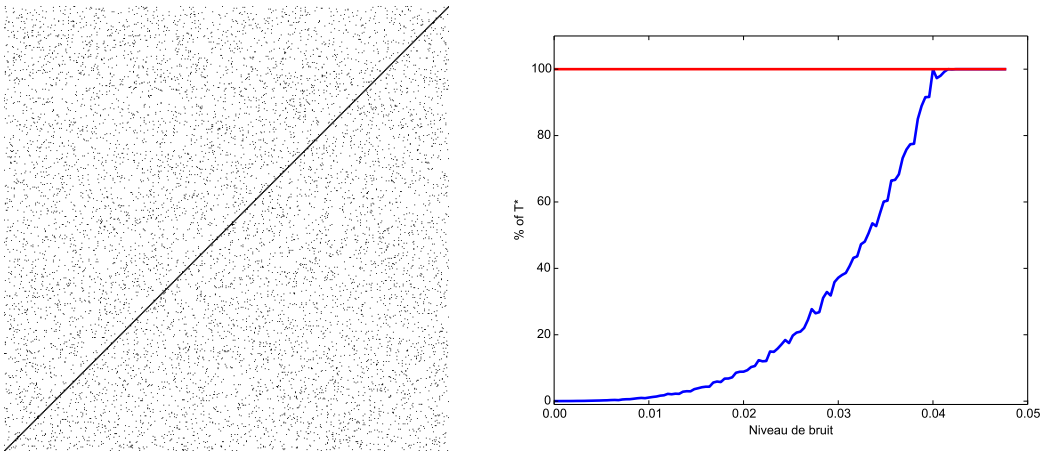
$$\hat{p}_{m_k} - \hat{p}_{m_{k+1}} > z(\alpha)\sqrt{\frac{1}{T}} \tag{4.13}$$

4.5.4 Illustration sur un exemple 2D

La figure 4.6 est une illustration de l'efficacité du critère pour la recherche du maximum.

La scène est une image blanche bruitée sur laquelle est dessinée une ligne dans la diagonale. La figure 4.6 à gauche représente une telle scène avec un niveau de bruit de 3.4% (c-à-d 3.4% des pixels sont colorés en noir).

Un tirage correspond au choix d'une paire de pixels noirs. La taille de l'accumulateur utilisé est de 2.5×10^5 cases. Pour $\alpha = 0.99$ et $\delta = 0.01$, l'équation 4.9 donne un nombre maximal de tirages de $T^* = 88638$. La courbe de droite retranscrit la proportion de T^* atteinte avant que m^* soit déterminé. Pour chaque valeur de bruit, 10 simulations sont effectuées.



À gauche, exemple synthétique, une image représentant une ligne à laquelle a été ajouté 3.4% de bruit. À droite, le pourcentage de T^* atteint avant que m^* soit déterminé.

FIGURE 4.6 – Exemple d'utilisation du critère d'arrêt pour la recherche du maximum.

On observe que pour les bruits faibles l'utilisation du critère réduit beaucoup le nombre de tirages à effectuer pour déterminer la case de l'accumulateur représentant le maximum de la distribution. Dès que le niveau de bruit est élevé, le critère ne permet pas de déterminer le maximum avant d'atteindre T^*.

Ce critère est donc particulièrement utile dans le cas des données peu bruitées, où il va permettre un gain de temps de calcul important tout en garantissant la précision du résultat.

Chapitre 5

Estimation robuste de normales

5.1 Introduction

Supposant que des points soient des mesures effectuées sur une surface, l'estimation de normales est l'association à chaque point d'un vecteur (la normale) représentant l'orientation de la surface sous-jacente en ce point. Ce problème s'avère difficile car très localisé : deux points très proches spatialement peuvent avoir des normales très différentes. De plus, les données sont très partielles : on ne connaît que les points. Il faut donc tenter d'interpoler et d'ajouter des a priori sur la surface pour estimer correctement les normales.

La figure 5.1 montre trois surfaces possibles pour un même ensemble de points. Pour pouvoir estimer une normale cohérente, il est nécessaire de faire des suppositions sur la forme de la surface. Par exemple, la surface peut être régulière (S_2) ou planaire par morceaux (S_1).

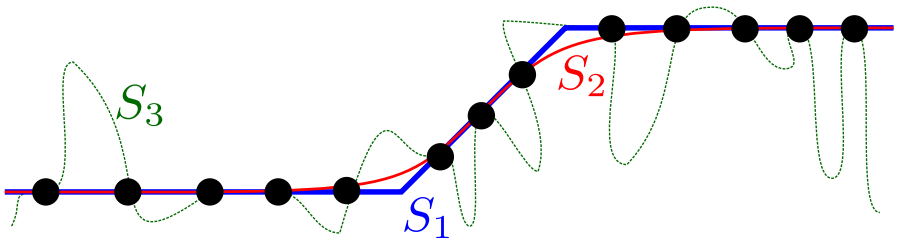
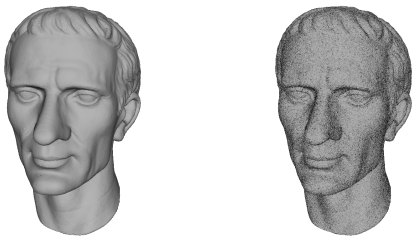


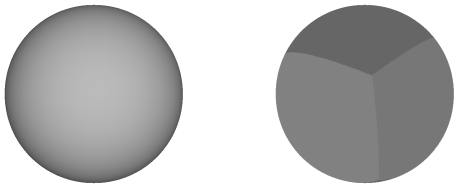
FIGURE 5.1 – Plusieurs surfaces possibles pour un même ensemble de points.

L'estimation de normales pour les nuages de points est un problème crucial en géométrie. Nombre d'algorithmes traitant de nuages de points prennent en entrée des couples points-normales. Les applications de ces algorithmes sont diverses, par exemple le rendu à partir de nuages de points [Rusinkiewicz and Levoy, 2000] et figure 5.2, la reconstruction de surfaces [Öztireli et al., 2009] ou de surfaces planaires par morceaux [Chauve et al., 2010] ou encore l'extraction de primitives géométriques dans les nuages de points [Schnabel et al., 2007]. La qualité des sorties de ces algorithmes dépend beaucoup de la qualité de ses entrées : points et normales. Il est donc important de fournir des normales bien estimées, ou du moins les mieux adaptées aux besoins de l'algorithme.

Toutes les méthodes d'estimation de normales se basent à un moment ou à un autre sur l'observation des points dans le voisinage de la zone où l'on souhaite estimer la normale. La compréhension de l'allure locale de la surface permet de calculer sa normale.



(a) Masque de Jules César [aim@shape, 2006], à gauche, rendu original, à droite, les normales bruitées donnent une impression de granularité.



(b) Nuage de points sur une sphère. À gauche, les normales sont correctes. À droite les normales sont celles d'un cube, donnant ainsi l'impression d'une zone anguleuse à la surface de la sphère.

FIGURE 5.2 – Exemple de rendu à partir de nuages de points, et influence de la normale sur la perception visuelle de la forme. La position des points est fixe, seule la normale varie.

Par ailleurs, un bon estimateur de normales doit remplir plusieurs critères (figure 5.3):

- être robuste au bruit (figure 5.3a),
- être insensible aux points aberrants (figure 5.3b),
- retrouver la normale près des angles sans lisser (figure 5.3c),
- s'accommoder de l'anisotropie locale, densité de points variable à l'échelle du voisinage d'un point (figure 5.3d),
- être rapide.

L'anisotropie est un problème réel, tant dans les nuages photogrammétriques que dans les nuages laser. La figure 5.4 illustre des cas courants d'anisotropie. Les détails mis en exergue illustrent les variations de densité au niveau des arêtes vives. La densité dépend de l'angle d'incidence du rayon laser. Les normales y ont été estimées avec l'algorithme développé dans ce chapitre.

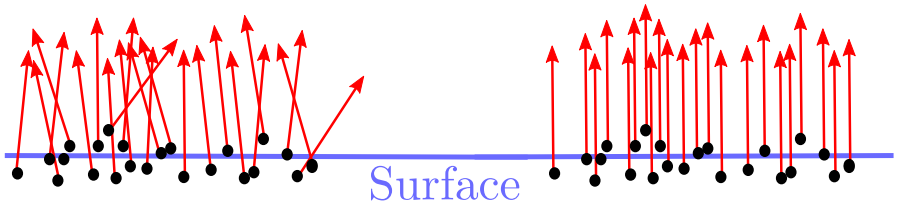
Ce chapitre présente un nouvel estimateur de normales pour les nuages de points non structurés. Après avoir donné la définition d'une normale (section 5.2) et présenté les travaux existants sur l'estimation de normales (section 5.3), nous proposons dans la section 5.4 un estimateur de normales basé sur la transformée de Hough robuste du chapitre 4. La section 5.5 est consacrée aux résultats. Les performances de notre méthode sont mesurées sur plusieurs expériences, qui montrent qu'elle est au moins aussi précise et robuste au bruit que l'état de l'art tout en étant plus rapide.

5.2 Notations et définitions

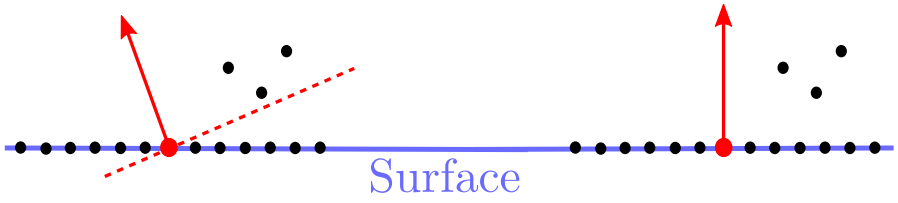
Définition 1. *Un point de la surface est dit régulier lorsque la surface admet un plan tangent.*

Pour une surface usuelle, tous les points sont réguliers hormis ceux qui se trouvent sur une arête ou un coin (Figure 5.5).

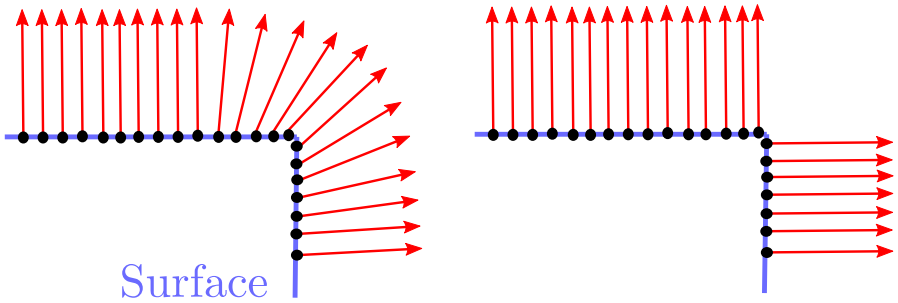
Définition 2. *Pour un point P régulier de $\mathcal{M}$, la normale en P est la direction orthogonale au plan tangent.*



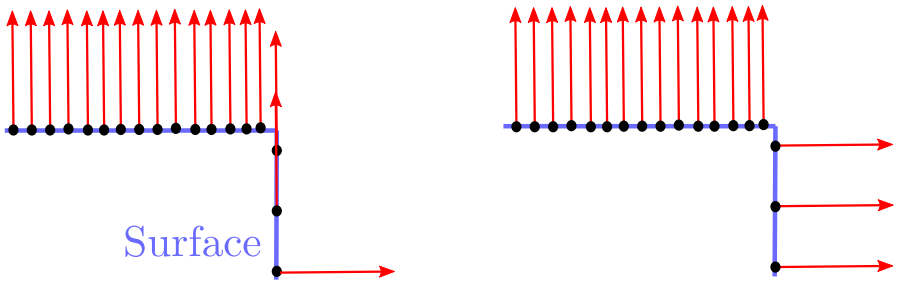
(a) Influence du bruit.



(b) Influence de points aberrants.



(c) Effet de lissage



(d) Influence de l'anisotropie.

Figures de gauche, estimations possibles et erronées. Figures de droite, estimations correctes recherchées.

FIGURE 5.3 – Exigences pour un bon estimateur de normales.

Dans le cas d'un paramétrage (u, v), $\mathcal{C}^1$:

$$\mathcal{M}(u, v) = (x(u, v), y(u, v), z(u, v))$$

C'est-à-dire où x,y et z sont $\mathcal{C}^1$, la normale est donnée par :

$$\mathbf{n}(u, v) = \frac{\partial \mathcal{M}}{\partial u} \wedge \frac{\partial \mathcal{M}}{\partial v}$$

Si la surface est définie par une équation cartésienne $f(x, y, z) = 0$, avec f $\mathcal{C}^1$. La normale est le gradient. Un point est régulier si le gradient est non nul.

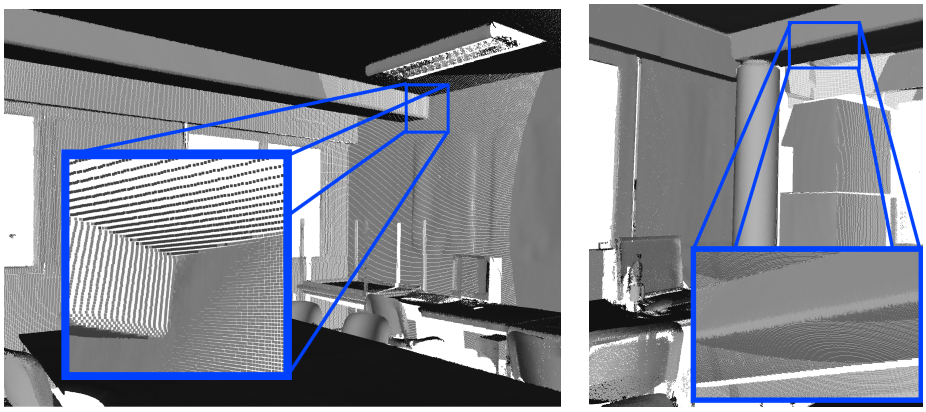


FIGURE 5.4 – Estimation de normales pour un nuage laser. Le nuage accuse une grande variété de densité, particulièrement au niveau des arêtes vives.

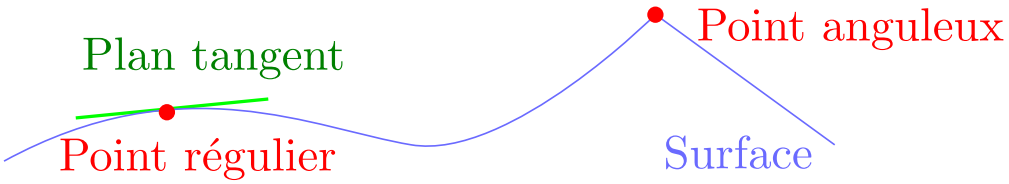


FIGURE 5.5 – À gauche, un point régulier et son plan tangent. À droite, un point anguleux, le plan tangent n'y est pas défini.

La normale est en pratique une direction, définie par deux angles (coordonnées sphériques). Lorsque la surface est orientable, c'est à dire qu'il est possible de discerner l'intérieur de l'extérieur, on peut ajouter à la direction un sens, la normale est alors *orientée*. La convention usuelle est de choisir la normale orientée de l'intérieur vers l'extérieur.

5.3 Travaux précédents

Régression. Les méthodes d'estimation de normales les plus utilisées reposent sur la régression. Hoppe et al. [1992] estiment les normales en considérant un point et ses voisins pour calculer un plan de régression aux moindres carrés, calculé par Analyse en Composantes Principales (ACP). La normale du point est alors la normale au plan. Pour améliorer la précision, il est possible d'utiliser d'autres types de surfaces comme des sphères [Guennebaud and Gross, 2007] ou des "jets" d'ordre k (les k premiers termes de la somme de Taylor de la surface au point considéré) [Cazals and Pouget, 2005]. Les méthodes de régression sont des méthodes robustes au bruit car elles lissent la surface en cherchant localement une surface moyenne. Mitra et al. [2004] ont proposé un algorithme qui utilise une taille de voisinage adaptative en vue d'améliorer la qualité de l'estimation. La robustesse au bruit et aux points aberrants a été accrue dans les travaux de [Huang et al., 2009; Yoon et al., 2007].

Un inconvénient inhérent à ces méthodes est qu'elles sous-entendent que les surfaces sont $\mathcal{C}_1$, c'est-à-dire que la normale existe en chaque point et que leurs variations sont continues. L'effet principal est donc un lissage aux points anguleux, plus ou moins prononcé en fonction de la taille du voisinage considéré.

Amélioration de normales. Une autre approche est de partir d'une première estimation des normales et de la raffiner. Certains algorithmes estiment implicitement la surface représentée par le nuage de points, parmi eux on compte les Moving Least Squares (MLS) de [Alexa et al., 2001] et une version adaptative [Pauly et al., 2003] ou encore le robust Local Kernel Regression [Öztireli et al., 2009]. Ici, comme dans la régression, tous les points sont considérés comme réguliers, la normale est alors la valeur du gradient en chaque point. Cependant du fait que la surface peut avoir des zones de forte courbure, ces méthodes trouvent des normales au voisinage des points anguleux. Le principal inconvénient de ces méthodes est de nécessiter une normale initiale assez fiable, et parfois orientée. De même, le filtre bilatéral pour les normales par Jones et al. [2004] permet de détecter les zones anguleuses tout en lissant les régions régulières. Cependant cette méthode peut être lente et dépend aussi grandement de la qualité de la première normale.

Diagramme de Voronoï. Une approche originale est celle de Dey and Goswami [2006] qui construit le diagramme de Voronoï et choisit une normale alignée avec la direction principale de la cellule. Cette construction permet de retrouver les points anguleux mais n'est pas robuste au bruit. Une variation dans la position des points peut grandement changer la forme de la cellule et perturber la direction principale. [Alliez et al., 2007] ont proposé une variante robuste au bruit, qui ne se base plus seulement sur la cellule de Voronoï du point considéré, mais aussi sur celles de ses voisins.

RANSAC. Pour traiter à la fois le bruit et les points anguleux, [Li et al., 2010] ont utilisé une méthode alliant une estimation robuste du bruit et la recherche de plans tangents basée sur un RANSAC. Ce RANSAC se base sur le niveau de bruit détecté, ce qui en fait une méthode adaptative et efficace. Cependant, cette méthode n'est pas très rapide, environ une demi-heure pour 1.5 million de points. De plus elle ne prend pas en compte l'anisotropie pouvant être présente dans les données.

Le tableau 5.1 résume les différentes propriétés de quelques algorithmes d'estimation de normales. Aucune des méthodes citées n'est en mesure de répondre à tous les critères mis en exergue.

Robustesse à ...	Bruit	Points aberrants	Points anguleux	Anisotropie	Rapidité
Régression planaires	✓				✓
Régression Jet	✓				✓
NormFet (Dey et al.)			✓	✓	✓
RANSAC (Li et al.)	✓	✓	✓		

TABLE 5.1 – Résumé des capacités de quelques algorithmes.

5.4 Méthode

L'algorithme présenté ici a pour but d'estimer une normale en un point, étant donnés les points de son voisinage. Basé sur la transformée de Hough aléatoire et robuste du

chapitre 4, il a pour objectif de trouver le plan tangent le plus probable.

Pour ce faire, des triplets de points sont tirés au hasard dans le voisinage du point considéré. Chaque triplet définit un plan dont l'orientation, repérée par deux angles, (figure 5.6), vote dans un accumulateur en deux dimensions.

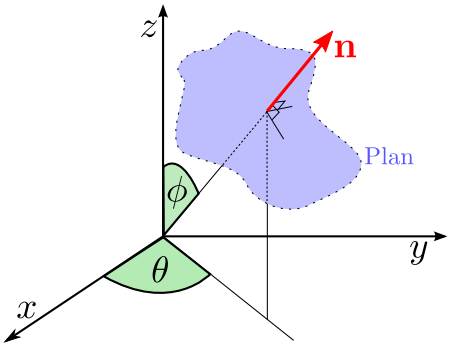


FIGURE 5.6 – La normale à un plan est représentée par deux angles, ϕ et θ.

La génération de ces triplets peut être effectuée de trois manières différentes, offrant trois compromis différents de robustesse à l'anisotropie et de vitesse. De plus, pour pallier les effets de seuil liés à l'utilisation d'un accumulateur discret, plusieurs explorations sont faites avec des accumulateurs ayant subi une rotation aléatoire. En utilisant les relations définies au chapitre 4, on extrait alors la normale comme la direction la plus probable.

5.4.1 Accumulateur

Le choix de la forme de l'accumulateur est crucial pour une transformée de Hough.

Dans le cas de la recherche de direction ou plus généralement de l'extraction de plans, l'accumulateur représente la sphère unité (deux angles). Il est important, pour ne pas introduire de distorsion dans la détection, que toutes les cellules représentent le même angle solide, ou tout du moins que les différences soient les plus minimes possibles.

L'accumulateur proposé par Duda and Hart [1972] est une simple discrétisation uniforme en ϕ et θ. Les coordonnées dans le tableau sont très simples à calculer mais on observe une grande distorsion des cases une fois le tableau dessiné sur la sphère (figure 5.7, à gauche). Les cases aux pôles sont plus petites qu'à l'équateur.

Censi and Carpin [2009] ont utilisé un accumulateur calqué sur un cube. La distorsion est plus faible que précédemment mais est toujours présente (figure 5.7, au milieu). Il est possible d'utiliser d'autres polyèdres, Zaharia and Prêteux [2002] ont utilisé un octaèdre. En effet plus le polyèdre est proche de la sphère, moins il y a de déformation des cases lors de la projection sur la sphère.

Pour réduire cette irrégularité dans la forme des cellules, il faut utiliser un accumulateur directement dessiné sur la sphère. L'utilisation d'une sphère géodésique semble une bonne solution, cependant le calcul des coordonnées de la case à incrémenter, étant donné un ou deux angles est plus fastidieux que précédemment. Borrmann et al. [2011] ont proposé un accumulateur directement dessiné sur la sphère, dont les coordonnées sont faciles à calculer et avec des cellules d'aires similaires (figure 5.7, à droite). Par la suite, c'est cet accumulateur qui sera utilisé.

Calcul des coordonnées

L'accumulateur de Borrmann et al. [2011] est paramétré par un unique paramètre $k \in \mathbb{N}$.

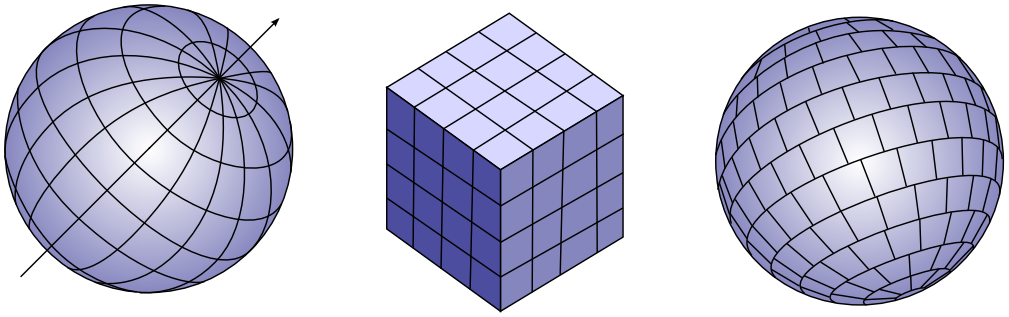


FIGURE 5.7 – Trois accumulateurs différents.

Le nombre de couronnes de l'accumulateur est $2k+1$. L'ouverture angulaire d'une couronne est ϕ', défini par :

$$\phi' = \frac{\pi}{2k},\ k \in \mathbb{N}$$

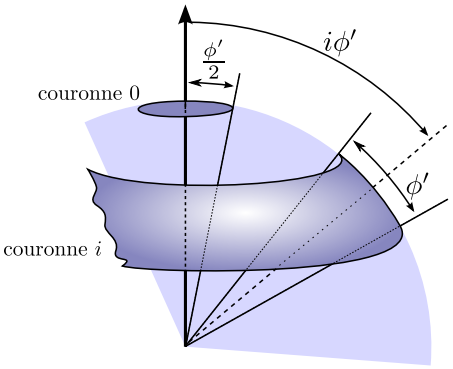


FIGURE 5.8 – Angles pour l'accumulateur de Borrmann et al. [2011].

Les couronnes aux pôles ont une ouverture de $\frac{\phi'}{2}$ (figure 5.8). Les couronnes sont repérées par l'indice $i \in \{0, \ldots, 2k\}$, les couronnes 0 et $2k$ correspondant aux pôles.

Soit $l_i,\ i \in \{1, \ldots, 2k-1\}$, le nombre de cases de l'accumulateur pour la couronne d'indice i.

Pour tout $i \in \{1, \ldots, 2k-1\}$, l'aire de la couronne est donnée par :

$$\mathcal{A}_{couronne_i} = 4\pi\ \sin(i\phi')\ \sin\frac{\phi'}{2} \tag{5.1}$$

Où $i\phi'$ est l'angle ϕ médian pour la couronne (figure 5.8).

On souhaite diviser chaque couronne en cases d'aires identiques. De plus, l'échelle de l'accumulateur, l'objectif est d'avoir des cases d'aire similaire.

A l'équateur, pour avoir des cases les plus isotropes possibles, on veut la même ouverture angulaire en ϕ et en θ :

$$l_k = \frac{2\pi}{\phi'} = 4k \tag{5.2}$$

Et l'aire d'une case à l'équateur :

$$a_k = \frac{\pi\ \sin\frac{\phi'}{2}}{k} \tag{5.3}$$

Comme l'objectif est d'avoir des cases les plus similaires possibles, on prend :

$$l_i = \begin{cases} 1 & \text{si } i \in \{0, 2k\} \\ \lfloor 4k \ \sin\left(\frac{i\pi}{2k}\right) \rceil & \text{sinon } ^1 \end{cases} \tag{5.4}$$

Pour une direction (ϕ, θ) et pour k données, les coordonnées de la case dans le tableau sont :

$$\begin{cases} (0,0) & \text{si } \phi < \frac{\pi}{4k} \\ (2k,0) & \text{si } \phi > \pi - \frac{\pi}{4k} \\ \left(i = \left\lfloor \frac{2k}{\pi} - \frac{1}{2} \right\rfloor, \ \left\lfloor \frac{\theta \, l_i}{2\pi} \right\rfloor\right) & \text{sinon} \end{cases} \tag{5.5}$$

Lors du calcul des normales, celles-ci ne sont pas orientées. Seule la moitié du tableau est alors utilisée.

5.4.2 Algorithme

Soit p un point du nuage de point P, pour générer des hypothèses de normales, nous générons aléatoirement des triplets dans le voisinage de p. Le voisinage N_p de p est soit l'ensemble des points dans une boule de rayon R centrée en p, soit les K plus proches voisins de p. Pour cet ensemble, nous utilisons un kd-tree.

L'utilisation de la transformée de Hough aléatoire et robuste du chapitre 4 donne le nombre de tirages en fonction du nombre de cases de l'accumulateur (équation 4.9):

$$T^* = \min\left(\left\lceil \frac{1}{2\delta^2} \log\left(\frac{2M}{1-\alpha}\right)\right\rceil, \ T_{max}\right) \tag{5.6}$$

Le nombre de cases de l'accumulateur est donné, en fonction de k dans le tableau 5.9.

k	4	5	7	10	12
M	23	82	171	290	441

FIGURE 5.9 – Nombre de cases de l'accumulateur en fonction de k.

Comme l'objectif est de trouver la normale la plus probable, la relation 4.11 est appliquée pour stopper les tirages lorsque le maximum est identifié avec une forte probabilité, c'est-à-dire dès que :

$$\hat{p}_{m_1} - \hat{p}_{m_2} > \frac{z(\alpha)}{\sqrt{T}} \tag{5.7}$$

où m_1 et m_2 sont les nombres de votes des deux cases ayant reçu le plus grand nombre de votes, et T le nombre de tirages effectués.

L'algorithme 2 présente le procédé général d'estimation de normales. Les deux procédures **triplet**(.) et **normale**(.), produisent respectivement un triplet de points étant donné un ensemble de points et une normale étant donné un ensemble de vecteurs. Ces procédures sont décrites en détail dans les sections 5.4.3 et 5.4.4. L'utilisation de plusieurs rotations, menant à la création de plusieurs normales sera elle aussi discutée dans la section 5.4.4.

5.4.3 Tirage des triplets de points

Soit p un point et N_p l'ensemble des points du voisinage de p. Un plan est généré par le tirage aléatoire d'un triplet de points dans N_p (procédure **triplet**(.) de l'algorithme 2). Nous ne nous limitons pas à la création d'un doublet auquel viendrait s'ajouter le point p. En effet, si la position de p est bruitée, utiliser p dans tous les cas créerait une normale erronée.

Trois manières de sélectionner ces triplets sont proposées.

Algorithme 2 Estimation de normales par transformée de Hough randomisée.

Paramètre: P # Nuage de points
Paramètre: k_ϕ # Discrétisation en ϕ
Paramètre: n_{rot} # Nombre de rotations de l'accumulateur
Paramètre: R ou K # Taille du voisinage

$T^* \leftarrow$ Résultat équation 5.6

Génération des rotations
$R_{init} \leftarrow n_{rot}$ rotations générées aléatoirement

Boucle principale
pour tout $p \in P$ **faire**
  $R \leftarrow R_{init}$
  $N_p \leftarrow$ voisinage de p
  $T \leftarrow 0$
  $Acc \leftarrow n_{rot}$ tableaux vides

  # initialisation des mémoires pour les cases ayant reçu le plus de votes.
  pour tout $r \in R$ **faire**
    $m_1[r] \leftarrow 0$, $m_2[r] \leftarrow 0$
  fin pour

  # Remplissage des accumulateurs
  tant que $T < T^*$ **faire**
    $pl = (p_1, p_2, p_3) \leftarrow$ **triplet**(N_p) # Génération de triplets de points (section 5.4.3)
    $\mathbf{n}_{pl} \leftarrow$ normale au plan support de pl
    pour tout $r \in R$ **faire**
      $\mathbf{n}_r \leftarrow r\ \mathbf{n}_{pl}$
      $Acc[r](\mathbf{n}_r) \leftarrow Acc[r](\mathbf{n}_r) + 1$ # Incrémentation dans l'accumulateur
      Mise à jour de $m_1[r]$ et $m_2[r]$
      si $\hat{p}_{m_1} - \hat{p}_{m_2} > z(\alpha)\sqrt{\frac{1}{T}}$ **alors**
        $R \leftarrow R \setminus \{r\}$
      fin si
    fin pour
    $T \leftarrow T + 1$
  fin tant que

  # Sélection de la normale
  $\mathcal{N} \leftarrow \emptyset$
  pour tout $r \in R_{init}$ **faire**
    $\mathcal{N} \leftarrow \mathcal{N} \cup \{\mathbf{n}(m_1[r])\}$
  fin pour
  $\mathbf{n}(p) \leftarrow$ **normale**$(\mathcal{N})$ # Génération de la normale finale à partir des hypothèses de normales (section 5.4.4)
fin pour

Tirage sur les points

Le plus immédiat pour sélectionner des points aléatoirement dans N_p est un tirage uniforme sur les points. Cette approche est très rapide, cependant les zones plus denses

sont plus fréquemment tirées. De ce fait, elles ont une influence plus importante sur le résultat que celles qui regroupent moins de points. Cette méthode de tirage n'est pas robuste à l'anisotropie.

Tirage quasi uniforme dans la boule

Pour être robuste à l'anisotropie, il faut être capable de tirer les points des zones moins denses avec une probabilité non biaisée par le nombre de points. Pour ce faire, on souhaite tirer un point q uniformément dans la boule du voisinage de p. Nous allons premièrement tirer un point uniformément sur la sphère unité avant d'obtenir q en tirant un rayon. Comme expliqué par [Weisstein, site], il existe plusieurs manières de tirer des points aléatoirement sur la sphère. Parmi elles, celle de [Cook, 1957] a l'avantage de ne pas nécessiter le calcul d'une racine carrée.

Soient U_0, U_1, U_2, U_3 trois variables aléatoires indépendantes de loi uniforme sur $[-1, 1]$. Notant u_0, u_1, u_2, u_3 les valeurs prises lors d'un tirage.

Pour tout quadruplet tel que :

$$u_0^2 + u_1^2 + u_2^2 + u_3^2 < 1$$

La variable $X_S = (X, Y, Z)$, prenant les valeurs (x, y, z) où :

$$x = \frac{2(x_1x_3 + x_0x_2)}{u_0^2 + u_1^2 + u_2^2 + u_3^2} \tag{5.8}$$

$$y = \frac{2(x_2x_3 - x_0x_1)}{u_0^2 + u_1^2 + u_2^2 + u_3^2} \tag{5.9}$$

$$z = \frac{u_0^2 + u_3^2 - u_1^2 + u_2^2}{u_0^2 + u_1^2 + u_2^2 + u_3^2} \tag{5.10}$$

a une distribution uniforme sur la sphère.

Soit R_{N_p} le rayon de la boule centrée en p correspondant au voisinage N_p. R_{N_p} est soit R si N_p est défini par un rayon de recherche, soit la distance entre p et le point de N_p le plus éloigné de p si N_p comme les K plus proches voisins de p.

Soit d la distance au centre tirée aléatoirement.

$$\mathbb{P}(d < R_{N_p}) = 1$$

On souhaite une répartition uniforme des points dans la boule. Pour $r \in [0, R_{N_p}]$, et notant $V(r)$ le volume de la boule de rayon r :

$$\mathcal{P}(d < r) = \frac{V(r)}{V(R_{N_p})} = \left(\frac{r}{R_{N_p}}\right)^3$$

Soit U une variable aléatoire de loi uniforme sur $[0, R_{N_p}]$, et soit R tel que :

$$R = R_{N_p}\, U^{\frac{1}{3}} \tag{5.11}$$

Alors R a la loi souhaitée pour le tirage du rayon.

Ainsi X_B, variable aléatoire ayant pour loi la loi uniforme dans une boule de rayon R_{N_p}, est décrite par :

$$X_B = R_{N_p}\, U^{\frac{1}{3}}\, X_S \tag{5.12}$$

Les points dans la boule sont tirés avec ce procédé.

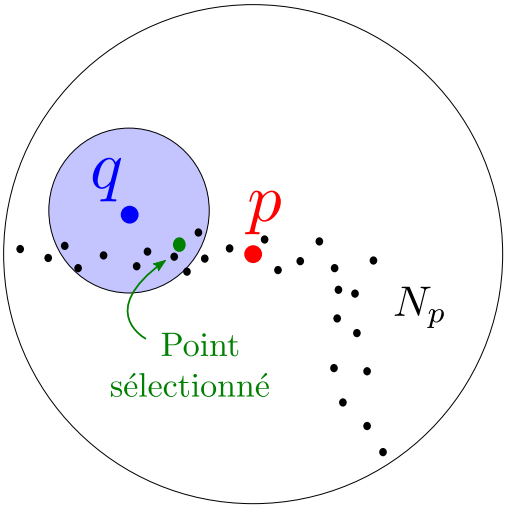


FIGURE 5.10 – Tirage quasi uniforme dans la boule du voisinage de p.

Une fois le point q tiré avec ce procédé, on prend un point de N_p au hasard dans la sphère de rayon R_{unif} centrée en q. R_{unif} est un paramètre de l'algorithme, c'est une fraction de R_{N_p}. Si la sphère ne contient aucun point de N_p, on recommence le procédé. Le tirage de points est illustré sur la figure 5.10.

La figure 5.11 montre le résultat de l'estimation de normales au niveau d'un coin avec une densité différente de chaque côté de celui-ci. À gauche, l'algorithme de Li et al. [2010], ne gère pas l'anisotropie, et à droite, le tirage quasi uniforme s'accommode de la variation de densité de points.

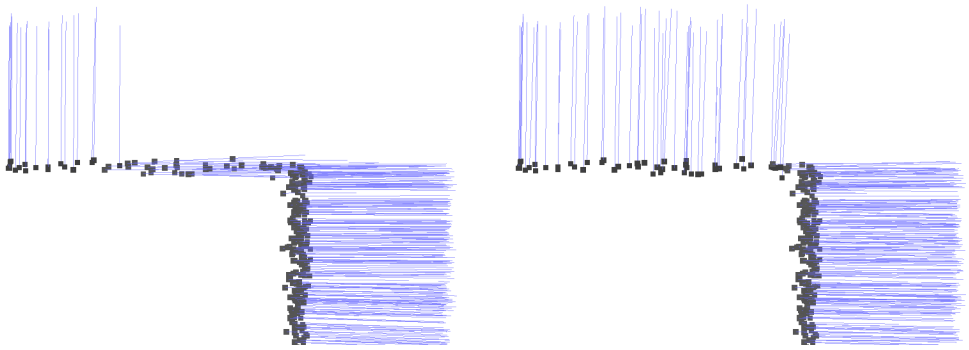
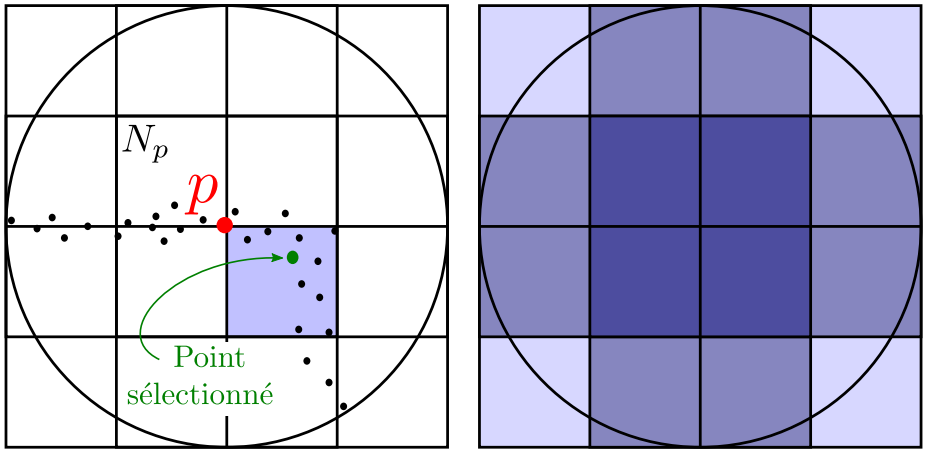


FIGURE 5.11 – Effet de l'anisotropie sur l'estimation de normales. À gauche, Li et al. [2010] et à droite, notre méthode avec triage quasi uniforme.

Tirage quasi uniforme discrétisé

Le triage quasi uniforme dans la boule donne de bons résultats mais est cependant plus lent que le tirage uniforme sur les points du voisinage.

Un compromis est de discrétiser la boule de voisinage en voxels. Dans ce cas, un cube est tiré uniformément parmi l'ensemble des voxels. S'il contient des points, on y tire un point avec une probabilité fonction du volume du voxel intersectant la boule. La figure 5.12a illustre ce tirage et la figure 5.12b donne une idée visuelle de la probabilité associée à chaque voxel.



(a) Tirage avec voxélisation de la boule du voisinage de p.

(b) Probabilité de tirage des cubes en fonction du volume de voxels intersecté par la boule.

FIGURE 5.12 – Tirage avec voxélisation et probabilités associées.

5.4.4 Effets de discrétisation

La normale est choisie dans la case de l'accumulateur qui a reçu le plus de votes. Pour éviter une normale quantifiée (une normale prédéfinie par case), la normale retenue est la moyenne des directions qui ont voté dans la case.

En pratique l'implémentation ne mémorise pas chaque normale, les vecteurs sont incrémentalement additionnés puis le vecteur résultant est normalisé pour produire la normale finale.

Cet effet de quantification n'est pas le seul problème lié à l'usage d'un accumulateur discret.

Premièrement, la disposition des cases, fixée a priori, peut être source d'erreur. Si le pic principal de la distribution à estimer se situe près d'une frontière entre deux cases, les votes correspondant à ce pic se répartissent dans les cases adjacentes. Le nombre de votes reçu par chaque case en est réduit d'autant, ce qui peut mener à une ambiguïté de choix entre ces cases. Un cas plus embêtant est celui de la présence de deux pics dans la distribution. Si le pic maximal est réparti entre deux cases, il est possible que la normale sélectionnée corresponde au second pic. Cet effet est d'autant plus présent que les données sont bruitées. Les pics sont alors aplatis par le bruit et l'identification du pic maximum est plus difficile.

Un deuxième effet, lié au choix de l'accumulateur de Borrmann et al. [2011] est la non isotropie des cases. Cet accumulateur garantit en effet que les cases ont toutes des aires similaires, mais elles ont des formes différentes. A l'équateur, ce sont des carrés tandis qu'aux pôles elles sont circulaires.

Pour pallier ces effets de discrétisation, il est possible de répartir les normales dans plusieurs cases, une normale votant dans les cases adjacentes avec des poids différents. Cependant, les adjacences entre cases ne sont pas bien définies. La solution retenue est de lancer l'algorithme plusieurs fois avec des accumulateurs ayant subi une rotation aléatoire. Comme le montre les expériences de la section 5.5, cela permet d'atténuer notablement les effets de discrétisation. Du point de vue de l'implémentation, il est plus facile d'effectuer une rotation des directions que de l'accumulateur.

Le recours à plusieurs accumulateurs fournit plusieurs normales. Il faut maintenant sélectionner la normale finale. Trois alternatives sont proposées.

- La normale retenue peut être une moyenne des normales fournies par tous les accumulateurs. Ceci à l'avantage d'être simple à calculer. Cependant au niveau des arêtes, si les accumulateurs ont fait des choix très différents, la normale est lissée.
- Ou bien la normale ayant reçu le plus de votes est retenue. Les normales près des arêtes et des angles sont alors bien estimées mais les surfaces planes peuvent apparaître granuleuses lorsque du bruit est présent.
- Alternativement, les normales proches peuvent être regroupées et moyennées pour fournir la normale finale. Cette fusion des deux points précédents est un bon compromis entre régularisation dans les parties planes et détection des angles.

5.5 Résultats

Cette partie, consacrée aux résultats, résume dans un premier temps les différents paramètres de l'algorithme. Ensuite la méthode est confrontée à l'état de l'art à l'aide de quatre méthodes représentatives :

- [Hoppe et al., 1992], pour la regression, qui est aussi la méthode la plus couramment utilisée.
- [Cazals and Pouget, 2005], une autre méthode de régression plus précise que la précédente.
- [Dey and Goswami, 2006], l'estimation de normales à l'aide d'un diagramme de Voronoï (NormFet).
- [Li et al., 2010], une méthode basée sur RANSAC donnant de bons résultats sur les points anguleux.

Ces comparaisons sont effectuées sur des données synthétiques pour avoir accès à la surface soujacente. Enfin, des visuels sur des scènes réelles, créés à l'aide d'un laser ou par photogrammétrie sont présentés.

Pour les données synthétiques, le bruit ajouté est un bruit gaussien isotrope avec un écart type exprimé en pourcentage de la boîte englobante du modèle avant bruitage.

5.5.1 Les paramètres

L'estimation de normale présentée ici se base sur plusieurs paramètres :

- K ou R, respectivement le nombre de voisins ou le rayon de la boule à utiliser pour calculer le voisinage du point considéré.
- T_R ou (α, δ), respectivement le nombre de triplets à tirer ou la paire (niveau de confiance, distance) permettant de calculer le nombre de triplets.
- k la discrétisation selon la colatitude ϕ, c'est le paramètre régissant la taille de l'accumulateur.
- n_{rot}, le nombre de rotations de l'accumulateur.

- c ou R_{unif}, respectivement la taille de la grille de voxels pour l'utilisation des cubes, ou le rayon de recherche après tirage dans la boule dans le cas du tirage quasi uniforme.

- a, la tolérance angulaire utilisée pour le regroupement des directions données par les multiples accumulateurs.

Bien que nombreux, ces paramètres sont faciles à estimer et peuvent être réutilisés. Dans toutes nos expériences, sauf mention contraire, les paramètres utilisés sont $T_R = 700$ (ce qui correspond pour $\alpha = 0.95$ à $\delta = 0.08$), $k = 7$, $n_{rot} = 5$, $c = 4$ ou $R_{unif} = 1/4$ et $a = \frac{\pi}{4}$. Il reste la taille du voisinage qui est le paramètre classique des méthodes d'estimation de normales autres que celles reposant sur un diagramme de Voronoï. Dans le cas des données synthétiques, nous utilisons $K = 500$, et pour les données réelles, nous fixons soit $K = 500$, soit le rayon de recherche est précisé.

Les tailles de voisinage utilisées pour les comparaisons varient d'un algorithme à l'autre. Pour les méthodes de regression ([Cazals and Pouget, 2005; Hoppe et al., 1992]), $K = 80$ est un bon compromis entre effet de lissage et sensibilité au bruit. Pour [Li et al., 2010], qui est la méthode se rapprochant le plus de celle présentée ici, le voisinage est de $K = 500$.

Les trois variantes présentées à la section 5.4.4 sur les problèmes liés à la discrétisation sont présentées séparément :

- **RRHT_Points** est l'échantillonage direct sur les points,

- **RRHT_Unif** est l'échantillonage quasi uniforme par tirage dans la boule de voisinage,

- **RRHT_Cubes** correspond à l'utilisation de la voxélisation de la boule.

5.5.2 Temps de calcul

La figure 5.13 présente l'évolution du temps de calcul en fonction du nombre de points (calculs effectués avec 2 processeurs Intel(R) Xeon(R) X5472 3.00GHz, 4 threads chacun). Les points, échantillonnés uniformément sur une sphère, ont été bruités à 0.2%. Ces résultats ont été obtenus en désactivant le critère d'arrêt basé sur les intervalles de confiance. Les courbes **RRHT** sont donc ici des bornes supérieures du temps de calcul nécessaire à l'estimation des normales.

Les deux versions les plus rapides de notre algorithme RRHT_points et RRHT_cubes ont un temps de calcul comparable avec la méthode de jet fitting et celle de Dey, basée sur un diagramme de Voronoï (Normfet). Celles-ci sont plus rapides que la méthode par RANSAC (Li & al) d'un ordre de grandeur, avec la même taille de voisinage. RRHT_Unif est quant à elle beaucoup plus lente, mais elle a l'avantage supplémentaire d'être robuste à l'anisotropie (voir la partie 5.5.6 sur l'anisotropie).

Tous les algorithmes ont une complexité proche de $O(n \log n)$ où n est la taille du nuage d'entrée. Le facteur en $\log n$ est dû aux recherches dans un Kd-tree, cette opération étant répétée pour chacun des points du nuage. Les constantes multiplicatives par contre varient grandement. Il faut cependant garder à l'esprit que le cas présenté figure 5.13 est en quelque sorte le pire des cas, le temps de calcul se réduisant avec l'utilisation des intervalles de confiance.

La figure 5.14 montre le temps que l'on peut espérer gagner en utilisant la relation 5.7. Pour plusieurs modèles géométriques et le modèle dragon [S3DR, 2005], sont présentés les rapports du temps de calcul avec le critère sur les intervalles de confiance et sans les intervalles de confiance. On observe que l'utilisation du critère réduit grandement

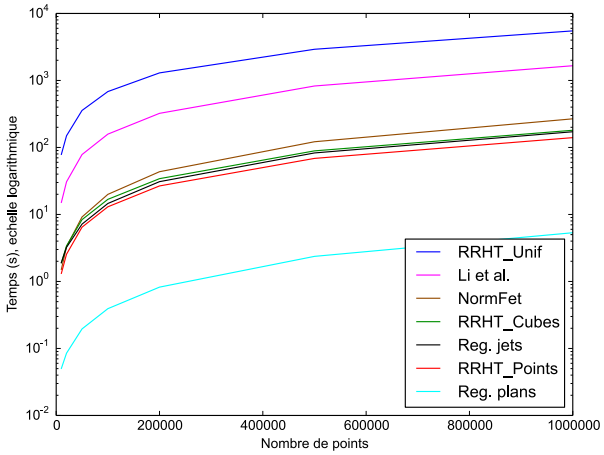


FIGURE 5.13 – Temps de calcul en fonction du nombre de points, pour différentes méthodes, sur une sphère avec 0.2% de bruit.

le temps de calcul. Dans la mesure où le bruit a pour effet d'aplatir les distributions, on observe effectivement que le bénéfice du critère diminue avec un bruit croissant. Le modèle utilisé a aussi une grande importance. Lorsque le modèle a peu de zones planes ou régulières mais beaucoup de points anguleux ou de parties à forte courbure, le temps nécessaire pour identifier le pic principal de la distribution est plus long.

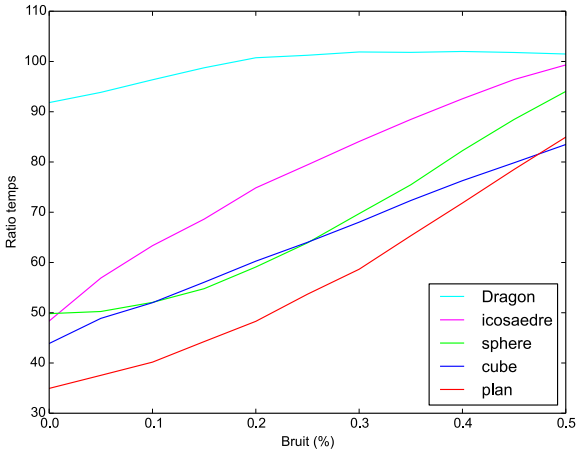


FIGURE 5.14 – Ratio du temps de calcul avec et sans utilisation des intervalles de confiance pour estimer la normale, en fonction du bruit et pour différentes formes géométriques et le modèle dragon [S3DR, 2005].

Le tableau 5.2 donne les temps de calculs pour des modèles réels. Comme il n'y a pas de bruit dans les données, la taille de voisinage est réduite à $K = 100$. Ici encore, l'utilisation du critère sur les intervalles de confiance est particulièrement utile pour les modèles avec de grandes zones planes comme la salle de réunion scannée avec un laser.

Modèle (# points)	T_R=700 $n_{rot}=5$ sans interv.	T_R=700 $n_{rot}=5$ avec interv.	T_R=300 $n_{rot}=2$ sans interv.	T_R=300 $n_{rot}=2$ avec interv.
Armadillo (173k)	*21 s*	*20 s*	*3 s*	*3 s*
Dragon (438k)	*55 s*	*51 s*	*8 s*	*7 s*
Buddha (543k)	1.1	1	*10 s*	*10 s*
Circular Box (701k)	1.5	1.3	*13 s*	*12 s*
Omotondo (998k)	2	1.2	*18 s*	*10 s*
Statuette (5M)	11	10	1.5	1.4
Salle de réunion scannée (6.6M)	14	8	2.3	1.6
Lucy (14M)	28	17	4	2.5

RRHT_points avec une taille de voisinage $K = 100$ points, avec et sans intervalles de confiance, pour deux jeux de paramètres (T_R, n_{rot}). Sauf mentionné explicitement, les temps sont en minutes.

TABLE 5.2 – Temps de calcul sur des données réelles.

5.5.3 Mesures d'erreur

Une mesure commune de l'erreur est la valeur efficace. Soit P un nuage de points, $\mathbf{n}_{p,ref}$ la normale de référence au point p (la normale théorique) et $\mathbf{n}_{p,est}$ la normale estimée. La valeur efficace (Root Mean Square en anglais) est définie par :

$$RMS = \sqrt{\frac{1}{|P|}\sum_{p\in P}\widehat{\mathbf{n}_{p,ref}\mathbf{n}_{p,est}}^2} \tag{5.13}$$

Cette mesure tend à favoriser les algorithmes minimisant le carré de l'erreur angulaire entre les normales. Cependant cette erreur ne reflète pas forcément une erreur « visuelle ». Dans le cas d'une arête vive, l'œil humain va être beaucoup plus sensible à un effet de lissage qu'à quelques points de part et d'autre de l'arête ayant reçu la normale de la face opposée. Dans cette optique, la valeur efficace seuillée, RMS_τ est définie par :

$$RMS\_\tau = \sqrt{\frac{1}{|P|}\sum_{p\in P} v_p^2} \tag{5.14}$$

où :

$$v_p = \begin{cases} \widehat{\mathbf{n}_{p,ref}\mathbf{n}_{p,est}} & \text{si } \widehat{\mathbf{n}_{p,ref}\mathbf{n}_{p,est}} < \tau \\ \frac{\pi}{2} & \text{sinon} \end{cases}$$

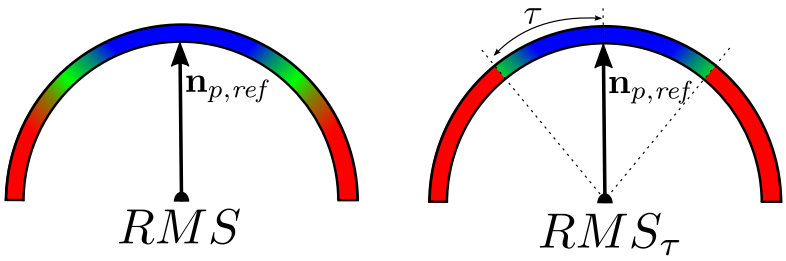
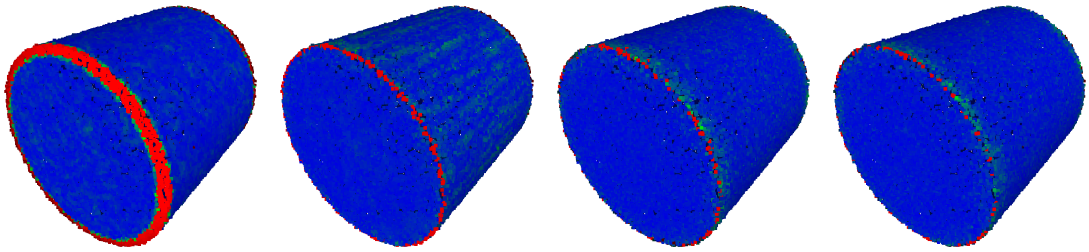


FIGURE 5.15 – Illustration des coûts de la RMS et de la RMS_τ. Pour la RMS_τ, dès que l'angle est supérieur à τ, la pénalité est maximale.

Cette mesure d'erreur donne une pénalité constante au-delà d'un seuil (figure 5.15). Une erreur angulaire supérieure à τ est aussi mauvaise qu'une erreur de $\frac{\pi}{2}$. Elle a pour effet de ne pas favoriser un effet de lissage par rapport à la valeur efficace.



Le modèle est un cylindre échantillonné avec 50000 points et un niveau de bruit à 0.2%. *Code couleur correspondant à* RMS_τ *avec* $\tau = 10°$ *(Figure 5.15).*

FIGURE 5.16 – Rendu de précision pour quatre algorithmes. De gauche à droite, les jets de Cazals and Pouget [2005], Li et al. [2010], RRHT_cubes et RRHT_unif utilisant le regroupement des normales issues des multiples accumulateurs.

La figure 5.16 donne un aperçu des erreurs des différentes méthodes avec un nuage de points échantillonné sur un cylindre. RRHT retrouve une normale plus fiable au niveau des angles que la régression (Jets) [Cazals and Pouget, 2005] et lisse mieux les parties régulières que la méthode de [Li et al., 2010].

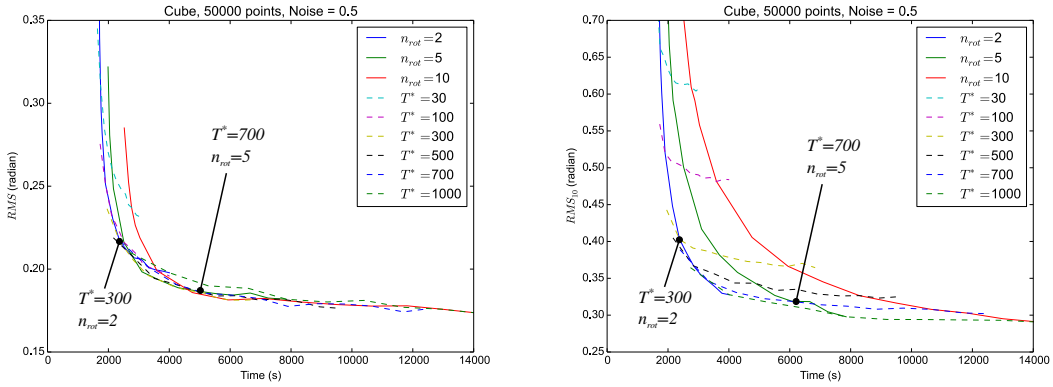
5.5.4 Influence des paramètres

La figure 5.17 présente l'influence de la taille du voisinage sur l'estimation des normales. Les normales du modèle Armadillo non bruité ont été calculées pour des tailles de voisinage de 20, 200 et 500 points. Un voisinage croissant tend à gommer les détails du modèle, comme pour les méthodes de régressions. Cependant les zones anguleuses principales (oreilles, pectoraux...) sont toujours correctement estimées. Ici, la position des points est constante, seule l'orientation des normales varie entre les trois estimations, ce qui donne une perception visuelle différente dans chaque cas.

FIGURE 5.17 – RRHT_points sur le modèle Armadillo [S3DR, 2005] pour des voisinages de tailles $K = 20$, $K = 200$ et $K = 500$. (de gauche à droite)

La figure 5.18 présente la précision en fonction du temps de calcul pour des points échantillonnés sur un cube. Pour chaque courbe soit n_{rot} (trait plein), soit T_R (pointillés) ont été fixés. Deux points sont repérés sur les courbes. Ces points, $(n_{rot} = 5, T_R = 700)$ et $(n_{rot} = 2, T_R = 300)$, sont les points utilisés dans le tableau 5.2. Le premier point

favorise la précision au prix d'un temps de calcul plus long que le second. Selon nos expérimentations, changer le modèle change aussi les pentes, mais l'allure générale ne varie pas.



Le niveau de bruit est de 0.1% *(à gauche) et* 0.5% *(à droite). Les courbes en trait plein représentent un n_{rot} constant et un T_R variable, tandis que les courbes en pointillés représentent un T_R constant et une r_{rot} variable.*

FIGURE 5.18 – RMS (à gauche) et $RMS_{\tau=10}$ (à droite) en fonction du temps de calcul pour un nuage de points de 50000 points sur un cube.

D'autres expériences montrent que le temps de calcul varie d'un facteur d'ordre 2 à 2.5 quand la taille de la grille c varie de 2 à 10 pour un gain de 25 à 30 centièmes de radian en RMS. De même, alors que k doit être le plus grand possible pour une précision accrue, la robustesse est plus importante si k est petit. Pour un k variant de 2 à 12, le temps de calcul double. Pour les modèles testés, le k optimal se situe dans cette plage (figure 5.19). Dans toutes les expériences présentées ici, c a été fixé à 4, et k à 7.

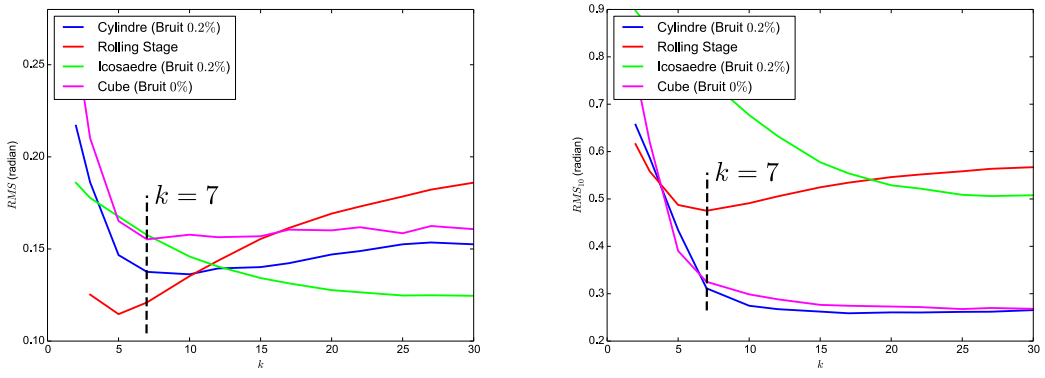


FIGURE 5.19 – Précision pour différents modèles en fonction de k, à autres paramètres fixés.

La figure 5.20 montre le résultat du calcul de normales pour un détail du modèle Statuette. Les temps de calculs sont donnés dans le tableau 5.2. Pour $(n_{rot} = 2, T_R = 300)$, l'aspect est plus granuleux, moins lissé dans les zones régulières, et les angles sont moins nets (pas assez de rotations pour découvrir le vrai maximum).

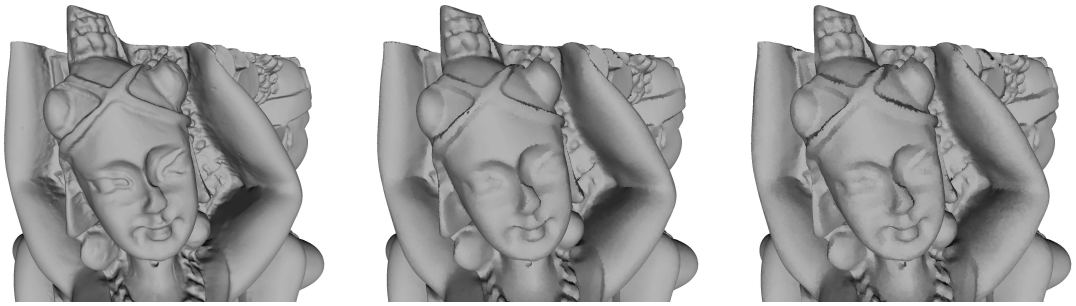


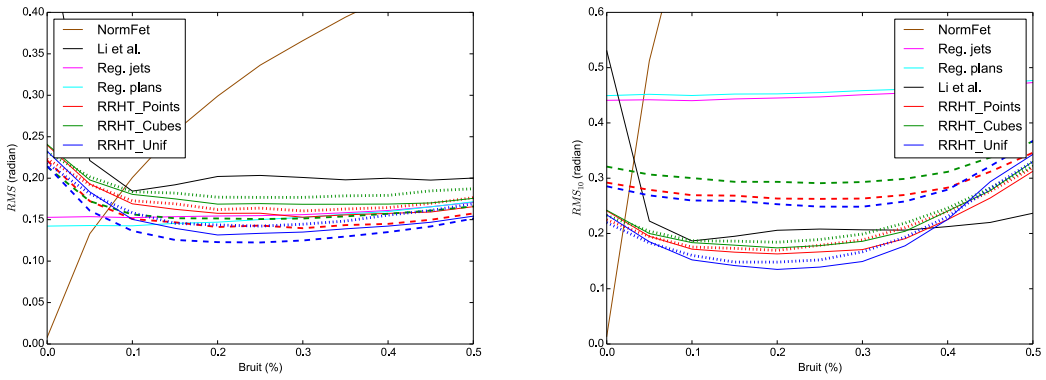
FIGURE 5.20 – Détail du modèle statuette (5M de points) pour $K = 100$. A gauche le modèle original, au centre $T_R = 500$ et $n_{rot} = 5$, et à droite $T_R = 300$ et $n_{rot} = 2$.

5.5.5 Robustesse au bruit

Pour un cylindre sur lequel sont tirés 50000 points, la figure 5.21 montre l'évolution de la précision en fonction du niveau de bruit. Les trois méthodes de choix de normales présentées dans la section 5.4.4 sont représentées par trois courbes différentes :

- **RRHT_mean**, la moyenne des normales issues des rotations de l'accumulateur, représentée par des tirets,
- **RRHT_best**, la meilleure des normales issues des rotations de l'accumulateur, représentée par des points,
- **RRHT_cluster**, la normale issue du regroupement, en trait plein.

Les algorithmes RRHT sont compétitifs quelle que soit la mesure. RRHT_m donne de meilleurs résultats pour la RMS, dus à un effet de lissage plus prononcé que les autres variantes. Cet effet attendu pour la RMS n'est plus favorisé par la RMS_10 où RRHT_b et RRHT_c sont plus performants. Il faut noter que NormFet de Dey and Goswami [2006] est la méthode la plus performante pour les très faibles bruits, pour un temps de calcul proche des autres méthodes. L'algorithme de Li donne des résultats similaires ou légèrement moins bons que les RRHT_b et RRHT_c.

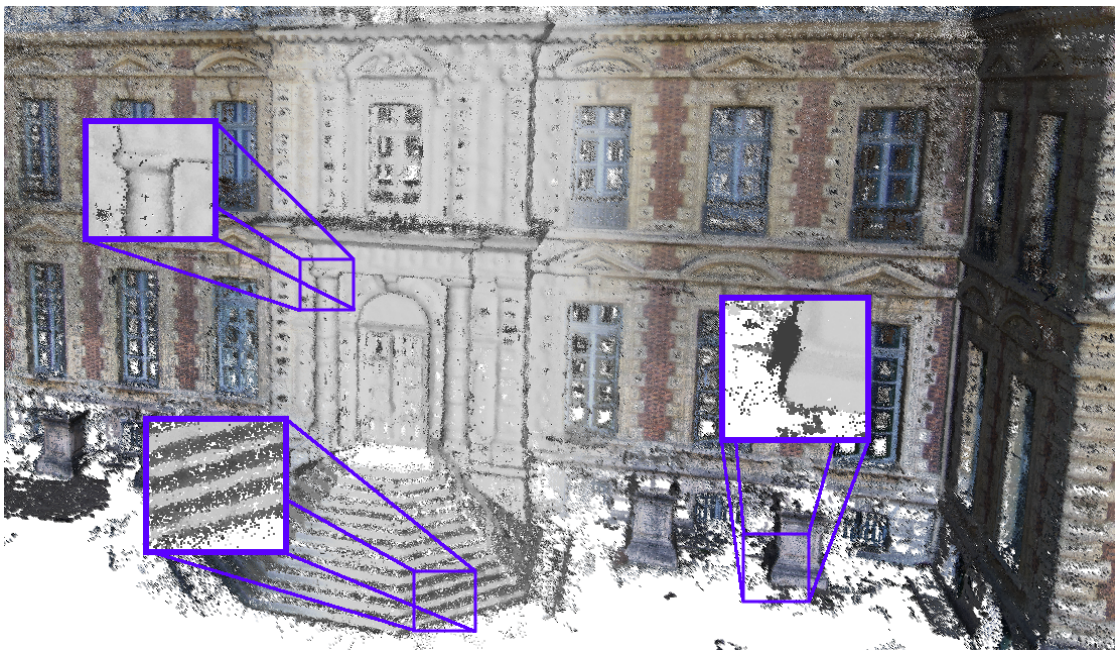


Pour les méthodes RRHT, en points la normale est la meilleure normale. Avec des tirets, elle est obtenue par moyenne et en trait plein par regroupement.

FIGURE 5.21 – Evolution de la RMS et de la $RMS_{\tau=10}$ en fonction du bruit pour un cylindre de 50000 points.

La figure 5.22 donne un visuel d'un nuage issu de photogrammétrie. La scène, le château de Sceaux, est une scène naturellement bruitée et la densité de points n'est pas

uniforme. Ici le voisinage utilisé est un rayon de recherche égal à 0.1% de la diagonale de la boîte englobante. Certains détails de la scène ont été mis en avant afin de montrer la capacité de l'estimateur à retrouver les normales dans les zones anguleuses.



RRHT_Cubes_c avec un rayon de recherche égal à 0.1% *de la diagonale de la boîte englobante.*

FIGURE 5.22 – Rendu du Château de Sceaux, nuage de points obtenu par photogrammétrie.

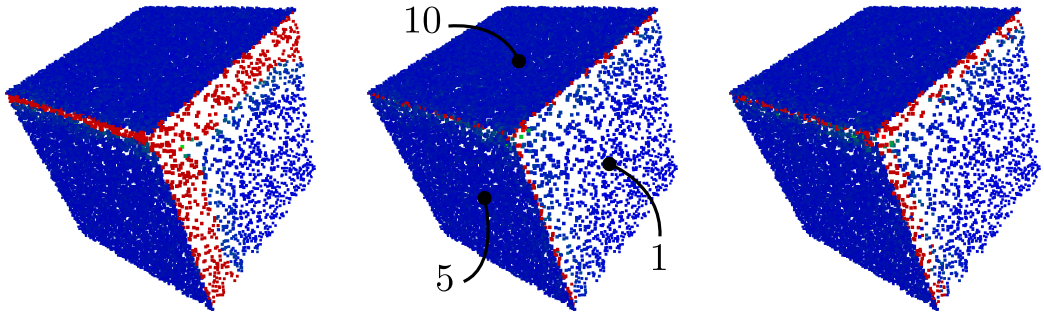
5.5.6 Robustesse à l'anisotropie

L'influence de l'anisotropie est étudiée au travers d'un modèle synthétique de coin. Chaque face est échantillonnée avec une densité différente. Pour une densité de 1 sur la face la plus clairsemée, les autres ont une densité de 5 et de 10. La figure 5.23 illustre visuellement le résultat de quatre algorithmes. Le code couleur est donné par la figure 5.15 (RMS_τ). Les points rouges, qui représentent une normale erronée, sont associés à la normale de la face la plus dense. On observe que le tirage uniforme sur les points n'est pas robuste à l'anisotropie contrairement au tirage quasi uniforme. Le tirage quasi uniforme discrétisé se situe quant à lui en les deux.

Les courbes de la figure 5.24 montrent l'évolution de la RMS_10 en fonction du bruit pour un coin anisotrope. Comme attendu RRHT_Points n'est pas robuste à l'anisotropie, son score est très proche de celui de Li et al. [2010], tandis que RRHT_Unif remplit quant à lui l'objectif d'être insensible à la variation de densité. RRHT_Cubes est un bon compromis entre les deux variantes précédentes. La méthode de Dey and Goswami [2006] est, là aussi, la méthode à favoriser en cas de bruit faible.

5.5.7 Robustesse aux points aberrants

Du fait que la normale retenue est un pic dans une distribution statistique, la présence de points aberrants dans le voisinage d'un point p a une influence modérée, contrairement aux méthodes de régression qui utilisent sans distinction tous les points du voisinage. Pour illustrer cette robustesse aux points aberrants, le modèle Armadillo est bruité à 0.2%,



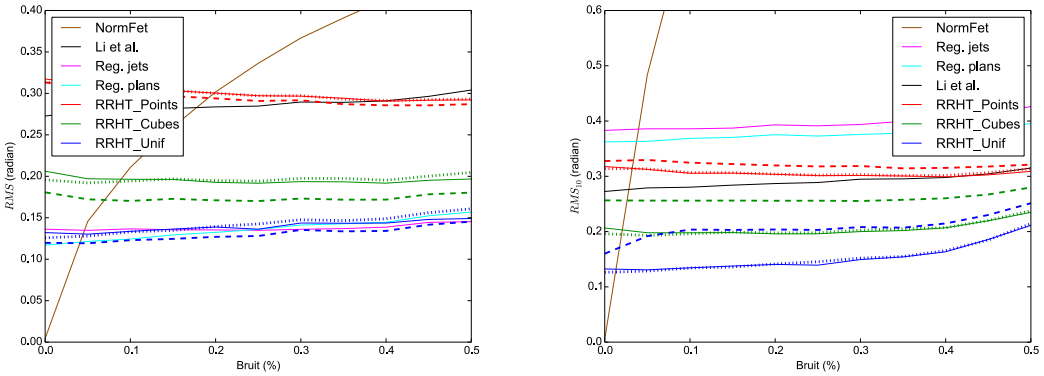
(a) Tirage uniforme sur les points du voisinage.

(b) Tirage quasi uniforme dans la boule du voisinage.

(c) Discrétisation de la boule du voisinage.

Le modèle est un coin, échantillonné avec 20000 *points. Les densités de points par faces sont* 1, 5 *et* 10.

FIGURE 5.23 – Trois manières différentes pour tirer les triplets de points. Les points en rouge sont ceux pour lesquels la normale est mal estimée.

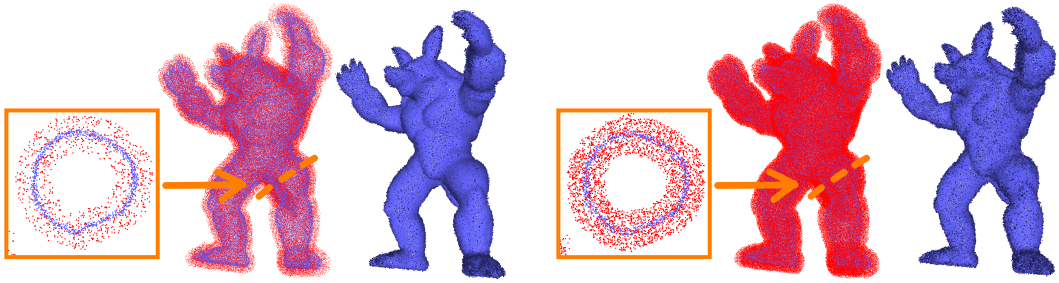


Tirets, ***RRHT_mean****. Points,* ***RRHT_best****. Trait plein,* ***RRHT_cluster****.*

FIGURE 5.24 – Évolution de la RMS et de la $RMS_{\tau=10}$ pour un coin dont les faces sont échantillonnées avec des densités différentes (figure 5.23).

puis on ajoute 100% de points aberrants (figure 5.25), ceux-ci étant tirés aléatoirement à une distance au plus $R = 0.3\%$ de la diagonale de la boîte englobante. R est aussi le rayon de recherche de voisinage. Cette limitation pour les points aberrants s'explique par le fait qu'un point situé à une distance de plus de R des points du nuage n'aura pas d'incidence sur l'estimation de la normale pour les points du nuage.

La mesure RMS avec 0.2% de bruit et aucun point aberrant est de 0.39 radian. Lorsqu'on ajoute 100% de points aberrants, elle augmente de 0.04 radian. Ce chiffre passe à 0.15 radian lorsque l'ajout de points aberrants passe de 100% à 300%. Les points du modèle les plus dégradés sont les zones anguleuses, ceci s'explique par le profil de la distribution en ces points. Celle-ci est moins piquée que dans une zone plane et un grand nombre de points aberrants introduit un bruit sur la distribution englobant ses principaux pics.



Armadillo avec 0.2% *de bruit et* 100% *de points aberrants (à gauche),* 300% *à droite. Le rayon de recherche est de* $R = 0.3\%$ *de la diagonale de la boîte englobante.*

FIGURE 5.25 – Robustesse de RRHT aux points aberrants.

Robustesse à ...	Bruit	Points aberrants	Points anguleux	Anisotropie	Rapidité
Régression planaires	✓				✓
Régression Jet	✓				✓
NormFet (Dey et al.)			✓	✓	✓
RANSAC (Li et al.)	✓	✓	✓		
RRHT	✓	✓	✓	✓	✓

TABLE 5.3 – Résumé des capacités des algorithmes utilisés pour la comparaison et de la transformée Hough aléatoire et robuste pour l'estimation de normales (RRHT).

5.6 Conclusion

La transformée de Hough aléatoire et robuste, appliquée à l'estimation de normales, est une nouvelle méthode qui offre de bons compromis entre qualité et performance. Elle est robuste au bruit, aux points aberrants et capable de reconstruire fidèlement les normales dans les zones anguleuses, le tout en gagnant près d'un ordre de grandeur en temps de calcul par rapport à l'état de l'art (Li & al). Proposée en plusieurs variantes, au prix d'un temps de calcul plus long, l'algorithme s'accommode aussi d'une forte anisotropie de densité de points. Le tableau 5.3 résume les points forts de chacun des algorithmes utilisé en comparaison de RRHT.

Travaux ultérieurs

Les résultats présentés dans ce chapitre et le précédent ont été publiés en 2013 et présentés au Symposium on Geometry Processing 2012. par la suite, [Zhang et al., 2013] ont proposé une méthode basée sur une segmentation en sous espaces de faible rang. Cette méthode donne de meilleurs résultats au niveau des zones anguleuses lorsque les angles dihédraux sont faibles. Cependant le prix est un temps de calcul beaucoup plus long, de l'ordre de 20 fois supérieur.

Remerciements

Merci à Pierre Alliez pour ses conseils, à Bao Li et Tamal Dey pour avoir mis le code de leurs estimateurs à disposition. Les modèles Armadillo, Lucy, Dragon, Buddha, et Statuette sont disponibles à [S3DR, 2005]. Les nuages de points et maillages de Circular Box et Omotondo sont issus de [aim@shape, 2006]. Merci à Pierre Moulon pour le modèle de photogrammétrie et à Arnak Dalalyan pour ses conseils précieux en statistiques.

Publication

Ce travail a été présenté au Symposium On Geometry Processing 2012, à Tallin, Estonie. Il a été publié dans la revue Computer Graphics Forum :

Alexandre Boulch, Renaud Marlet.
Fast and Robust Normal Estimation for Point Clouds with Sharp Features.
Comput. Graph. Forum 31(5): 1765-1774 (2012)

Bibliographie

aim@shape (2006). Aim@Shape Repository. http://www.aimatshape.net/.

Akinlar, C. and Topal, C. (2011). EDLines: A real-time line segment detector with a false detection control. *Pattern Recognition Letters*, 32(13):1633–1642.

Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D., and Silva, C. T. (2001). Point set surfaces. In *IEEE Visualization*.

Alliez, P., Cohen-Steiner, D., Tong, Y., and Desbrun, M. (2007). Voronoi-based variational reconstruction of unoriented point sets. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 39–48. Eurographics Association.

Ballard, D. H. (1981). Generalizing the Hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2):111–122.

Barinova, O., Lempitsky, V. S., and Kohli, P. (2010). On detection of multiple object instances using Hough transforms. In *CVPR*, pages 2233–2240.

Barinova, O., Lempitsky, V. S., and Kohli, P. (2012). On detection of multiple object instances using Hough transforms. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(9):1773–1784.

Borrmann, D., Elseberg, J., Lingemann, K., and Nüchter, A. (2011). The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, 2(2):1–13.

Bughin, E. (2011). *Towards automated, precise and validated vectorisation of disparity maps in urban satellites stereoscopy*. PhD thesis, ENS Cachan.

Cazals, F. and Pouget, M. (2005). Estimating differential quantities using polynomial fitting of osculating jets. *Computer Aided Geometric Design*, 22(2):121–146.

Censi, A. and Carpin, S. (2009). HSM3D: Feature-less global 6DOF scan-matching in the Hough/Radon domain. *IEEE International Conference on Robotics and Automation*, pages 3899–3906.

Chaperon, T. and Goulette, F. (2001). Extracting cylinders in full 3D data using a random sampling method and the gaussian image. In *VMV*, pages 35–42.

Chauve, A.-L., Labatut, P., and Pons, J.-P. (2010). Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *CVPR*, pages 1261–1268.

Cook, J. (1957). Rational formulae for the production of a spherically symmetric probability distribution. *Mathematics of Computation*, 11(58):81–82.

Davies, E. R. (1988). Application of the generalised Hough transform to corner detection. *Computers and Digital Techniques, IEE Proceedings E*, 135(1):49–54.

Dey, T. K. and Goswami, S. (2006). Provable surface reconstruction from noisy samples. *Journal of Computational Geometry*, 35(1-2):124–141.

Duda, R. O. and Hart, P. E. (1972). Use of the Hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15.

Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Gall, J. and Lempitsky, V. S. (2009). Class-specific hough forests for object detection. In *CVPR*, pages 1022–1029.

Guennebaud, G. and Gross, M. H. (2007). Algebraic point set surfaces. *ACM Transactions on Graphics (TOG)*, 26(3):23.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. A., and Stuetzle, W. (1992). Surface reconstruction from unorganized points. In *SIGGRAPH*, pages 71–78.

Hough, P. V. C. (1962). Method and means for recognizing complex patterns. *U.S. Patent*, 3.069.654.

Huang, H., Li, D., Zhang, H., Ascher, U. M., and Cohen-Or, D. (2009). Consolidation of unorganized point clouds for surface reconstruction. *ACM Transactions on Graphics (TOG)*, 28(5).

Illingworth, J. and Kittler, J. (1988). A survey of the Hough transform. *Computer Vision, Graphics, and Image Processing*, 44(1):87–116.

Jones, T. R., Durand, F., and Zwicker, M. (2004). Normal improvement for point rendering. *IEEE Computer Graphics and Applications*, 24(4):53–56.

Kiryati, N., Eldar, Y., and Bruckstein, A. M. (1991). A probabilistic Hough transform. *Pattern Recognition*, 24(4):303–316.

Knopp, J., Prasad, M., Willems, G., Timofte, R., and Gool, L. J. V. (2010). Hough transform and 3D SURF for robust three dimensional classification. In *ECCV (6)*, pages 589–602.

Li, B., Schnabel, R., Klein, R., Cheng, Z.-Q., Dang, G., and Jin, S. (2010). Robust normal estimation for point clouds with sharp features. *Computers & Graphics*, 34(2):94–106.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110.

Mitra, N. J., Nguyen, A., and Guibas, L. J. (2004). Estimating surface normals in noisy point cloud data. *Int. J. Comput. Geometry Appl.*, 14(4-5):261–276.

Nistér, D. (2005). Preemptive RANSAC for live structure and motion estimation. *Machine Vision and Applications*, 16(5):321–329.

Okada, R. (2009). Discriminative generalized Hough transform for object dectection. In *ICCV*, pages 2000–2005.

Öztireli, A. C., Guennebaud, G., and Gross, M. H. (2009). Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum*, 28(2):493–501.

Pauly, M., Keiser, R., Kobbelt, L., and Gross, M. H. (2003). Shape modeling with point-sampled geometry. *ACM Trans. Graph.*, 22(3):641–650.

Pham, M.-T., Woodford, O. J., Perbet, F., Maki, A., Stenger, B., and Cipolla, R. (2011). A new distance for scale-invariant 3D shape recognition and registration. In *ICCV*, pages 145–152.

Roth, G. and Levine, M. (1993). Extracting Geometric Primitives. *CVGIP: Image Understanding*, 58(1):1–22.

Rusinkiewicz, S. and Levoy, M. (2000). Qsplat: a multiresolution point rendering system for large meshes. In *SIGGRAPH*, pages 343–352.

S3DR (2005). Stanford 3d scanning repository. https://graphics.stanford.edu/data/3Dscanrep/.

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226.

Shen, F. and Wang, H. (2002). Corner detection based on modified Hough transform. *Pattern Recognition Letters*, 23(8):1039 – 1049.

Toldo, R. and Fusiello, A. (2008). Robust multiple structures estimation with J-Linkage. In *ECCV (1)*, pages 537–547.

Torr, P. H. S. and Zisserman, A. (1998). Robust computation and parametrization of multiple view relations. In *ICCV*, pages 727–732.

Torr, P. H. S. and Zisserman, A. (2000). MLESAC: A new robust estimator with application to estimating image geometry. *Computer Vision and Image Understanding*, 78(1):138–156.

Tsuji, S. and Matsumoto, F. (1978). Detection of ellipses by a modified Hough transformation. *IEEE Trans. Computers*, 27(8):777–781.

von Gioi, R. G., Jakubowicz, J., Morel, J.-M., and Randall, G. (2008). On straight line segment detection. *Journal of Mathematical Imaging and Vision*, 32(3):313–347.

von Gioi, R. G., Jakubowicz, J., Morel, J.-M., and Randall, G. (2010). LSD: A fast line segment detector with a false detection control. *IEEE Trans. Pattern Anal. Mach. Intell.*, 32(4):722–732.

Weisstein, E. W. (s.i.t.e.). Wolfram MathWorld: Sphere point picking. http://mathworld.wolfram.com/SpherePointPicking.html.

Xu, L. and Oja, E. (2009). Randomized hough transform. In *Encyclopedia of Artificial Intelligence*, pages 1343–1350. IGI Global.

Xu, L., Oja, E., and Kultanen, P. (1990). A new curve detection method: Randomized Hough transform (RHT). *Pattern Recognition Letters*, 11(5):331–338.

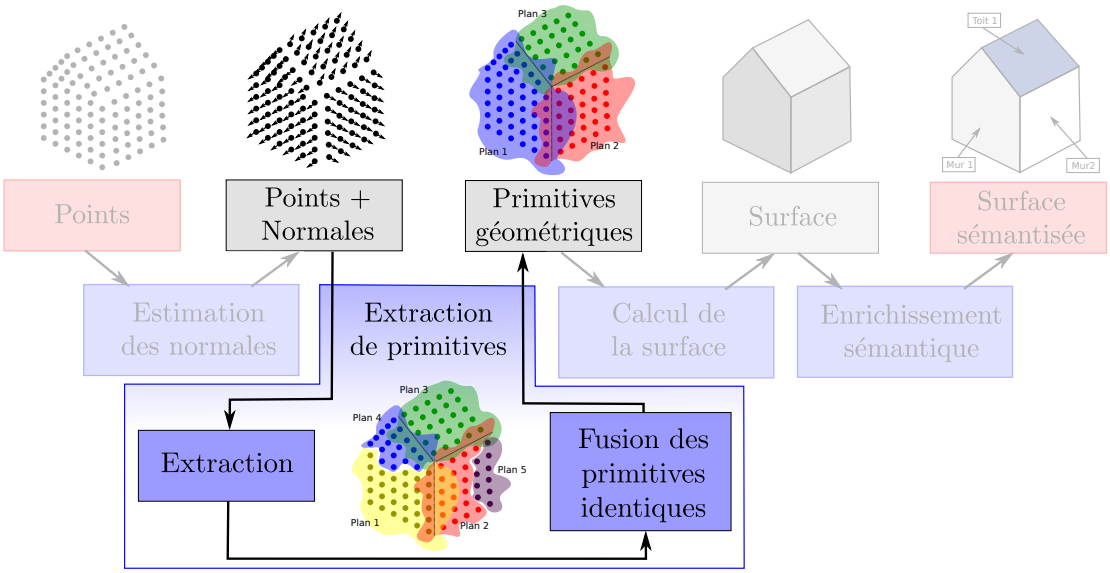
Yoon, M., Lee, Y., Lee, S., Ivrissimtzis, I. P., and Seidel, H.-P. (2007). Surface and normal ensembles for surface reconstruction. *Computer-Aided Design*, 39(5):408–420.

Zaharia, T. B. and Prêteux, F. J. (2002). Shape-based retrieval of 3D mesh models. pages 437–440.

Zhang, J., Cao, J., Liu, X., Wang, J., Liu, J., and Shi, X. (2013). Point cloud normal estimation via low-rank subspace clustering. *Computers and Graphics*, 37(6):697 – 706. Shape Modeling International (SMI) Conference 2013.

Deuxième partie

Primitives géométriques et nuages de points



Chapitre 6

Extraction de primitives

6.1 Introduction

La géométrie d'une maquette numérique est une version abstraite de la réalité. Hormis pour quelques objets aux formes compliquées et représentés par des maillages, les éléments d'une maquette sont construits à l'aide de formes géométriques simples.

Pour de nombreux algorithmes, l'extraction de primitives géométriques simples, telles que les plans, les sphères, les cylindres, etc. est une première étape en vue de reconstruire une surface idéalisée. C'est par exemple le cas dans [Lafarge and Alliez, 2013; Li et al., 2011]. Cette première étape est cruciale et conditionne généralement les résultats des algorithmes utilisés pour la reconstruction de la surface.

De plus, avec les développements des dispositifs d'acquisition et de la puissance de calcul, les nuages de points créés sont de plus en plus volumineux. La découverte de primitives permettant de remplacer des ensembles de points cohérents est une manière de compresser les données. De telles primitives sont décrites par peu de paramètres, les mémoriser est moins coûteux que de retenir la position de tous les points qu'elles représentent.

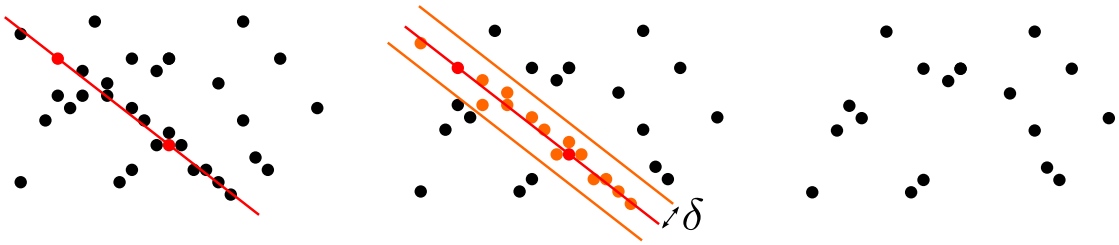
Dans cette thèse, les algorithmes de détection de primitives dans des nuages de points sont des outils. Les primitives détectées seront par la suite fusionnées en cas de sur-segmentation (chapitre 7) et utilisées pour reconstruire des surfaces planaires par morceaux (partie III). Ce chapitre présente principalement les deux méthodes que nous employons dans les chapitres suivants : le RANSAC et la croissance de régions.

6.2 RANSAC

Le RANSAC ou RANdom SAmple Consensus est l'une des méthodes les plus utilisées pour la sélection de modèles (section 4.2). Il a été appliqué à une vaste gamme de problèmes, de la détection de primitives géométriques [Schnabel et al., 2007] à la calibration de caméras [Moulon et al., 2012].

L'idée générale du RANSAC est de construire un ensemble d'hypothèses de formes, par exemple, des plans, et d'évaluer le nombre de points qui sont à une faible distance (inférieure à un seuil) de la forme. Ensuite, la forme correspondant à l'ensemble de points de plus grand cardinal est considérée comme la meilleure. D'abord utilisé par Fischler and Bolles [1981] pour rechercher une unique forme dans les données, RANSAC a ensuite servi à y détecter plusieurs occurrences du modèle.

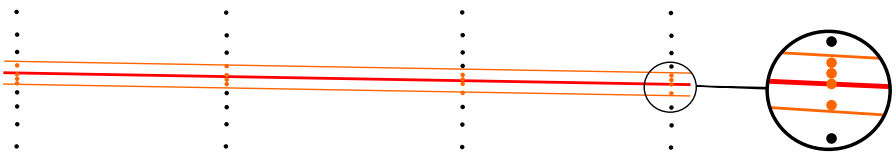
Pour extraire plusieurs primitives, le RANSAC séquentiel ([Kanazawa and Kawakami, 2004; Vincent and Laganière, 2001]) est simplement basé sur la répétition de l'algorithme précédent. La meilleure hypothèse est sélectionnée par RANSAC. Puis les points qui lui sont associés sont retirés et on réitère l'opération sur les données restantes. La figure 6.1 illustre ce procédé.



Tirage de deux points dans le nuage (à gauche). Au centre, détection des points validant le modèle de la ligne, c.-à-d. à distance inférieure à la tolérance δ. Si cette ligne est la meilleure, les points concernés sont retirés du nuage (à droite) avant de réitérer l'opération.

FIGURE 6.1 – Illustration du principe de l'algorithme RANSAC séquentiel sur l'exemple de l'extraction de lignes.

Cependant cette approche gloutonne n'assure pas une bonne détection dans tous les cas. Pour la détection de plans en 3D, il est possible de retenir des hypothèses de plan erronées. En effet, il arrive que des hypothèses de plans non-souhaitables regroupent plus de points que les plans voulus. Or, une fois retirés, les points ne peuvent plus être réattribués, exposant l'utilisateur non seulement à de mauvaises détections, mais aussi à des détections manquantes. La figure 6.2 illustre ce cas. Quatre lignes sont échantillonnées avec une densité plus importante dans la zone centrale. Du fait de cette variation de densité, le meilleur candidat n'est pas une des lignes échantillonnées.



Nuage de points échantillonnant quatre lignes verticales, avec une densité variable. La meilleure hypothèse pour le nombre de points est une ligne quasiment horizontale.

FIGURE 6.2 – Illustration d'une mauvaise détection possible.

Pour pallier cet inconvénient majeur, plusieurs solutions ont été proposées. Zuliani et al. [2005] estiment plusieurs modèles à la fois mais cela implique de connaître préalablement le nombre de primitives à extraire. Zhang and Kosecká [2006] proposent une méthode pour évaluer ce paramètre.

Rabin et al. [2010] utilisent le RANSAC a contrario de [Moisan and Stival, 2004] associé à un critère de validation statistique permettant de détecter plusieurs groupes à la fois. Développé pour la détection d'objets similaires entre deux images, il ne prend en compte qu'un faible nombre de points d'intérêt détectés dans les images. La méthode n'est pas adaptée pour des nuages de points beaucoup plus volumineux.

Pour traiter ces gros ensembles de données, Forlani et al. [2003] se basent sur le fait que des points voisins sont généralement associés à la même primitive. Après une

croissance de régions (section 6.3) pour obtenir un ensemble de points pouvant appartenir à la même primitive, RANSAC est appliqué sur les points de chaque région.

Schnabel et al. [2007] utilisent aussi cette notion de voisinage. Des triplets de points (dans le cas de plans) sont sélectionnés dans un même voisinage, réduisant ainsi le nombre de tirages à effectuer pour obtenir une hypothèse valide. De plus, comme dans [Labatut et al., 2009], les points y sont associés à une information de normale. L'angle ainsi que la distance à la primitive sont pénalisés afin de ne conserver que des points dont l'orientation locale est cohérente.

Méthode utilisée

La variante de RANSAC utilisée dans la suite de ce document est celle de Schnabel et al. [2007]. Cet algorithme s'applique à tous types de nuages de points 3D. Il permet de détecter une grande variété de primitives géométriques telles que des plans, des cylindres, des cônes ou des sphères. Il offre une bonne robustesse au bruit tout en garantissant une grande vitesse d'exécution. Celle-ci est atteinte en réduisant l'espace de recherche en favorisant les points voisins pour la création des hypothèses de surface. La figure 6.3 montre le résultat de l'extraction de plans et de cylindres sur des modèles photogrammétriques.

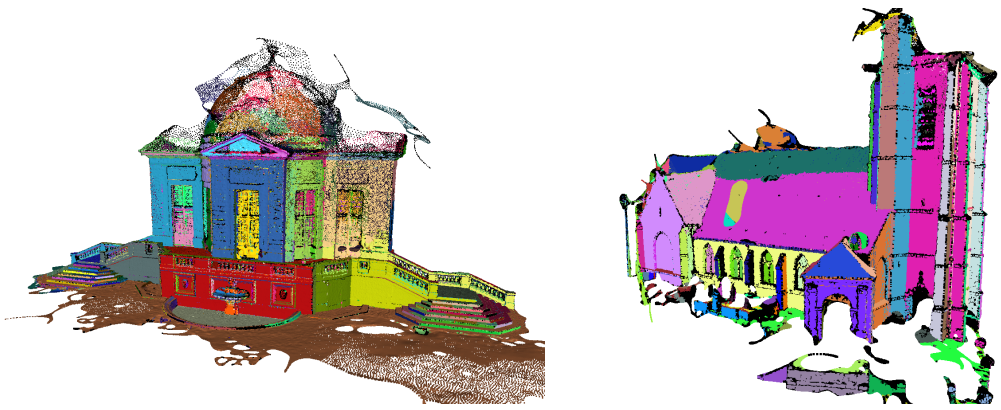


FIGURE 6.3 – Résultat de segmentation en plan par RANSAC [Schnabel et al., 2007] sur deux nuages de points créés par photogrammétrie.

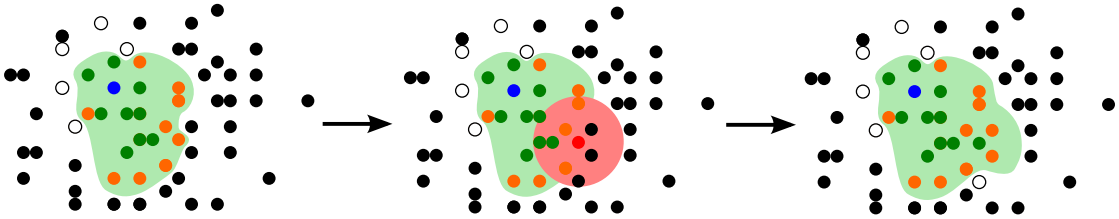
6.3 Croissance de régions

La croissance de régions est une méthode générale pour la création d'ensembles connexes dont les éléments partagent des caractéristiques communes.

Généralement, la segmentation souhaitée des données par l'utilisateur est composée d'ensembles connexes. C'est, par exemple, le cas lors de la segmentation d'objets dans des images où un objet apparaît en un bloc (occlusions mises à part). L'idée de la croissance de régions est de partir d'un élément des données, la graine, en accord avec un modèle (couleur, géométrie ...) et d'étendre, de proche en proche, la région dans le voisinage de la graine.

La figure 6.4 illustre le principe de la méthode. La région est représentée en vert clair. La frontière (les points en orange sur la figure) correspond aux points qui ont été précédemment observés et déclarés comme correspondants au modèle, mais dont on n'a pas encore observé le voisinage pour étendre la région. Sur cette région un point est choisi

(en rouge) et son voisinage est calculé. Les points de ce voisinage qui n'ont pas encore été traités sont testés par rapport au modèle. S'ils passent le test, ils sont ajoutés à la frontière et la région s'étend. Sinon, ils sont écartés.



En bleu, la graine. En vert, les points observés inclus dans le modèle. En orange la frontière active (les points validant le modèle dont on n'a pas encore étudié le voisinage). En rouge, le point courant. En blanc cerclés de noir, les points observés ne validant pas le modèle.

FIGURE 6.4 – Principe de la croissance de régions. Sur la frontière active, un point est choisi (au centre), dans son voisinage les points validant le modèle sont ajoutés à la régions, les autres sont écartés (à droite).

La croissance continue jusqu'à ce qu'aucun point ne puisse plus être ajouté à la région (c.-à-d. que la taille de la frontière soit nulle). La conformité de la région (nombre d'éléments, aspect ...) est alors testée pour décider de son rejet ou non. Les points sont alors retirés du nuage, et l'algorithme est relancé sur les données restantes.

La croissance de régions a été particulièrement utilisée en segmentation d'images dont [Hojjatoleslami and Kittler, 1998; Shih and Cheng, 2005; Trémeau and Borel, 1997] ne sont que des exemples. Sur les images, le voisinage d'un pixel est aisément accessible, rendant les algorithmes rapides.

Initiée par Besl and Jain [1988], la croissance de régions pour la segmentation des images de profondeur s'est vu appliquée à différentes scènes et modèles. Sur de telles images, les voisinages sont directement accessibles, comme dans le cas des images usuelles. Fitzgibbon et al. [1997] utilisent la courbure locale pour choisir les graines. C'est aussi le cas en 3D dans [Digne et al., 2010] qui utilisent la connectivité d'une triangulation pour segmenter des maillages.

Dans [Poppinga et al., 2008; Pu and Vosselman, 2006], les régions recherchées sont planaires et c'est la distance au plan de régression, calculé sur l'ensemble des points de la région, qui est le critère d'arrêt.

Bughin et al. [2010] estiment le niveau de bruit de chaque région au fur et à mesure de la croissance, pour déterminer automatiquement le critère d'arrêt. Une idée similaire se retrouve dans la reconstruction de villes de Poullis and You [2009] où, de plus, une information de locale de normale est prise en compte.

Chauve et al. [2010] reconstruisent une surface planaire par morceaux. L'extraction de plans dans le nuage de points non structuré y est effectuée par croissance de régions. Pour ce faire, le voisinage des points est calculé en utilisant un kd-tree.

Méthode utilisée

Les acquisitions laser par balayage horizontal et vertical présentent une structure proche des images de profondeur précédemment citées. La différence réside dans la surface de reprojection des données, une sphère pour les premières, un plan pour les secondes.

Le calcul du voisinage de chaque point étant immédiat, cela favorise un algorithme de segmentation très rapide. Dans la suite, nous utilisons la 8-connectivité de l'image de profondeur.

Nous utilisons une méthode proche de celles précédemment décrites. Les graines peuvent être générées de deux manières. Soit elles sont tirées aléatoirement, soit elles sont classées d'après la planarité du voisinage (le plan de régression étant calculé via analyse en composantes principales).

Des exemples de croissances de régions sur des images laser sont donnés figure 6.5.

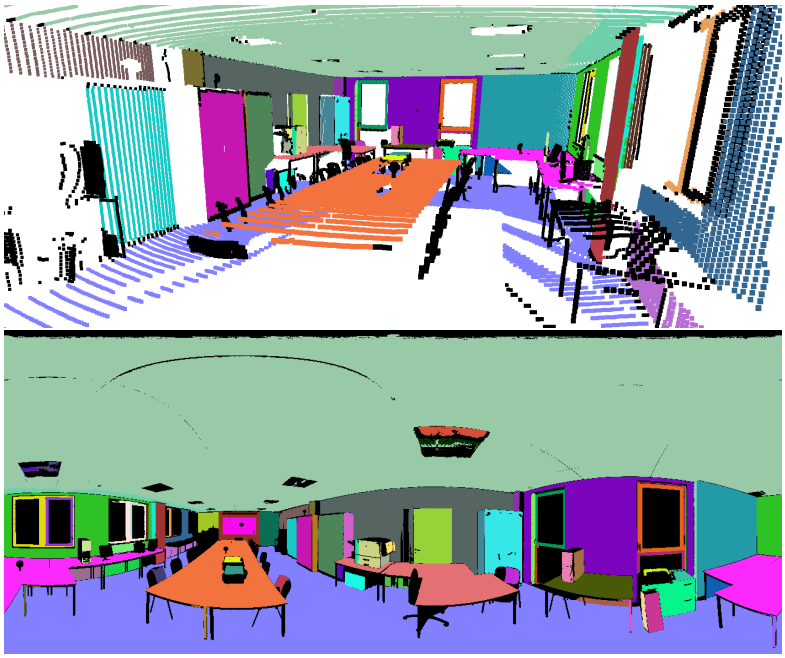
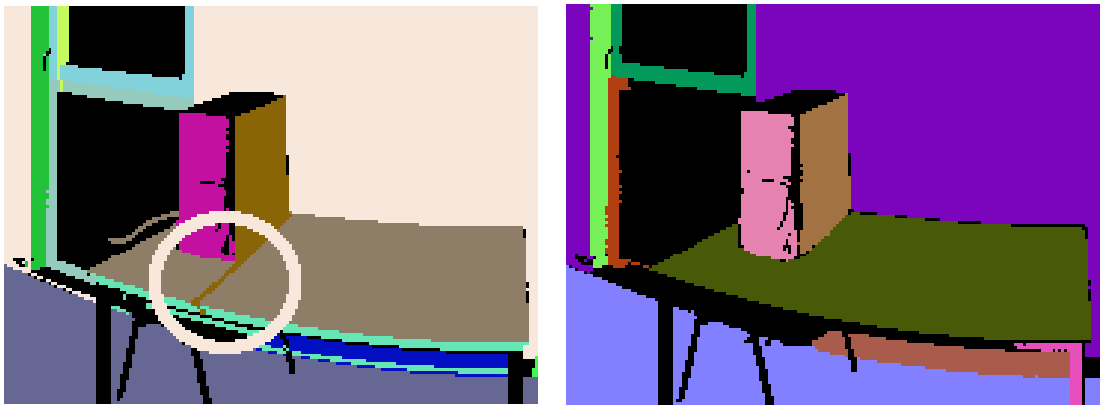


FIGURE 6.5 – Exemple de croissance de régions : vue du nuage 3D (en haut) et image laser (en bas), une couleur différente par segment.

De plus, pour éviter la propagation d'une région dans une zone d'orientation différente, nous utilisons un seuil sur l'angle entre la normale au point et le plan. La figure 6.4 illustre la nécessité de cette limitation, particulièrement dans le cas des acquisitions laser où de nombreux points se retrouvent alignés.



(a) Sans seuil. (b) Avec Seuil.

FIGURE 6.6 – Nécessité du seuil sur l'angle entre la normale et la primitive.

6.4 Autres méthodes

Il existe d'autres méthodes d'extraction de primitives 3D. Elles sont, en général, des adaptations d'algorithmes développées pour la reconnaissance de formes dans des images 2D. Le chapitre 4 sur la transformée de Hough présente une partie de ces algorithmes. La présente section, reprend la liste de ces méthodes dans une optique 3D.

Transformée de Hough

Introduite par Hough [1962], elle a connu un développement important au fil des années, pour détecter différentes formes dans les images. En 3D, elle est utilisée principalement pour l'extraction de primitives telles que les plans [Borrmann et al., 2011], des éléments sphériques [Han et al., 1987], ou encore en reconnaissance de formes 3D [Knopp et al., 2010; Pham et al., 2011].

Dans un contexte plus spécialisé, Overby et al. [2004]; Vosselman and Dijkman [2001] utilisent la transformée de Hough pour extraire les toits des bâtiments dans des images aériennes de profondeur.

Borrmann et al. [2011] proposent un résumé des variantes de transformée de Hough ainsi qu'une présentation des différents accumulateurs.

J-linkage

Le J-linkage [Toldo and Fusiello, 2008] est une méthode de séparation en ensembles cohérents. Comme la transformée de Hough, le J-linkage recherche plusieurs formes à la fois. Des hypothèses valides sont générées à la manière d'un RANSAC et sont agrégées pour ne retenir que les clusters prépondérants. Toldo and Fusiello [2008] l'ont utilisé pour la détection de lignes et de plans dans des nuages de points non structurés. Cette méthode a ensuite été reprise pour divers modèles et applications, comme par exemple la détection de plans dans des paires d'images [Fouhey et al., 2010].

Approche a contrario

Bughin [2011] couple une croissance de régions avec un critère a contrario pour la détection de plans dans des images de profondeur.

Chapitre 7

Fusion statistique de surfaces

7.1 Introduction

De nombreux algorithmes de reconstruction de surfaces à partir de nuages de points ont été développés au cours de ces dernières années. [Chauve et al., 2010; Lafarge and Alliez, 2013; Li et al., 2011] n'en sont que quelques exemples. Une grande partie de ces algorithmes se basent sur une étape de détection de formes géométriques simples, généralement des plans mais aussi parfois des cylindres, des sphères ou des cônes. Comme présenté aux chapitres 6 et 4, il existe plusieurs familles d'algorithmes d'extraction de primitives, dont les plus utilisées sont le RANSAC [Schnabel et al., 2007], la transformée de Hough [Hough, 1962; Knopp et al., 2010; Pham et al., 2011] et la croissance de régions [Bughin et al., 2010; Chauve et al., 2010].

Cette étape d'extraction est, en général, une brique importante de l'algorithme et la qualité des primitives influence directement le résultat final ainsi que le temps de calcul.

L'utilisateur doit souvent trouver un compromis sur le nombre de segments à extraire. Un grand nombre de segments favorisant la précision tandis qu'un nombre réduit de primitives est synonyme de temps de calcul décru. De plus, une détection fine s'accompagne fréquemment d'une sur-segmentation.

Ce phénomène apparaît aussi dans le cas de la croissance de régions où les segments sont des ensembles connexes. Si la primitive est constituée de plusieurs composantes connexes, il en résulte plusieurs segments. La figure 7.1 montre le résultat d'une croissance de régions sur un nuage laser. Les flèches blanches pointent sur des segments différents mais appartenant à un même plan.

Ainsi l'extraction de primitives est souvent suivie d'une étape de fusion visant à regrouper les segments appartenant à la même primitive.

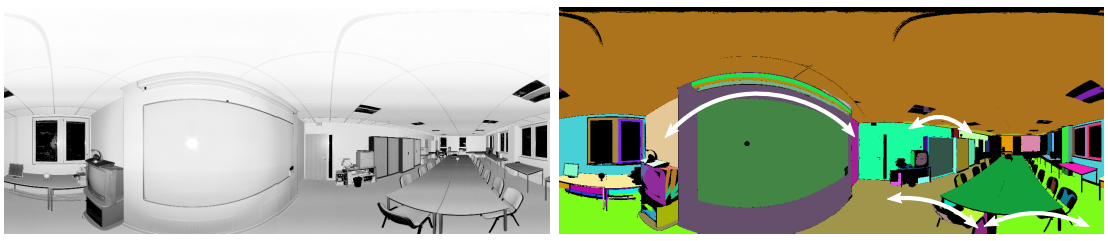


FIGURE 7.1 – Croissance de régions et sur-segmentation. À gauche, image laser, à droite résultat de la croissance de régions (une couleur par segment détecté).

Les critères de fusion sont généralement très dépendants du type de primitives utilisées (plans, cylindres...) et difficilement généralisables à plusieurs sortes de surface sans une multiplication des paramètres. Ce chapitre étudie une nouvelle approche pour la fusion de primitives basée sur des tests statistiques. Notre méthode traite de surfaces quelconques et s'applique aux nuages de points non structurés. Son principal avantage est de limiter au maximum la connaissance des surfaces : seule une fonction distance est requise.

7.2 Travaux précédents

La plupart des travaux réalisés jusqu'à présent se concentrent sur la fusion de plans. Les plans sont les primitives les plus communément utilisées car les plus élémentaires (à une certaine échelle, toute surface régulière peut être approximée par un ensemble de plans).

Vosselman et al. [2004] fusionnent des plans d'après la similitude de leurs équations. Cela se ramène à la comparaison des normales de chaque plan et des distances respectives des plans à un point de l'espace donné (généralement l'origine). Pour les plans, ceci met en jeu deux paramètres, un angle et une distance. Cette méthode s'étend à toute surface paramétrique, mais lorsque ces surfaces sont complexes, il peut y avoir un grand nombre de paramètres. Ceci rend l'algorithme difficile à régler, particulièrement lorsque les paramètres ne représentent pas les mêmes grandeurs physiques.

Pour gérer le problème de paramètres, Bughin [2011] propose une méthode a contrario pour décider si un ensemble de points peut être interprété comme un plan. Cependant, cette méthode est spécifique aux plans et aux images de profondeur.

Aspert et al. [2002] comparent tous types de surfaces, tels que des maillages. Pour ce faire, ils utilisent la distance de Hausdorff, qui repose sur un unique paramètre. Cependant, les points aberrants, potentiellement loin des surfaces, peuvent grandement influer sur le résultat. De plus, déterminer rapidement le maximum des distances entre chaque surface n'est pas chose aisée et peut s'avérer coûteux en temps de calcul, d'autant plus lorsque les nuages sont volumineux.

7.3 Présentation de la méthode

Afin d'être robuste aux points aberrants, il est possible d'utiliser des tests non-paramétriques issus des statistiques robustes. De telles approches ont déjà été utilisées pour de la segmentation d'images [Fiorio and Nock, 2000; Nock and Nielsen, 2005; Wolf et al., 2006].

Les tests de Mann-Whitney [Mann and Whitney, 1947; Wilcoxon, 1945] et de Kolmogorov-Smirnov [Kolmogorov, 1933; Smirnov, 1948] seront utilisés ici pour la fusion de surfaces. Ces tests se basent sur la comparaison des distributions de distances aux primitives. Ils utilisent deux paramètres et une fonction distance pour chacune des surfaces observées (seule connaissance préalable à l'utilisation des tests).

Pour accélérer le calcul, l'inégalité de Dvoretsky-Kiefer-Wolfowitz [Dvoretzky et al., 1956] permet de déterminer le nombre minimal de points nécessaires pour estimer correctement les distributions de distances.

Enfin, les tests pouvant rejeter des fusions que l'utilisateur voudrait accepter, un paramètre supplémentaire, homogène à une distance, est introduit pour permettre le réglage du niveau d'acceptation de la fusion.

7.4 Objectif

Soient deux ensembles de points P_1 et P_2 associés respectivement à deux surfaces $\mathcal{S}_1$ et $\mathcal{S}_2$. L'objectif est de dire si les deux surfaces sont identiques avec un certain risque α. Plus précisément, la probabilité que $\mathcal{S}_1$ et $\mathcal{S}_2$ soient identiques mais qu'on ne les considère pas comme identiques est α. Un petit α aura donc tendance à identifier plus facilement deux surfaces, tandis qu'un grand α sera plus exigeant pour les identifier.

7.5 Tests statistiques [1]

Soient $X_1, \ldots, X_m$ et $Y_1, \ldots, Y_n$ deux échantillons tels que les $X_i, i \in \{1, \ldots, m\}$ soient indépendants et identiquement distribués, de distribution $\mathcal{D}_X$ et de fonction de répartition F_X, et les $Y_j, j \in \{1, \ldots, n\}$ soient i.i.d. de distribution $\mathcal{D}_Y$ et de fonction de répartition F_Y.

$$F_X(t) = P(X < t),\ t \in]-\infty, +\infty[l \tag{7.1}$$

En pratique F_X et F_y sont inconnues. Les données auxquelles nous avons accès sont les $x_1, \ldots, x_m$ et les $y_1, \ldots, y_n$, qui sont les observations associées aux variables aléatoires $X_1, \ldots, X_m$ et $Y_1, \ldots, Y_n$.

Soient $\hat{F}_x$ et $\hat{F}_y$ les fonctions de répartition empiriques correspondant aux observations :

$$\hat{F}_X(t) = \frac{1}{m} \sum_{i=1}^{m} \mathbf{1}_{x_i<t},\ t \in \mathbb{R} \tag{7.2}$$

L'objet des tests présentés ici est de décider si F_X et F_Y sont les mêmes pour un risque α. On veut tester l'hypothèse $H_0 :\ F_X = F_Y$.

Deux tests classiques pour comparer deux distributions sont présentés par la suite : le test de Kolmogorov-Smirnov et le test de Mann-Whitney.

7.5.1 Test de Kolmogorov-Smirnov

Le test de Kolmogorov-Smirnov [Kolmogorov, 1933; Smirnov, 1948] (test KS) mesure la distance entre deux distributions. Il va principalement rejeter H_0 du fait des différences de formes entre les deux distributions.

Soit $D_{X,Y}$ la distance maximale entre les fonctions de répartition empiriques :

$$D_{X,Y} = \sup_{t \in \mathbb{R}} |\hat{F}_X(t) - \hat{F}_Y(t)| \tag{7.3}$$

En pratique pour calculer $D_{X,Y}$, on crée la séquence $z_1, \ldots, z_{m+n}$ en fusionnant et en triant dans l'ordre croissant les deux ensembles $x_1, \ldots, x_m$ et $y_1, \ldots, y_n$. Soit S_k une valeur associée à z_k, telle que :

$$S_k = \begin{cases} \frac{1}{m} & z_k \in \{x_1, \ldots, x_m\} \\ -\frac{1}{n} & z_k \in \{y_1, \ldots, y_n\} \end{cases} \tag{7.4}$$

Alors la distance $D_{X,Y}$ peut s'écrire :

[1] Les notions mathématiques utilisées dans cette section sont issues du livre de cours « Statistique et Analyse de Données » 2010-2011 de l'École des Ponts ParisTech

$$D_{X,Y} = \max_{1 \leq k \leq m+n} |\sum_{i=1}^{k} S_k| \tag{7.5}$$

L'hypothèse H_0 est rejetée lorsque $D_{X,Y} > a$ où a est indépendant de F_X et F_Y. Lorsque $m+n$ est petit (inférieur à 50) les valeurs de a sont tabulées pour différentes valeurs de α.

Pour les grandes valeurs de $m+n$, on rejette H_0 si :

$$D_{X,Y} > c(\alpha)\sqrt{\frac{m+n}{mn}} \tag{7.6}$$

La valeur de $c(\alpha)$ dans l'équation (7.6) est fixée à partir de tables de la distribution de Kolmogorov-Smirnov. Le tableau 7.1 montre quelques valeurs de $c(\alpha)$ pour différentes valeurs de α.

α	0.10	0.05	0.025	0.01	0.005	0.001
$c(\alpha)$	1.22	1.36	1.48	1.63	1.73	1.95

TABLE 7.1 – $c(\alpha)$ pour des valeurs classiques de α.

7.5.2 Test de Mann-Whitney

Le test de Mann-Whitney [Mann and Whitney, 1947; Wilcoxon, 1945] (test MW) mesure en particulier la distance entre les moyennes des distributions. Comparé au test précédent, il va plus particulièrement repérer les différences de positionnement dans l'espace.

Soit $U_{Y,X}$ la statistique de Mann-Withney, de loi $\mathcal{U}(n,m)$:

$$U_{Y,X} = \sum_{\substack{1 \leq i \leq m \\ 1 \leq j \leq n}} \mathbf{1}_{\{Y_j > X_i\}} \tag{7.7}$$

Soit $U_{n,m}$ de loi $\mathcal{U}(n,m)$, si $\min(m,n) \longrightarrow \infty$, alors $\zeta_{m,n}$ définie par :

$$\zeta_{m,n} = \frac{U_{m,n} - \frac{mn}{2}}{\sqrt{\frac{mn(m+n+1)}{12}}} \tag{7.8}$$

converge en loi vers la loi normale centrée réduite $\mathcal{N}(0,1)$.

Sous l'hypothèse, $H_1 = \{\mathbb{P}(Y_j > X_i) \neq \frac{1}{2}\}$, avec $m \to +\infty$, $n \to +\infty$ et $\frac{m}{n} \to l \in]0, +\infty$, la statistique tend presque sûrement vers $-\infty$ ou $+\infty$. La région de rejet est de la forme $]-\infty, -a[\cup]a, +\infty[$. Pour un test de rejet de la loi H_0 au niveau α, on utilise pour $a = z(\alpha/2)$ le quantile de la loi normale centrée réduite d'ordre $1 - \frac{\alpha}{2}$.

Comme dans le cas du KS-test, pour un calcul rapide $U_{Y,X}$, on fusionne et on ordonne les $x_1, \ldots, x_m$ et les $y_1, \ldots, y_n$, pour former une séquence $z = z_1, \ldots, z_{m+n}$.

Notant S_j le rang de y_j dans Z, c.-à-d. $S_j = \underset{k}{\text{argmin}}(z_k = y_j)$, on calcule la somme des rangs pour Y :

$$R_Y = \sum_{j=1}^{n} S_j \tag{7.9}$$

et :

$$U_{Y,X} = R_Y - \frac{n(n+1)}{2}. \tag{7.10}$$

Finalement, $U_{Y,X}$ est comparée à $z(\alpha/2)$, quantile de la loi normale centrée réduite d'ordre $1 - \frac{\alpha}{2}$. Si $U_{Y,X} > z(\alpha/2)$, le test rejette l'hypothèse H_0.

Les valeurs de $z(\alpha/2)$ sont accessibles via les tables de la distribution normale centrée réduite. Le tableau 7.2 montre quelques valeurs de $z(\alpha/2)$ pour différentes valeurs de α.

α	0.10	0.05	0.025	0.01	0.005
$z(\alpha/2)$	1.65	1.96	2.24	2.58	2.81

TABLE 7.2 – $z(\alpha/2)$ pour des valeurs classiques de α.

7.6 Critères pour les surfaces

Soit $d(p, \mathcal{S})$ la distance d'un point p à la surface $\mathcal{S}$. Deux cas sont envisagés ici.

7.6.1 Comparaison de surfaces

Étant donnés deux surfaces $\mathcal{S}_1$ et $\mathcal{S}_2$ et des ensembles de points correspondants P_1 et P_2, nous cherchons à utiliser les résultats de la section 7.5 pour comparer $\mathcal{S}_1$ et $\mathcal{S}_2$. Pour cela, nous cherchons X et Y tels que si $\mathcal{S}_1 = \mathcal{S}_2$ alors les tests définis précédemment retiennent l'hypothèse H_0.

Une première formulation pourrait être :

$$X = \{d(p_1, \mathcal{S}_1),\ p_1 \in P_1\} \tag{7.11}$$

$$Y = \{d(p_2, \mathcal{S}_2),\ p_2 \in P_2\} \tag{7.12}$$

Cependant rien ne garantit que la distribution des points de P_1 autour de $\mathcal{S}_1$ et celle des points de P_2 autour de $\mathcal{S}_2$ soient identiques. Si elles sont différentes, les tests précédents peuvent rejeter H_0 du fait des formes des distributions, même dans le cas où $\mathcal{S}_1 = \mathcal{S}_2$. La figure 7.2 illustre ce cas.

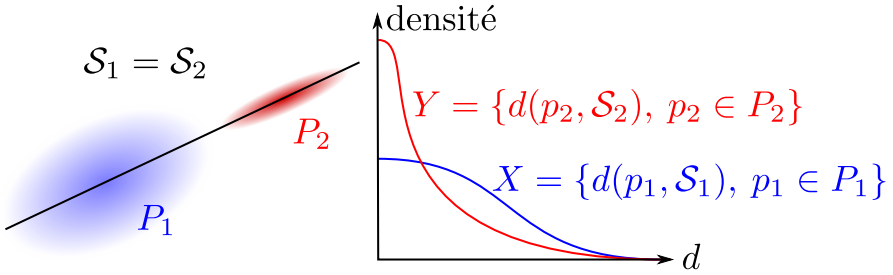


FIGURE 7.2 – Cas où $\mathcal{S}_1 = \mathcal{S}_2$, X et Y sont définis par les équations (7.11) et (7.12). Les tests rejettent l'hypothèse H_0.

Pour éviter ce problème, il faut symétriser les ensembles X et Y :

$$X = \{d(p_1, \mathcal{S}_1),\ p_1 \in P_1\} \cup \{d(p_2, \mathcal{S}_2),\ p_2 \in P_2\} \tag{7.13}$$

$$Y = \{d(p_1, \mathcal{S}_2),\ p_1 \in P_1\} \cup \{d(p_2, \mathcal{S}_1),\ p_2 \in P_2\} \tag{7.14}$$

Ici, si $\mathcal{S}_1 = \mathcal{S}_2$ alors $X = Y$ et les tests retiennent H_0.

Il est important de noter que pour créer les ensembles X et Y seules les fonctions distances relatives à chaque surface sont nécessaires. Il n'y a donc pas de restriction sur le type de surface utilisable (plan, cylindre,...) pourvu que la distance soit calculable. La

figure 7.3 montre l'allure des distributions X et Y définies par l'équation (7.14) pour trois paires de surfaces $(\mathcal{S}_1, \mathcal{S}_2)$. Les surfaces sont des lignes parallèles plus ou moins éloignées.

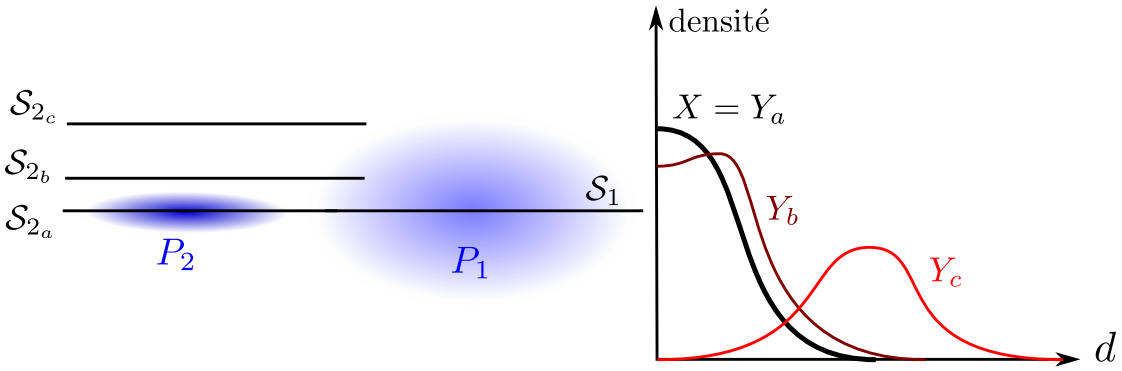


FIGURE 7.3 – Allures des distributions X et Y définis par l'équation (7.14) pour trois paires de surfaces $(\mathcal{S}_1, \mathcal{S}_2)$.

7.6.2 Surface de fusion

Les tests définis précédemment valident ou non l'hypothèse d'identité des surfaces $\mathcal{S}_1$ et $\mathcal{S}_2$. Cependant, il est possible de vouloir tester l'adéquation non pas de $\mathcal{S}_1$ à $\mathcal{S}_2$ mais de $\mathcal{S}_1$ et $\mathcal{S}_2$ à une troisième surface $\mathcal{S}_3$. Ce serait par exemple le cas lors de la fusion de deux plans en un troisième, plan de régression pour le nuage de points $P_1 \cup P_2$. On se pose la question de savoir si cette troisième surface $\mathcal{S}_3$ est un bon consensus pour $P_1 \cup P_2$. Étant donnée une fonction distance pour $\mathcal{S}_3$, on définit l'ensemble Y par :

$$Y = \{d(p_1, \mathcal{S}_3),\ p_1 \in P1\} \cup \{d(p_2, \mathcal{S}_3),\ p_2 \in P2\} \tag{7.15}$$

L'ensemble X reste celui de la section précédente.

Exemple Soient deux droites $\mathcal{S}_1$ et $\mathcal{S}_2$ pour les ensembles de points P_1 et P_2. Il est aisé de calculer la droite de régression S_3 pour $P_1 \cup P_2$. Les tests doivent alors décider si la distribution des points autour de $\mathcal{S}_1$ et $\mathcal{S}_2$ est proche de celle des mêmes points autour de $\mathcal{S}_3$. La figure 7.4 illustre ce procédé. Elle présente les droites $\mathcal{S}_1$, $\mathcal{S}_2$ et $\mathcal{S}_3$ et les points de P_1 et P_2, ainsi que les profils des distributions pour X et Y (pour les équations (7.14) et (7.15)).

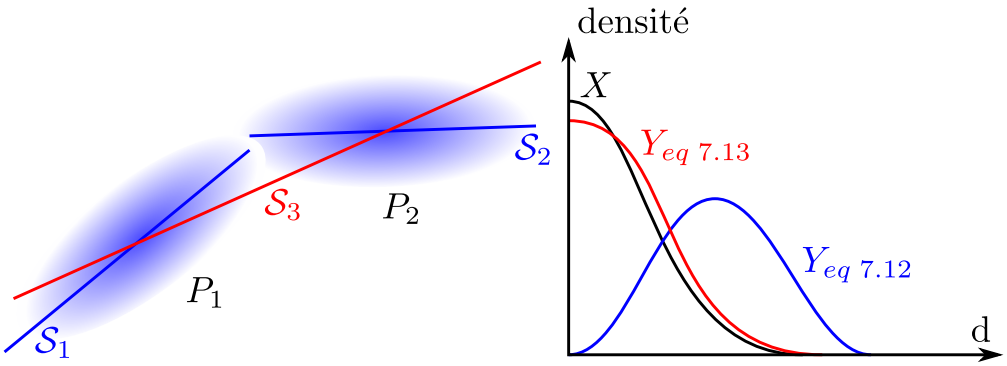


FIGURE 7.4 – Exemple d'utilisation de la droite de régression pour $P_1 \cup P_2$.

7.7 Taille des échantillons

La taille des échantillons, notée n, influence le temps de calcul et la qualité des tests de Kolmogorov-Smirnov et de Mann-Withney

D'un côté les calculs des statistiques des tests KS et MW se basent sur le calcul de la distance pour chaque point de l'échantillon et le tri de ces distances. Ainsi une taille des échantillons accrue augmente le temps de calcul.

D'un autre côté, n définit aussi implicitement la qualité d'estimation des distributions. Plus le nombre de points est important, mieux les distributions de X et Y sont estimées. Une faible différence entre les distributions pourra conduire au rejet de H_0 d'autant plus sûrement que n est grand.

Ainsi, la taille de X et Y apparaît un paramètre possible pour contrôler l'acceptation des tests. Cependant le réglage de ce paramètre dans cette optique est très difficile. Il n'est pas directement relié à une grandeur physique comme la distance entre deux plans parallèles. Nous choisissons de fixer n de sorte à avoir des garanties sur la qualité des estimations des distributions. Le réglage de l'acceptation des tests sera développé dans la section 7.8.

Borne inférieure pour la taille des échantillons

La méthode présentée ici est similaire à celle du chapitre 4, section 4.4 mais dans un espace continu.

Soient $X_1, \ldots, X_n$, n variables aléatoires indépendantes et identiquement distribuées. On note F la fonction de répartition de leur distribution et $\hat{F}_n$ leur fonction de répartition empirique.

Soit $\beta \in]0, 1[$, une probabilité qui représente le niveau de confiance pour l'estimation des distributions. Soit $\epsilon > 0$ un seuil de tolérance pour la distance entre deux distributions. L'objectif est d'estimer le cardinal n minimal de X tel que F et $\hat{F}_n$ soient à distance inférieure à ϵ, avec probabilité β :

$$\mathbb{P}(\sup_{t\in\mathbb{R}}|\hat{F}_n(t) - F(t)| \leq \epsilon) \geq \beta \tag{7.16}$$

Théorème 3.

$$n \geq \frac{1}{2\epsilon^2}\ln(\frac{2}{1-\beta}) \tag{7.17}$$

est une condition suffisante pour que l'équation 7.16 soit vraie.

Démonstration. Le pendant de l'inégalité de Hoeffding [Hoeffding, 1963] (cas discret, chapitre 4) pour le domaine du continu est l'inégalité de Dvoretzky-Kiefer-Wolfowitz (DKW) [Dvoretzky et al., 1956] :

$$\mathbb{P}(\sup_{t\in\mathbb{R}}|\hat{F}_n(t) - F(t)| > \epsilon) \leq 2e^{-2n\epsilon^2} \tag{7.18}$$

Ainsi :

$$\mathbb{P}(\sup_{t\in\mathbb{R}}|\hat{F}_n(t) - F(t)| \leq \epsilon) \geq 1 - 2e^{-2n\epsilon^2} \tag{7.19}$$

En y insérant l'équation (7.16), on obtient pour n la borne inférieure suivante :

$$n \geq \frac{1}{2\epsilon^2}\ln(\frac{2}{1-\beta}) \tag{7.20}$$

□

En pratique, étant donnés ϵ et β, nous choisirons le plus petit n possible vérifiant l'équation (7.17), c'est-à-dire :

$$n = \left\lceil \frac{1}{2\epsilon^2} \ln(\frac{2}{1-\beta}) \right\rceil \tag{7.21}$$

Dans la suite, nous utiliserons β tel $\beta = 1 - \alpha$, où α est le risque de la section 7.5. Le tableau 7.3 donne les valeurs de n pour quelques couples (α, ϵ).

ϵ \ α	0.001	0.005	0.01	0.05	0.1
0.01	38005	29958	26492	18445	14979
0.05	1521	1199	1060	738	600
0.1	381	300	265	185	150
0.2	96	75	67	47	38
0.5	16	12	11	8	6

TABLE 7.3 – Valeurs de n pour différentes valeur de (α, ϵ).

7.8 Donner le contrôle à l'utilisateur

Les critères présentés précédemment admettent deux paramètres, le niveau de risque α et le seuil ϵ sur l'estimation des distributions. Une fois ces deux paramètres fixés, l'utilisateur n'a plus aucun contrôle sur l'acceptation des tests. Une petite différence dans les distributions peut mener à un rejet de H_0.

Par exemple, il est commun de vouloir fusionner des plans parallèles si ceux-ci sont à une distance inférieure à un seuil. Or si les points sont échantillonnés avec du bruit sur la surface, une distance infime entre les plans pourra conduire à un rejet.

L'idée, ici, est d'introduire un paramètre, homogène à une distance, définissant une sorte de distance d'acceptation. Afin de forcer l'acceptation par les tests, les distributions doivent être d'allure similaire. Pour ce faire, un bruit gaussien d'écart type σ est ajouté aux points. Ceci a pour effet d'aplatir les distributions, rendant deux distributions plus similaires. Ainsi plus le niveau de bruit est important, plus deux segments auront tendance à fusionner. La figure 7.5 illustre ce phénomène.

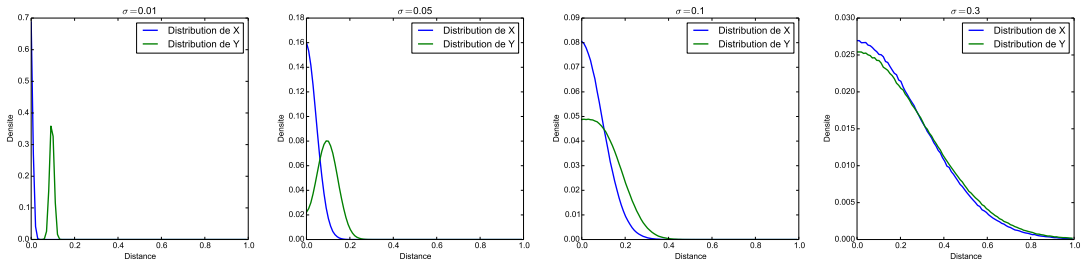


FIGURE 7.5 – Effet de l'ajout de bruit sur les distributions X et Y. Expérience effectuée avec deux droites parallèles à distance 0.1 (unité arbritraire), et différentes valeurs de σ.

σ est déterminé empiriquement pour le KS-test et pour des nuages de points échantillonnés sur deux surfaces. Cette évaluation est faite sur quatre configurations différentes :

- Deux plans parallèles à une distance h.
- Deux cylindres parallèles de rayon identique r dont les axes sont à distance $h \ll r$.
- Deux sphères de rayon identique r dont les centres sont à distance $h \ll r$.
- Deux plans sécants avec un angle $\theta \ll \pi/2$, la distance h est la distance moyenne des points au plan auquel ils ne sont pas associés.

Ici $a \ll b$ signifie $10a < b$. La situation est identique à un facteur d'échelle près. C'est pourquoi il suffit d'expérimenter pour les cylindres et les sphères avec $r = 1$.

La figure 7.6 présente les résultats de simulations pour les quatre configurations suscitées. Pour chacune d'entre elles, nous représentons par un niveau de couleur le taux d'acceptation en fonction de la distance spécifique étudiée h et et du niveau de bruit σ ajouté aux données. Comme attendu plus le niveau de bruit est important, plus le taux d'acceptation est lui-aussi élevé.

Sur chaque graphique, la droite noire représente la droite de régression pour le niveau à 50% de test positif. On note son coefficient directeur $\lambda_{0.5}$. Les droites blanches sont respectivement les droites au niveau d'acceptation à 95% et 5% dont les coefficients directeurs sont notés respectivement $\lambda_{0.95}$ et $\lambda_{0.05}$.

On observe que le niveau de bruit à ajouter pour un taux d'acceptation donné croît linéairement en fonction de la distance étudiée, quelle que soit la configuration.

Soit δ, la distance moyenne entre les points d'une surface et l'autre surface, nous avons une relation du type :

$$\sigma = \lambda\ \delta \tag{7.22}$$

où λ est un facteur multiplicateur qui dépend peu de la configuration choisie.

Pour les expériences de la section 7.10, nous ajoutons un bruit gaussien d'écart type σ défini par un paramètre de distance δ selon l'équation (7.22). Le λ utilisé est celui des plans parallèles. Ce choix est conservatif pour les surfaces non planaires. Du point de vue de l'utilisateur, δ est la distance maximale à laquelle deux plans parallèles vont être fusionnés.

7.9 Algorithme de fusion de primitives

La fusion des segments est effectuée de manière gloutonne. Premièrement, on calcule la statistique pour chaque paire de primitives. Les deux primitives ayant la statistique la plus faible fusionnent puis on recalcule les statistiques pour toutes les paires impliquant la surface nouvellement créée. On répète l'opération jusqu'à ce qu'aucune paire ne valide le test. L'algorithme 3 résume l'opération.

Pour des raisons de commodité, nous définissons deux fonctions : KS_{stat} et MW_{stat}. Soient $(\mathcal{S}_1, P_1)$ et $(\mathcal{S}_2, P_2)$ les couples surface-points à comparer, alors :

$$KS_{stat}((\mathcal{S}_1, P_1), (\mathcal{S}_2, P_2)) = D_{X,Y} * \sqrt{\frac{n}{2}} \tag{7.23}$$

et

$$MW_{stat}((\mathcal{S}_1, P_1), (\mathcal{S}_2, P_2)) = U_{X,Y} \tag{7.24}$$

où X et Y sont les ensembles définis par les équations (7.13) et (7.14) ou (7.15).

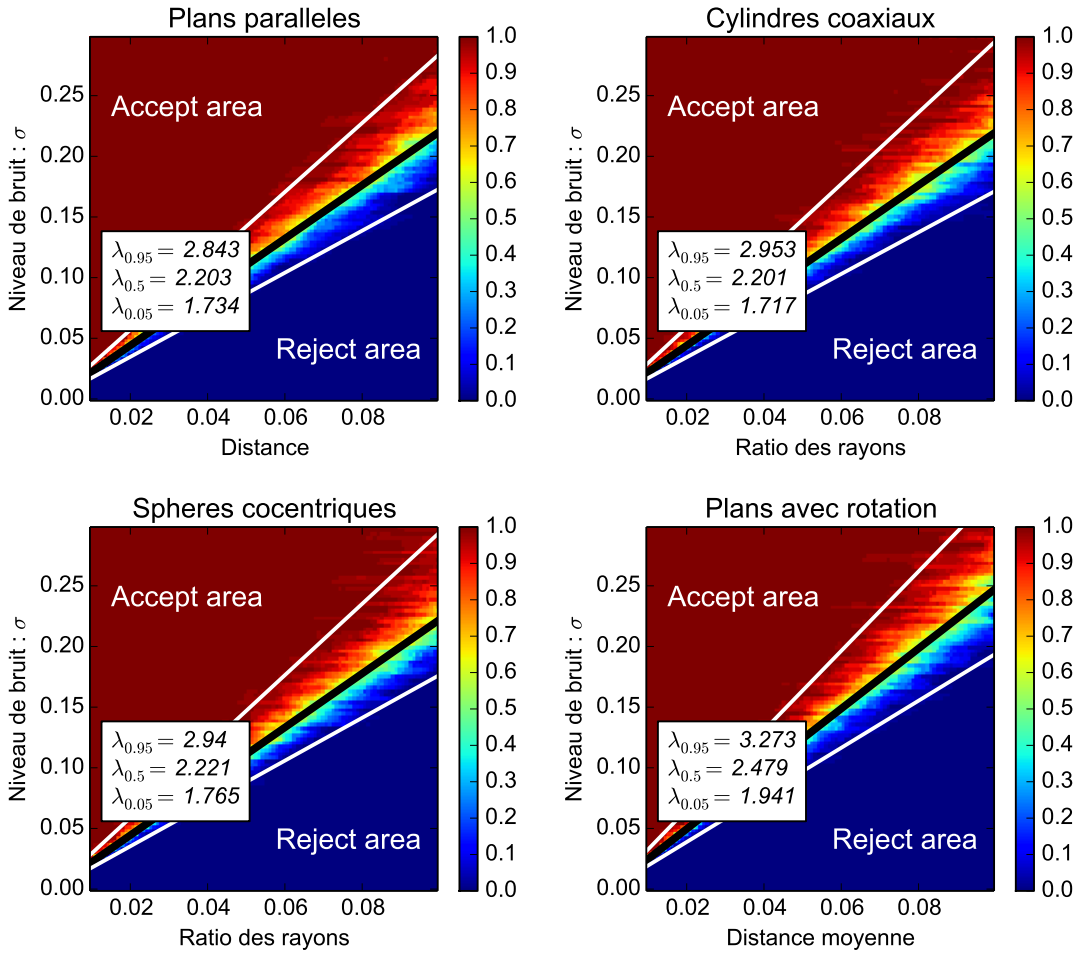


FIGURE 7.6 – Taux d'acceptation pour quatre configurations, KS-test et pour $\alpha = 0.01$ et $\epsilon = 0.05$.

7.10 Expériences

Cette section présente quelques expériences pour évaluer le comportement des critères de fusion developpés précédemment. Pour toutes les expériences, les valeurs $\alpha = 0.01$ et $\epsilon = 0.1$ sont fixes, ce qui correspond à un nombre de tirages $n_{min} = 265$ (équation (7.17)).

7.10.1 Comparaison empirique des tests

Cette section compare les tests KS et MW. La figure 7.7 présente la réponse des deux tests sur une scène synthétique composée de deux sphères concentriques, l'une unitaire, l'autre de rayon variable. Les courbes donnent la valeur de la statistique KS ou MW en fonction du rayon, pour $\alpha = 0.01$ et $\epsilon = 0.1$. La limite d'acceptation est donnée par une droite horizontale, correspondant à $c(\alpha)$ pour le test KS et $z(\alpha/2)$ pour le test MW. Chaque point est une moyenne sur 100 tests empiriques. Pour les valeurs de δ 0.01, 0.03 et 0.05, les deux tests sont en accord. La statistique de Kolmogorov-Smirnov semble néanmoins légèrement plus stable.

Les expériences suivantes utilisent le KS-test uniquement.

Algorithme 3 Algorithme de fusion de primitives.

Paramètre: n_{min} # Nombre de points pour l'estimation de la statistique
Paramètre: δ # Seuil d'acceptation
Paramètre: $Prims$ # Ensemble de primitives + points issus de la segmentation.
Paramètre: $c(\alpha)$ # Seuil d'acceptation du KS-test

$T \leftarrow \emptyset$
pour tout $q \in Prims \times Prims$ **faire**
  $T[q] \leftarrow KS_{stat}(q)$ # Mémorisation de la statistique
fin pour

tant que $\exists q \in T,\ T[q] < c(\alpha)$ **faire**
  $q_{min} = ((\mathcal{S}_1, P_1), (\mathcal{S}_2, P_2)) \leftarrow \underset{q}{\text{argmin}}\ T[q]$
  $P_{q_{min}} = P_1 \cup P_2$ # Union des nuages de la paire q_{min}
  $\mathcal{S}_{q_{min}} \leftarrow$ Surface de régression pour $P_{q_{min}}$

  # Mise à jour de $Prims$
  $Prims \leftarrow Prims \setminus \{(\mathcal{S}_1, P_1), (\mathcal{S}_2, P_2)\}$
  $Prims \leftarrow Prims \cup \{(\mathcal{S}_{q_{min}}, P_{q_{min}})\}$

  # Mise à jour de T avec les primitives fusionnées
  Suppression des entrées contenant $(\mathcal{S}_1, P_1)$ ou $(\mathcal{S}_2, P_2)$ dans T
  pour tout $s \in Prims$ **faire**
    $q \leftarrow (s, (\mathcal{S}_{q_{min}}, P_{q_{min}}))$
    $T[q] \leftarrow KS_{stat}(q)$
  fin pour
fin tant que

7.10.2 Fusion de segments planaires pour un nuage laser

L'utilisation pratique des critères de fusion sur une scène complète est décrite dans cette partie. L'étape d'extraction de primitives à partir du nuage de points est décrite dans le chapitre 6. On considère avoir en entrée un ensemble de primitives géométriques associées à des ensembles de points. Sur les exemples de scènes laser, c'est la croissance de régions qui est utilisée.

Évolution de la fusion pour différents δ

Le résultat d'une fusion progressive des primitives dans un nuage laser est donné figure 7.8. Le résultat est présenté pour trois δ différents. Un δ plus élevé génère plus de fusion, mais même pour un grand δ les fusions effectuées sont pertinentes.

On observe aussi que pour $\delta = 0$, aucune des primitives ne fusionne. Ceci est dû à la très grande précision du laser (erreur de l'ordre du millimètre). A cette échelle la moindre variation de l'orientation ou de la position relative des plans est rejetée par le test.

Fusion sur plusieurs acquisitions

Il est aussi possible d'utiliser le critère sur plusieurs acquisitions. En effet les nuages de points laser étant généralement lourds (plusieurs millions de points), il est intéressant de traiter les points de chaque acquisition séparément, puis seulement à la fin, de les

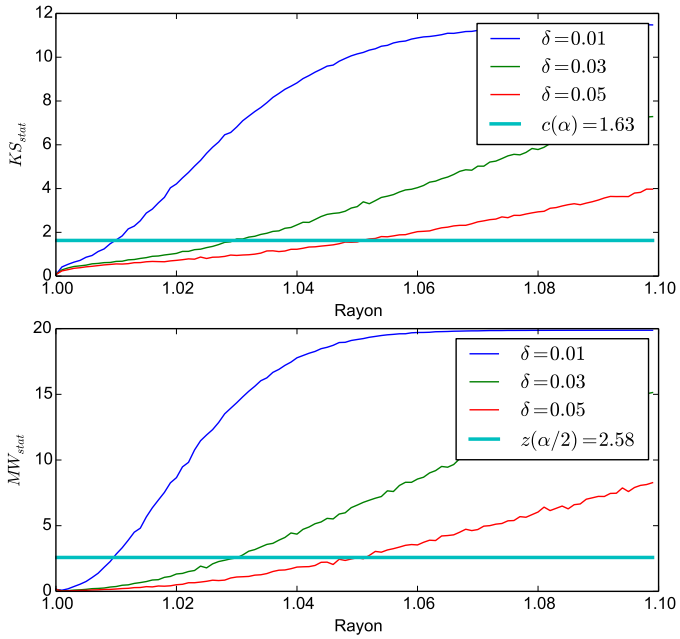
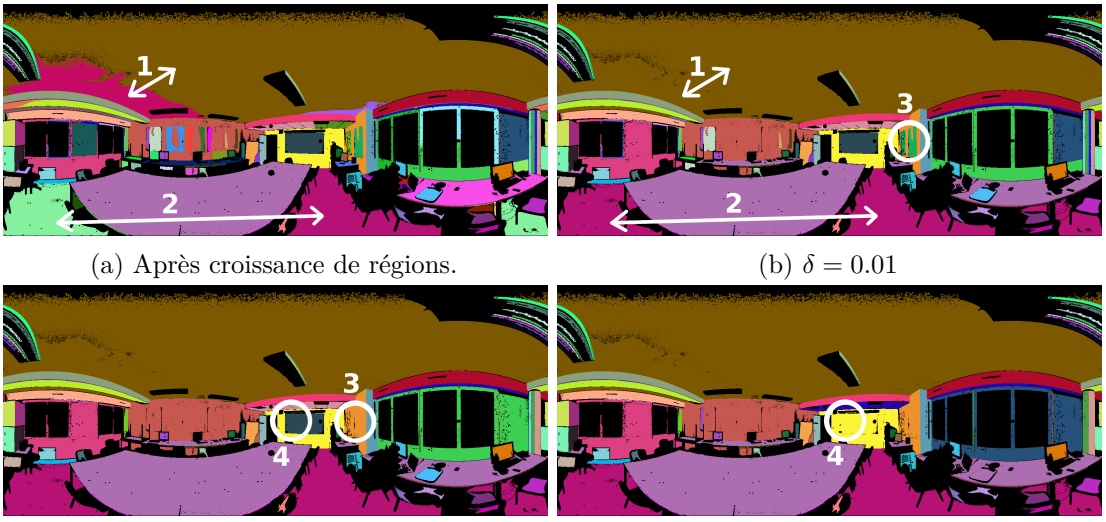


FIGURE 7.7 – Comparaison du test KS (en haut) et du test MW (en bas) sur deux sphères concentriques. L'une est la sphère unitaire, l'autre a un rayon variable. α et ϵ sont fixes et valent respectivement 0.01 et 0.1.

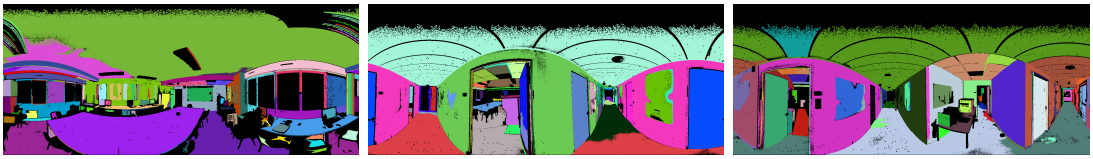


(a) Après croissance de régions. (b) $\delta = 0.01$

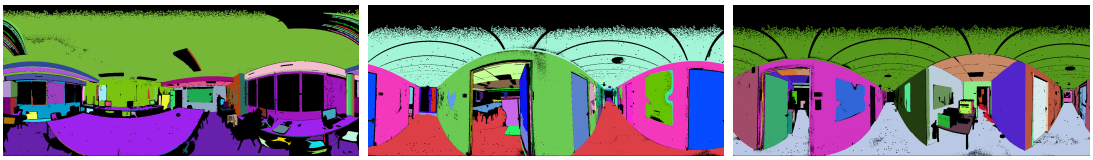
(c) $\delta = 0.02$ (d) $\delta = 0.03$

FIGURE 7.8 – Évolution de la fusion pour un delta croissant.

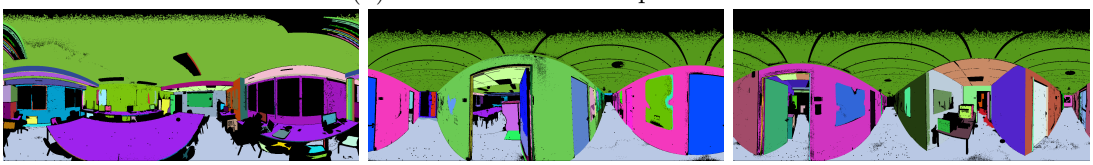
regrouper. La figure 7.9 présente le processus de fusion pour trois acquisitions laser, celles-ci ont été préalablement placées dans le même système de coordonnées. La croissance de régions effectuée dans le nuage de chaque acquisition (7.9a) puis les segments représentant les mêmes primitives sont fusionnés au sein du nuage avec l'algorithme 3 (7.9b). Enfin, les trois paires (nuage de points, primitives) sont regroupées et l'algorithme 3 est appliqué à nouveau (7.9c).



(a) Résultat de la croissance de région.



(b) Fusion interne à l'acquisition.



(c) Fusion à l'échelle des trois acquisitions.

FIGURE 7.9 – Processus de fusion de segments pour plusieurs acquisitions laser.

7.10.3 Comparer des torchons et des serviettes

Une des grandes forces de la méthode de fusion présentée ici est de ne nécessiter qu'une connaissance limitée des surfaces à comparer, à savoir être capable de calculer la distance d'un point à la surface. On peut donc appliquer le critère à des surfaces simples (plans, sphères...) mais aussi à n'importe quelle surface.

La formulation des ensembles X et Y (équation (7.14) et (7.15)) n'impose pas la similarité des surfaces $\mathcal{S}_1$, $\mathcal{S}_2$ et $\mathcal{S}_3$. Il est donc possible de comparer des surfaces de différentes natures.

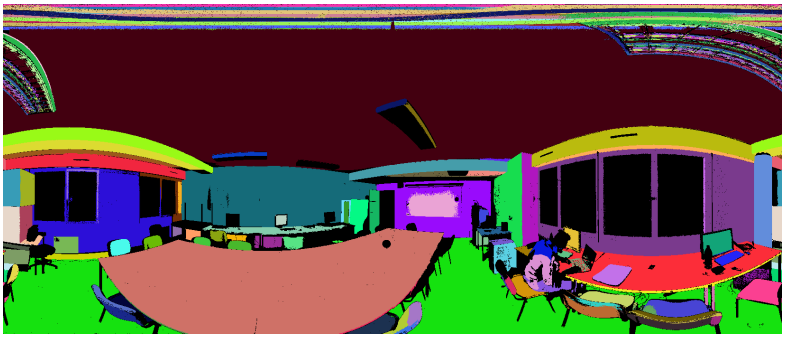
Pour illustrer ce type de comparaison, la figure 7.10 montre une scène laser sur laquelle a été appliqué un RANSAC [Schnabel et al., 2007] pour la recherche de plans et de cylindres. Le RANSAC est un algorithme glouton. Une fois la primitive détectée, les points associés à celle-ci sont écartés pour ne pas être réutilisés. Comme la primitive n'est jamais remise en cause, il est possible d'avoir détecté un plan comme un fragment de cylindre.

Comme il est généralement plus simple de manipuler des plans plutôt que des cylindres, nous souhaitons remplacer des cylindres par des plans. Nous nous posons ici la question de savoir si le plan de régression pour chaque segment associé à un cylindre est une bonne primitive, auquel cas il remplacera le cylindre.

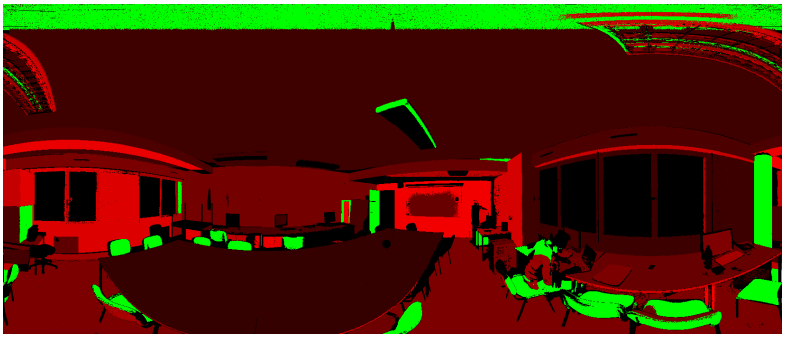
La figure 7.10a montre le résultat de la segmentation (une couleur par primitive) et la figure 7.10b leur type (vert pour un cylindre, rouge pour un plan).

On observe que certains plans naturels, comme une porte ou une partie du plafond ont été détectés comme des cylindres.

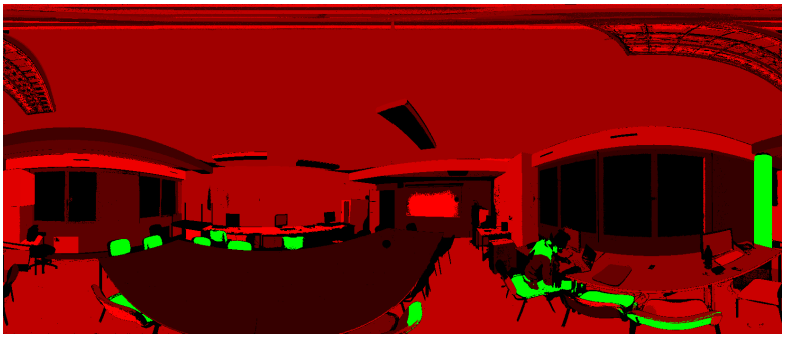
Pour chacun des segments associés à un cylindre, le plan de régression est calculé et est comparé au cylindre pour $\delta = 0.01$ (c.-à-d. 1cm). Si le test est positif, le segment n'est plus considéré comme un cylindre mais comme un plan, défini comme son plan de régression. La figure 7.10c présente les types de primitives après les tests. On peut observer que les points associés à tort à un cylindre sont désormais des plans. Les seuls éléments cylindriques restants sont une colonne, ainsi que les dossiers et assises de chaises.



(a) Segmentation par RANSAC.



(b) Types de primitives.



(c) Types après réassignation.

FIGURE 7.10 – Comparaison de plans et de cylindres.

7.10.4 Données photogrammétriques

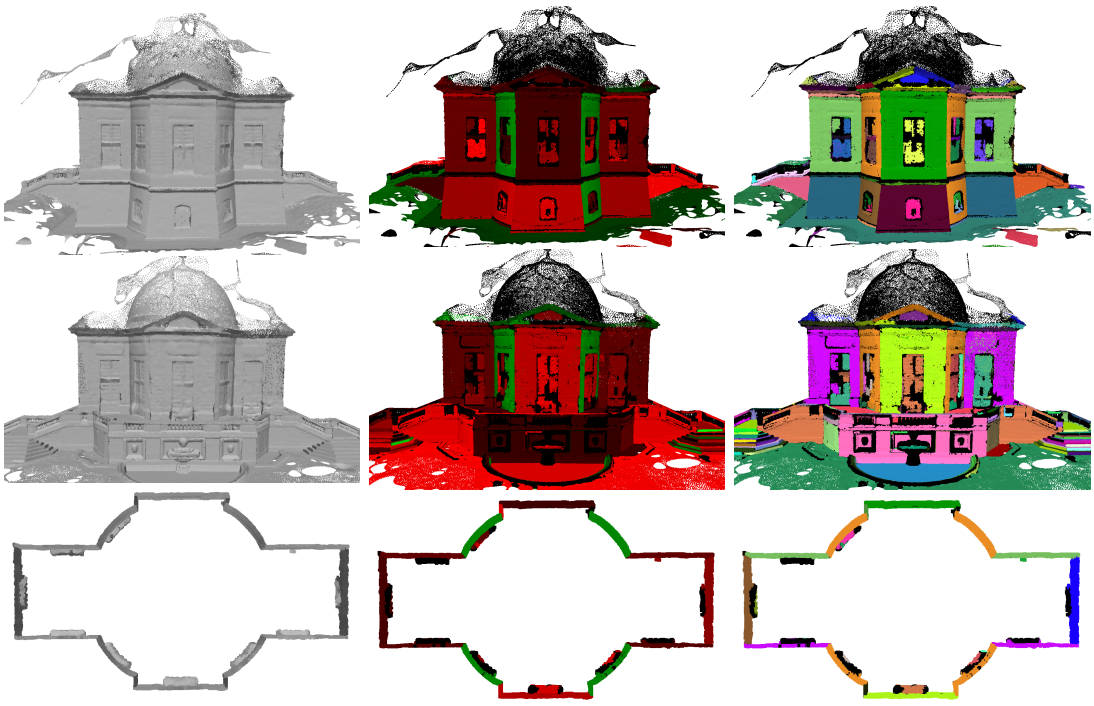
Les données photogrammétriques sont plus bruitées et localement moins régulièrement échantillonnées sur la surface. On observe aussi des points aberrants dus à des erreurs de mise en correspondance, notamment dû au fait de reflets ou de manque de texture. Pour les expériences, l'entrée est un nuage de points reconstruit à partir de multiples photos, les poses des caméras sont estimées en utilisant OpenMVG [site] [Moulon et al., 2013] et les points obtenus via la chaîne ImagineMVS [Vu et al., 2009].

La croissance de régions est légèrement différente de celle utilisée pour les images laser, où l'extension des primitives exploitait la 8-connectivité des pixels de l'image de profondeur. Ici, le voisinage d'un point est calculé à l'aide d'un KD-tree. Un voisin est ajouté au segment si la déviation de sa normale avec le point est faible. Il en résulte des segments potentiellement courbes.

Pour chaque segment, sont calculés le plan de régression et un cylindre à la manière de RANSAC (en tirant plusieurs couples de points et en gardant le meilleur cylindre). Ensuite, nous appliquons une méthode proche de celle de la section 7.10.3. Nous choisissons la

surface qui correspond le mieux aux données (au sens de la distance moyenne des points à la surface). Si celle-ci est le cylindre, nous testons l'adéquation avec le plan de régression grâce au test KS et le remplaçons par le plan si le test est positif. Enfin l'algorithme 3 est appliqué pour fusionner les primitives identiques.

La figure 7.11 présente les résultats de cette opération sur le *Pavillon de l'Aurore*, à Sceaux. Le nuage de points contient 1.6 millions de points. On observe notamment que le cylindre central est correctement identifié et fusionné.



(a) Nuage de point (b) Type de primitives. (c) Primitives après fusion

FIGURE 7.11 – Fusion sur des données photogrammétriques.

7.11 Conclusion

Ce chapitre présente deux variantes d'une méthode générale de décision pour la fusion de primitives. Basée sur des tests statistiques, elle repose sur un nombre limité de paramètres et sur une connaissance minimale des surfaces à fusionner ou à comparer. L'utilisation d'une inégalité statistique permet de limiter au maximum le nombre de points à traiter tout en rendant les tests insensibles à la taille des nuages, limitant ainsi le temps de calcul. Afin de rendre l'utilisation plus facilement paramétrable, un paramètre δ, homogène à une distance et aisément réglable, a été introduit.

Les expériences illustrent les possibilités de l'approche au travers de la fusion des primitives ou de leur comparaison. L'algorithme 3 est proposé en illustration, c'est un algorithme glouton dont les performances peuvent-être largement améliorées.

Remerciements

Ici encore, merci à Pierre Moulon pour les modèles photogrammétriques et à Arnak Dalalyan pour son aide en statistiques.

Publication

Ce travail a été présenté à International Conference on Pattern Recognition 2014, à Stockholm, Suède.

Alexandre Boulch, Renaud Marlet.
Statistical criteria for primitive merging
ICPR 2014

Bibliographie

Aspert, N., Santa-Cruz, D., and Ebrahimi, T. (2002). Mesh: measuring errors between surfaces using the Hausdorff distance. In *IEEE Int'l Conf. on Multimedia and Expo (ICME 2002)*, volume 1, pages 705–708.

Besl, P. J. and Jain, R. C. (1988). Segmentation through variable-order surface fitting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 10(2):167–192.

Borrmann, D., Elseberg, J., Lingemann, K., and Nüchter, A. (2011). The 3D Hough Transform for plane detection in point clouds: A review and a new accumulator design. *3D Research*, 2(2):1–13.

Bughin, E. (2011). *Towards automated, precise and validated vectorisation of disparity maps in urban satellites stereoscopy*. PhD thesis, ENS Cachan.

Bughin, E., Almansa, A., von Gioi, R. G., and Tendero, Y. (2010). Fast plane detection in disparity maps. In *ICIP*, pages 2961–2964.

Chauve, A.-L., Labatut, P., and Pons, J.-P. (2010). Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *CVPR*, pages 1261–1268.

Digne, J., Morel, J.-M., Souzani, C.-M., and Lartigue, C. (2010). Mesh segmentation and model extraction. In *Curves and Surfaces*, pages 236–252.

Dvoretzky, A., Kiefer, J., and Wolfowitz, J. (1956). Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *The Annals of Mathematical Statistics*, pages 642–669.

Fiorio, C. and Nock, R. (2000). A concentration-based adaptive approach to region merging of optimal time and space complexities. In *BMVC*, pages 1–10.

Fischler, M. A. and Bolles, R. C. (1981). Random Sample Consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395.

Fitzgibbon, A. W., Eggert, D. W., and Fisher, R. B. (1997). High-level model acquisition from range images. *Computer-Aided Design*, 29(4):321–330.

Forlani, G., Nardinocchi, C., Scaioni, M., and Zingaretti, P. (2003). Building reconstruction and visualization from lidar data. *International Archives Of Photogrammetry Remote Sensing And Spatial Information Sciences*, 34(5/W12):151–156.

Fouhey, D. F., Scharstein, D., and Briggs, A. J. (2010). Multiple plane detection in image pairs using J-Linkage. In *20th International Conference on Pattern Recognition, ICPR 2010, Istanbul, Turkey, 23-26 August 2010*, pages 336–339.

Han, J., Volz, R. A., and Mudge, T. N. (1987). Range image segmentation and surface parameter extraction for 3-D object recognition of industrial parts. *Robotics and Automation. Proceedings.*, 4:380–386.

Hoeffding, W. (1963). Probability inequalities for sums of bounded random variables. *Journal of the American Statistical Association*, 58(301):13–30.

Hojjatoleslami, S. A. and Kittler, J. (1998). Region growing: a new approach. *IEEE Transactions on Image Processing*, 7(7):1079–1084.

Hough, P. V. C. (1962). Method and means for recognizing complex patterns. *U.S. Patent*, 3.069.654.

Kanazawa, Y. and Kawakami, H. (2004). Detection of planar regions with uncalibrated stereo using distributions of feature points. In *BMVC*, pages 1–10.

Knopp, J., Prasad, M., Willems, G., Timofte, R., and Gool, L. J. V. (2010). Hough transform and 3D SURF for robust three dimensional classification. In *ECCV (6)*, pages 589–602.

Kolmogorov, A. N. (1933). Sulla determinazione empirica di una legge di distribuzione. *Giornale dell'Istituto Italiano degli Attuari*, 4(1):83–91.

Labatut, P., Pons, J. P., and Keriven, R. (2009). Hierarchical shape-based surface reconstruction for dense multi-view stereo. *IEEE International Workshop on 3-D Digital Imaging and Modeling (ICCV-3DIM)*, page 1598–1605.

Lafarge, F. and Alliez, P. (2013). Surface reconstruction through point set structuring. In *Proc. of Eurographics*, Girona, Spain.

Li, Y., Wu, X., Chrysathou, Y., Sharf, A., Cohen-Or, D., and Mitra, N. J. (2011). GlobFit: consistently fitting primitives by discovering global relations. In *ACM Trans. on Graphics (TOG)*, volume 30-4, pages 52:1–52:12.

Mann, H. B. and Whitney, D. R. (1947). On a test of whether one of two random variables is stochastically larger than the other. *The Annals of Mathematical Statistics*, 18(1):50–60.

Moisan, L. and Stival, B. (2004). A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. *International Journal of Computer Vision*, 57(3):201–218.

Moulon, P., Monasse, P., and Marlet, R. (2012). Adaptive structure from motion with *a contrario* model estimation. In *ACCV (4)*, pages 257–270.

Moulon, P., Monasse, P., and Marlet, R. (2013). Global fusion of relative motions for robust, accurate and scalable structure from motion. In *IEEE International Conference on Computer Vision, ICCV 2013, Sydney, Australia, December 1-8, 2013*, pages 3248–3255.

Nock, R. and Nielsen, F. (2005). Semi-supervised statistical region refinement for color image segmentation. *Pattern Recognition*, 38(6):835–846.

OpenMVG (s.i.t.e.). OpenMVG (open Multiple View Geometry). https://github.com/openMVG/openMVG.

Overby, J., Bodum, L., Kjems, E., and Iisoe, P. M. (2004). Automatic 3D building reconstruction from airborne laser scanning and cadastral data using hough transform. *International Archives of Photogrammetry and Remote Sensing*, 35(B3):296–301.

Pham, M.-T., Woodford, O. J., Perbet, F., Maki, A., Stenger, B., and Cipolla, R. (2011). A new distance for scale-invariant 3D shape recognition and registration. In *ICCV*, pages 145–152.

Poppinga, J., Vaskevicius, N., Birk, A., and Pathak, K. (2008). Fast plane detection and polygonalization in noisy 3D range images. In *IROS*, pages 3378–3383.

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

Pu, S. and Vosselman, G. (2006). Automatic extraction of building features from terrestrial laser scanning. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5):25–27.

Rabin, J., Delon, J., Gousseau, Y., and Moisan, L. (2010). MAC-RANSAC: a robust algorithm for the recognition of multiple objects. In *Proceedings of 3DPTV*.

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226.

Shih, F. Y. and Cheng, S. (2005). Automatic seeded region growing for color image segmentation. *Image Vision Comput.*, 23(10):877–886.

Smirnov, N. (1948). Table for estimating the goodness of fit of empirical distributions. *The Annals of Mathematical Statistics*, 19(2):279–281.

Toldo, R. and Fusiello, A. (2008). Robust multiple structures estimation with J-Linkage. In *ECCV (1)*, pages 537–547.

Trémeau, A. and Borel, N. (1997). A region growing and merging algorithm to color segmentation. *Pattern Recognition*, 30(7):1191–1203.

Vincent, E. and Laganière, R. (2001). Detecting planar homographies in an image pair. *IEEE Symp. Image and Signal Processing and Analysis*, pages 182–187.

Vosselman, G. and Dijkman, S. (2001). 3D building model reconstruction from point clouds and ground plans. *International Archives of Photogrammetry Remote Sensing and Spatial Information Sciences*, 34(3/W4):37–44.

Vosselman, G., Gorte, B. G., Sithole, G., and Rabbani, T. (2004). Recognising structure in laser scanner point clouds. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 46(8):33–38.

Vu, H., Keriven, R., Labatut, P., and Pons, J. (2009). Towards high-resolution large-scale multi-view stereo. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 1430–1437.

Wilcoxon, F. (1945). Individual comparisons by ranking methods. *Biometrics bulletin*, 1(6):80–83.

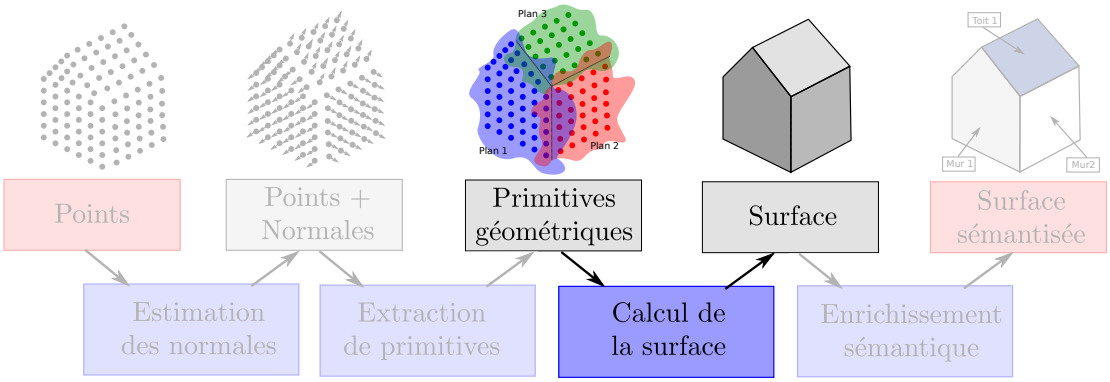
Wolf, L., Huang, X., Martin, I., and Metaxas, D. N. (2006). Patch-based texture edges and segmentation. In *ECCV*. LNCS 3952, pp. 481–493.

Zhang, W. and Kosecká, J. (2006). Nonparametric estimation of multiple structures with outliers. In *ECCV*, pages 60–74.

Zuliani, M., Kenney, C. S., and Manjunath, B. S. (2005). The multiRANSAC algorithm and its application to detect planar homographies. In *ICIP (3)*, pages 153–156.

Troisième partie

Reconstruction de surfaces planaires par morceaux



Chapitre 8

Reconstruction de surfaces

8.1 Introduction

Les modèles 3D de bâtiments basés sur des données réelles sont utilisés dans de nombreuses applications : le BIM pour des rénovations, la navigation virtuelle dans les bâtiments ou encore les simulations physiques telles que l'éclairage, les propriétés acoustiques ou les calculs d'accessibilité, etc. La reconstruction de tels modèles touche à des secteurs économiques aussi divers que la sauvegarde et la valorisation de patrimoine, la construction, l'industrie du cinéma ou le jeu vidéo.

Les applications qualitatives, nécessitant un très haut niveau de détail, comme pour un rendu visuel très précis, utilisent généralement des maillages fins. Cependant pour des applications quantitatives, le besoin de précision est limité et des modèles idéalisés sont généralement employés, mieux adaptés aux très grandes scènes. De plus pour l'attribution d'information sémantique pour la création de BIM ou de modèles de villes (par exemple CityGML), une géométrie simple est un atout.

Comme exposé dans la partie d'introduction, un marché s'est développé sur la reconstruction de surfaces. Les bâtiments sont communément recréés manuellement à partir de nuages de points laser ou photogrammétriques, en ajoutant itérativement des primitives géométriques simples. Cette méthode est coûteuse tant financièrement qu'en temps de production.

La reconstruction automatique est un domaine très étudié depuis ces dernières années. Ce chapitre se focalise en particulier sur les surfaces planaires par morceaux qui sont les surfaces les plus courantes dans le cadre des bâtiments et plus généralement dans des environnements créés par l'homme. Les données utilisées sont des images de profondeur laser.

Une méthode de reconstruction efficace doit s'accommoder des inconvénients du laser.

Les lasers sont généralement placés sur un trépied (position fixe), et le nombre de stations par zone est limité. De nombreuses zones de la scène sont invisibles, soit par concavité, soit par occultation. Même en augmentant le nombre d'acquisitions pour la scène, certaines parties restent cachées (dessous de table, arrière des piliers...), ceci est dû aux contraintes de placement du laser. Il faut donc être capable d'interpoler les surfaces dans l'invisible.

Enfin, il est nécessaire de gérer l'anisotropie locale et les différences de densités produites par le laser. L'angle d'incidence du laser ainsi que la distance des surfaces à l'origine du capteur influent sur la densité d'échantillonnage. De plus ce mode d'acquisition peut produire des zones très denses aux pôles (à la verticale du laser).

8.2 Travaux précédents

La reconstruction de surfaces à partir de nuages de points a été un sujet d'intérêt ces dernières années. Un très grand nombre de méthodes de reconstruction sont listées dans [Berger et al., 2014].

En 2006, Kazhdan et al. [2006] utilisent une formulation de Poisson pour reconstruire la surface. Dey and Goswami [2006] ont utilisé une triangulation de Delaunay en 3D.

Les points aberrants influant beaucoup sur la qualité des surfaces reconstruites, [Avron et al., 2010; Chazal et al., 2011; Song, 2010; Sotoodeh, 2006] ont développé des méthodes robustes, basées sur la détection de points aberrants, ou encore des normes ou des distances robustes.

Les méthodes adaptatives comme les *Moving Least Squares* (MLS) de Wang et al. [2009], ou les algorithmes de Mellado et al. [2012]; Unnikrishnan et al. [2010], modifient le niveau de lissage en fonction du bruit local. Elles permettent une reconstruction fidèle dans les zones de faible bruit tout en évitant une surface granuleuse là où le bruit est important. Elles améliorent ainsi les résultats des reconstructions où le paramètre de lissage est global.

Les nuages de points avec des zones manquantes sont fréquents. Pour gérer ce problème, les méthodes précédentes ferment les surfaces parfois en créant une sorte de bulle (c'est le cas de la reconstruction de Poisson). Shalom et al. [2010] utilisent les cônes de visibilité pour avoir une information de volume vide. Plus récemment, Giraudot et al. [2013] reconstruisent la surface en estimant le signe (intérieur ou extérieur) d'un ensemble de points dans l'espace puis définissent une fonction régulière dont les zéros sont la surface souhaitée.

Les méthodes ci-dessus font l'hypothèse d'une surface sans angle ou arête vive, et le résultat est un maillage, et non un modèle idéalisé. Les approches qui recherchent des modèles idéalisés détectent des primitives dans le nuage de points [Fayolle and Pasko, 2013; Schnabel et al., 2007] et partie II, puis reconstruisent les points anguleux à l'intersection de celles-ci. La création d'une intersection est souvent liée à la proximité des primitives et à la présence de points proches de l'intersection [Chen and Chen, 2008; Jenke et al., 2008; Lafarge and Alliez, 2013; Li et al., 2011]. La notion de proximité est un frein à la complétion de larges zones invisibles, ou peu denses. Arikan et al. [2013]; Chen and Chen [2008] pallient cet inconvénient en autorisant l'utilisateur à intervenir pour déterminer les plans à prolonger.

Les approches qui anticipent la présence de grandes zones invisibles utilisent des connaissances fortes sur la surface à reconstruire. Dans [Castellani, 2002], les primitives sont prolongées jusqu'à l'intersection d'un mur ou d'un sol préalablement détecté. Cependant, cette méthode agit sur les polygones détectés et ne garantit pas de surface étanche.

Schnabel et al. [2009] calculent une coupe de graphe sur un graphe aligné sur une grille régulière en 3D, proche des primitives détectées. Cette méthode ne reconstruit que les parties visibles, avec une possibilité d'artefact de discrétisation. L'hypothèse de monde Manhattan, utilisée dans de nombreuses publications [Budroni and Böhm, 2010; Furukawa et al., 2009a,b; Vanegas et al., 2012], suppose que tous les plans sont orientés selon l'une des trois directions du trièdre orthonormé. Cette hypothèse forte produit de très bons résultats pour des scènes simples, mais s'avère trop restrictive pour des scènes complexes. C'est par exemple le cas pour des scènes d'intérieur, avec beaucoup de mobilier et d'objets.

Dans [Chauve et al., 2010], les régions planaires peuvent s'étendre dans les zones cachées. Les plans détectés sont utilisés dans un complexe polyédral pour partitionner l'espace. Les cellules sont étiquetées pleines ou vides et la surface est l'interface

entre l'espace vide et l'espace plein. La surface est par définition fermée et sans auto-intersection. De plus pour que la complétion dans l'invisible soit plausible, des hypothèses de plans supplémentaires, des « plans fantômes », sont générées. Les régions détectées sont simplifiées pour être représentées par des polygones et, pour certaines arêtes de ces polygones, des plans passant par celles-ci sont générés. L'idée sous-jacente est que dans un environnement humain, une arête est l'intersection de plans qui sont le plus souvent orthogonaux. Cependant, Chauve et al. [2010] utilisent des α-shapes pour construire les polygones. Or les α-shapes sont sensibles à l'anisotropie et au bruit des données. Il en résulte des hypothèses de plans fantômes trop nombreuses et/ou fausses. Pour déterminer la surface dans l'invisible, la régularisation utilisée est une pénalisation de l'aire totale de la surface. Cette mesure de simplicité de la surface est inadaptée dans un environnement intérieur où existent des objets de faible volume mais avec une grande surface (tables, portes...).

8.3 Méthode

L'algorithme présenté ici n'a pas pour but de reconstruire une scène avec la plus grande précision possible, mais de produire une approximation planaire par morceaux à une échelle de précision σ donnée. La tolérance σ exprime le niveau de détail souhaité. Il est dicté par le type d'application. Par exemple, un σ de l'ordre de 5 cm ne considèrera pas des détails tels que des cadres de porte, des plinthes, des interrupteurs ou autres éléments fins.

Pour gérer l'anisotropie et les différences de densité dans les acquisitions, une normalisation des données est requise. Cette normalisation affecte une influence moindre aux points des zones denses, et un poids plus important lorsque la densité est plus faible. (Cette normalisation est à bien distinguer de l'estimation du degré de confiance dans la reconstruction qui, au contraire, est plus important dans les zones denses et plus faible dans les zones peu ou irrégulièrement échantillonnées.) Une approche courante pour prendre en compte l'anisotropie est de discrétiser l'espace selon une grille régulière de voxels [Chauve et al., 2010]. Cependant cette solution possède plusieurs inconvénients. Premièrement cela peut introduire des artefacts lorsque la surface reconstruite suit la grille de voxels. Ensuite, la taille des voxels est un nouveau paramètre qui influence le niveau de détail et la complexité des calculs. Enfin, si la taille des voxels est trop petite, des trous apparaissent dans les primitives contenant des zones à faible densité de points.

Les données traitées étant des images laser, le mode d'acquisition et la structure en image de profondeur sont exploités. Chaque point se voit attribuer un poids similaire à celui défini chapitre 2, pondéré par l'orientation de la surface en ce point.

Étapes de la méthode

Bien que chaque étape diffère de [Chauve et al., 2010], le déroulement global de l'algorithme est similaire.

- **Caractérisation des points.** Les normales sont calculées en chaque point (cf. chapitre 5), ainsi que le poids de normalisation de chaque point.
- **Détection de primitives** Des plans sont extraits dans le nuage de points (cf. chapitre 6) et fusionnés si nécessaire en utilisant les critères statistiques du chapitre 7.
- **Création des primitives fantômes.** Les bords des primitives sont extraits sous forme de polygones dans l'image laser, puis les arêtes des polygones sont

classifiées pour déterminer celles qui donneront naissance à un plan fantôme. De plus, des plans parallèles à la primitive détectée sont aussi générés, permettant ainsi de reconstruire des objets fins et plats (portes, plateaux de tables, etc.) dont les surfaces matérialisant l'épaisseur n'ont pu être détectées à l'étape d'extraction de primitives.

- **Construction du complexe polyédral.** Un complexe polyédral est créé par insertion successive des plans détectés et des fantômes. La convexité de chaque cellule est garantie par construction.

- **Définition d'une énergie.** A chaque cellule est attribuée une variable binaire (de valeur 0 ou 1), définissant l'état de la cellule, vide ou pleine. La surface reconstruite est l'interface entre le plein et le vide. Une énergie est définie sur ces variables binaires qui pénalise l'écart aux points détectés ainsi que la complexité de la surface (aire, longueur des arêtes ou nombre de coins).

- **Optimisation.** Le problème de minimisation entière de cette énergie est transformé en problème linéaire. La surface finale est extraite d'une solution au problème relaxé. Celle-ci s'avère proche de la solution optimale.

Contributions

Les principales contributions portent sur les différents aspects de l'algorithme.

- **Normalisation.** Tirant profit du type d'acquisition, une pondération des points est utilisée sans décimation ou perte de précision.

- **Primitives fantômes.** Nous définissons de nouveaux procédés pour la création de polygones et la génération de plans fantômes, pour notamment reconstruire des objets fins.

- **Régularisation.** Trois types de régularisation sont proposés. L'approche classique consistant à minimiser l'aire de la surface reconstruite, notamment pour faire des reconstructions plausibles dans l'invisible, mène à des potentiels d'ordre 2, qu'il est possible de résoudre par coupe de graphe. La minimisation de la longueur des arêtes vives et du nombre de coins de la surface, requiert des potentiels d'ordres supérieurs, respectivement 4 et 8. De telles contraintes d'ordres supérieurs ont été utilisées dans des simplifications gloutonnes de maillages mais pas pour des approches de minimisation globale. Notre algorithme a des points communs avec les algorithmes 2D de Schoenemann et al. [2009]; Shekhovtsov et al. [2012], où une régularisation d'ordre supérieur a été utilisée. Plus particulièrement, Schoenemann et al. [2009] découpent aussi l'espace en un ensemble de cellules convexes et utilisent une formulation en programme linéaire pour l'étiquetage des cellules. Cependant le problème résolu n'est pas le même, et la méthode ne se généralise pas directement à la 3D.

- **Minimisation de l'énergie.** L'énergie du problème discret est efficacement minimisée via la relaxation du problème sous forme de problème linéaire.

8.4 Hypothèses de surfaces

Le problème d'extraction de surface est formulé sous la forme d'un problème d'étiquetage binaire de volume. La surface est l'interface entre les volumes avec l'étiquette

« plein »et ceux avec l'étiquette « vide ». Ce type de formulation assure une surface fermée et sans auto-intersection.

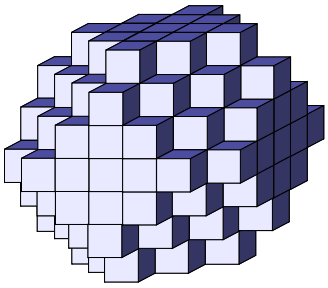
Les volumes sont créés en partitionnant l'espace avec des plans pour former un complexe polyédral (section 8.4.1). Les hypothèses de plans sont générées à partir du nuages de points (section 8.4.2). Pour assurer une reconstruction plausible dans l'invisible, des plans supplémentaires sont proposés, basés sur les plans détectés précédemment (section 8.4.3). Dans le but de limiter la complexité du complexe polyédral, l'extension des plans fantômes est spatialement limitée (section 8.4.4).

8.4.1 Complexe polyédral

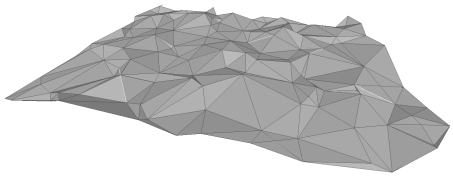
Pour la reconstruction 3D, deux approches ont principalement été utilisées : une grille régulière de voxels et une triangulation de Delaunay 3D.

La complexité de la première croît cubiquement en fonction du niveau de détail désiré et induit un biais dans l'estimation de la surface ainsi que de l'aliasing (figure 8.1a). De plus, l'utilisation de voxels est incompatible avec l'idée de régularisation sur la longueur des arêtes et le nombre de coins.

Lorsqu'il y a du bruit dans les données, la triangulation de Delaunay produit une surface bruitée qui n'est pas parfaitement planaire (figure 8.1b). Lafarge and Alliez [2013] pallient ce problème en enlevant les points proches des plans et en rééchantillonnant uniformément le plan, garantissant ainsi que les facettes correspondant aux plans visibles soient présentes dans la triangulation. L'intersection des primitives est traitée par un critère de proximité, ce qui interdit la prolongation de celles-ci loin dans l'invisible.



(a) Reconstruction d'une sphère avec des voxels.

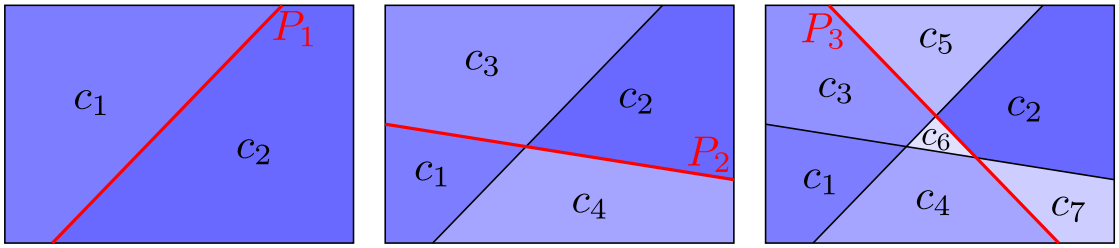


(b) Surface planaire obtenue avec une triangulation de Delaunay d'un nuage de points bruités.

FIGURE 8.1 – Inconvénients liés à l'utilisation de voxels ou d'une triangulation de Delaunay.

L'approche utilisée ici est celle de [Chauve et al., 2010]. L'espace est partitionné en un complexe polyédral en utilisant un arrangement de plans.

Les plans sont insérés itérativement dans l'arrangement. Chaque plan coupe l'espace en deux demi-espaces, il en résulte un ensemble de cellules (volumes), toutes convexes par construction. La figure 8.2 illustre la création d'un arrangement. L'arrangement final est indépendant de l'ordre d'insertion des plans. Pour des raisons pratiques, le graphe d'adjacence entre cellules est maintenu et mis à jour durant l'insertion des plans. Il permet un temps d'accès constant à tous les voisins d'une cellule donnée. L'ensemble des cellules est noté $\mathcal{C}$, celui des facettes (interface entre cellules) est $\mathcal{F}$, celui des arêtes est $\mathcal{E}$ et celui des sommets est $\mathcal{V}$.



La construction est indépendante de l'ordre d'insertion des droites. Toutes les cellules créées sont convexes.

FIGURE 8.2 – Création d'un arrangement de lignes 2D par insertions de trois droites dans une boîte englobante.

8.4.2 Primitives visibles

Les primitives visibles sont détectées dans le nuage de points laser (chapitre 6). Les méthodes les plus communes de détection de primitives sont le RANSAC ([Schnabel et al., 2007]) et la croissance de régions (chapitre 6).

La croissance de régions est employée ici pour tirer parti de la structure des acquisitions laser. L'exploitation de la 8-connectivité de l'image de profondeur permet une croissance rapide tout en gérant les effets d'anisotropie. Contrairement à Chauve et al. [2010], qui regroupent les points dans des voxels pour ré-échantillonner régulièrement le nuage de points, tous les points du nuage sont utilisés.

Les primitives sont ensuite fusionnées en utilisant les critères statistiques définis au chapitre 7 pour regrouper les segments appartenant aux mêmes plans.

8.4.3 Primitives fantômes

Les primitives fantômes sont des hypothèses de plans non détectés dans le nuage de points. Du fait des occultations et des contraintes de placement des lasers, un grand nombre de primitives ne sont pas vues lors de l'acquisition, comme le dessous des tables et les côtés invisibles des meubles. Les primitives trop fines, comme les bords de fenêtres et des tables, sont ignorées par l'étape de détection de primitives.

Ces plans sont cependant nécessaires à la reconstruction d'une surface plausible. Ils sont classés en deux catégories, les plans orthogonaux à des plans détectés, qui correspondent aux bords des primitives identifiées, et les plans fantômes parallèles définissant l'épaisseur des objets.

Fantômes orthogonaux

Les plans fantômes orthogonaux prennent appui sur les bords des primitives détectées, qu'il faut déterminer. Les primitives détectées ont initialement un contour indéfini (c'est seulement un ensemble de points). Le but est ici de créer un polygone correspondant aux bords du segment de points.

La figure 8.3 illustre ce procédé. La primitive P_1 est détectée dans le nuage de points, ses bords sont ensuite extraits et des fantômes sont générés (Fantôme_1 et Fantôme_2).

α-shapes : Une méthode pour produire un polygone correspondant à un ensemble de points est d'utiliser les α-shapes. C'est l'approche utilisée dans [Chauve et al., 2010], avec $\alpha = \sigma^2$. Les points sont projetés sur le plan associé à la primitive puis l'α-shape est calculée pour les nouveaux points en 2D. S'en suit une étape de simplification par fusion gloutonne jusqu'à une déviation de niveau σ.

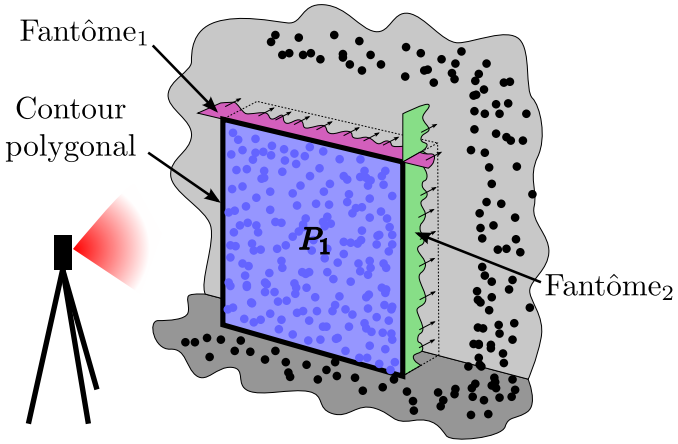
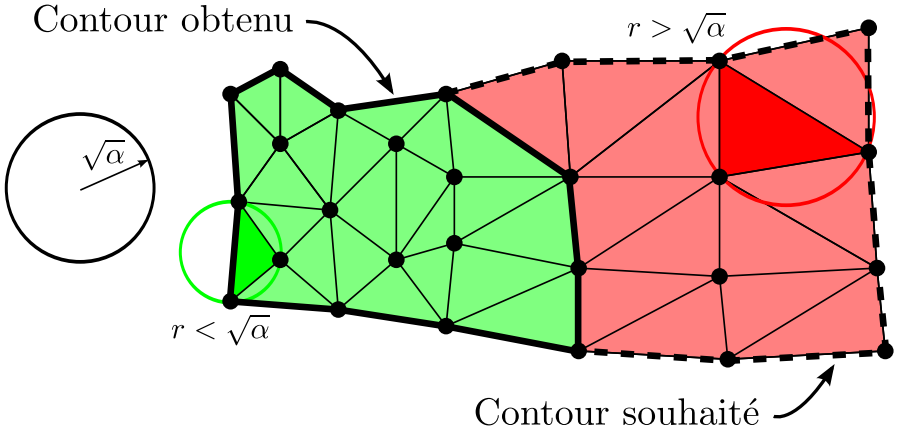


FIGURE 8.3 – Génération de fantômes orthogonaux à la primitive détectée dans le nuage de points pour modéliser des surfaces invisibles (en rose et vert).

Le principal inconvénient de cette approche est la création de plusieurs composantes connexes (isolation des points) lorsque l'écart entre les points est supérieur à $\sqrt{\alpha}$. Ceci est typiquement le cas lorsque l'incidence du laser est grande et que les primitives sont lointaines ou étendues. La figure 8.4 illustre ce phénomène. Le contour obtenu (en trait plein) n'est pas le contour souhaité (en pointillés). Augmenter α permet de récupérer les points isolés mais induit une perte de précision dans l'estimation du contour dans les zones denses. Une solution pourrait-être d'utiliser un α adaptatif, en fonction de l'incidence du laser et de l'orientation de la primitive.



En vert, les triangles dont le cercle circonscrit est de rayon $r < \sqrt{\alpha}$ sont inclus dans l'α-shape. Les autres, en rouge sont rejetés.

FIGURE 8.4 – Construction du contour polygonal par α-shapes lorsqu'il y a de l'anisotropie dans les données.

Travail dans l'image laser : La solution que nous retenons pour gérer le problème d'anisotropie et de variation de densité est de travailler directement dans l'image de profondeur. Les contours des primitives sont des chaînes de pixels. Les petites régions sont écartées pour des raisons de robustesse au bruit et aux objets non désirés. Les polygones à trous sont ensuite créés par simplification des contours, ce qui gomme l'aliasing (effet de discrétisation), effet dominant par rapport au bruit pour les nuages laser. Pour cela, les arêtes sont itérativement fusionnées. Considérant deux segments d'arc ab et bc dans

l'image laser, si la distance de b à ac est inférieure à un seuil δ fixe en nombre de pixels, les deux segments sont fusionnés. La simplification s'effectue par ordre des distances croissantes.

Dans le cas d'une acquisition sphérique, les lignes droites dans l'espace sont des courbes dans l'image de profondeur. Comme illustré par la figure 8.5, pour mesurer la distance d d'un point b en 2D à un segment 2D ac, ce sont leurs projections A, B et C sur la sphère unité qui sont considérées. Soit B' la projection géodésique de B sur l'arc AC et soit b' la projection de B' dans l'image de profondeur. Les segments de contour ab et bc sont fusionnés en ac si la distance entre b et b' est inférieure à un seuil δ.

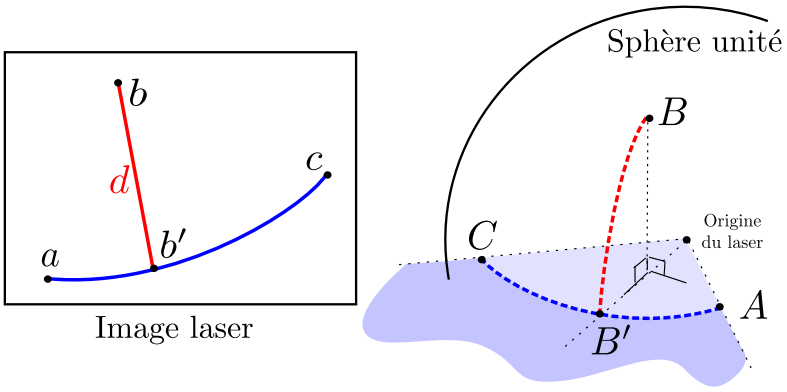


FIGURE 8.5 – Distance à une ligne dans l'image laser, par projection sur la sphère unité.

Pour éviter toute dérive, les extrémités des segments fusionnés sont mémorisées et une nouvelle fusion n'est autorisée que si l'ensemble de ces points est à une distance inférieure à δ du nouveau segment. Ceci garantit qu'à chaque itération, les nouveaux segments de contour sont au plus à une distance δ des segments initiaux. Ainsi la distance de Haussdorf de l'ancien contour au nouveau polygone, ajustée d'après la projection sphérique, est bornée par δ. C'est la garantie que les nouveaux bords sont au plus à δ du contour original. L'algorithme 4 résume la procédure de simplification des polygones.

Type d'arêtes Les arêtes 3D sont obtenues par projection des bords du polygone sur le plan 3D de la primitive. Ces arêtes 3D sont classifiées selon trois catégories.

- **Les arêtes d'adjacence :** à l'intersection visible de deux primitives détectées.
- **Les arêtes occultées :** lorsque la primitive est derrière les points qui sont de l'autre côté de l'arête dans l'image de profondeur.
- **Les arêtes occultantes :** lorsque la primitive est devant les points qui sont de l'autre côté.

La figure 8.6 montre les trois types de bords.

Générer un fantôme pour les arêtes d'adjacence n'a pas de sens, car les deux primitives détectées décrivent déjà cette arête dans l'arrangement de plans.

Comme au niveau d'une arête occultée, la surface continue généralement derrière l'élément placé devant la primitive, seules les arêtes occultantes vont générer un plan fantôme.

Une arête est occultante si :

1. elle n'est pas une arête d'adjacence. Deux primitives P et P' sont considérées comme adjacentes si au moins deux points (un pour chacune) sont voisins dans

Algorithme 4 Simplification des polygones.

Paramètre: V # Ensemble des sommets ordonnés du polygone

```
    # Initialisation de la mémoire des sommets après simplification
Mem ← ∅
pour tout i ∈ {1, ..., |V|} faire
  Mem(V[i], V[i+1]) ← ∅      # V[|V|+1] = V[1]
fin pour

    # Simplification
tant que ∃ i ∈ {1, ..., |V|}, tq (V[i], V[i+1], V[i+2]) non observé faire
  a ← V[i], c ← V[i+2]
      # Changement d'espace vers la sphère unité
  A ← vers_sphere(a)
  C ← vers_sphere(c)

      # Ensemble des points à observer
  Points_pour_dist ← Mem(V[i], V[i+1]) ∪ Mem(V[i+1], V[i+2]) ∪ {V[i+1]}
  OK_simplification ← Vrai

  pour tout b ∈ Points_pour_dist faire
    B ← vers_sphere(b)
    B' ← proj_AC(B)      # Projection de B sur l'arc AC
    b' ← vers_image(B')      # Changement d'espace vers l'image 2D
    si ||b b'||_2 > δ alors
      OK_simplification ← Faux
    fin si
  fin pour

  si OK_simplification alors
    V ← V \ V[i+1]      # Mise à jour de V
    Mem(a, c) ← Points_pour_dist
  fin si
fin tant que
```

l'image de profondeur et si la projection d'au moins un point de P est proche de P' (respectivement un point de P' est proche de P).

2. la majorité des points la constituant dans l'image de profondeur est proche de points situés derrière la primitive.

La figure 8.7 est le résultat de la création de polygones sur une scène laser. Le choix du type de bord est déterminé en observant l'ensemble des pixels correspondant au bord du polygone. Pour chaque pixel de bord, nous observons son voisinage (une fenêtre) et déterminons si c'est un pixel d'adjacence, caché ou cachant. Enfin, le statut du segment est déterminé en fonction de la proportion de pixel cachant. Comme attendu, les marches dont on voit le plateau et la contremarche ne génèrent pas de plan fantôme (arêtes en bleu). Lorsqu'on s'élève dans l'escalier, le plateau devient invisible, il est alors généré par une arête occultante de la contremarche (en rouge).

Dans [Chauve et al., 2010], les fantômes sont générés pour chaque arête du polygone associé à la primitive, sauf si celle-ci est considérée comme adjacente à une autre primitive

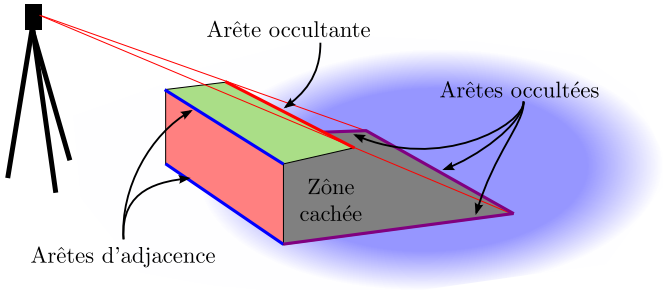
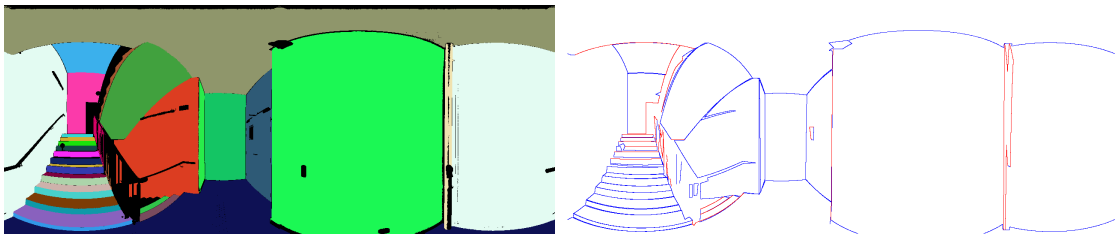


FIGURE 8.6 – Les différents types de segments pour les bords des primitives.



À gauche, les primitives. A droite, le résultat de la création de polygones et de la décision de génération de primitives fantômes. Les segments rouges génèrent un fantôme, les bleus, non.

FIGURE 8.7 – Fantômes orthogonaux générés pour une scène d'escalier.

(à distance inférieure à σ d'une autre primitive). Comparée au rejet des segments d'adjacence et des segments cachés, cette approche mène à la génération de nombreux fantômes inutiles. Cependant, l'algorithme de Chauve et al. [2010] a été développé pour la photogrammétrie et pour des nuages de points non structurés. Le classement en trois catégories est dès lors beaucoup plus difficile que dans le cas du laser (nuage structuré).

Fantômes parallèles

Les scènes d'intérieur contiennent de nombreuses zones cachées et des objets planaires fins (portes, fenêtres, tables...). En général, les points ne décrivent qu'un côté de ces objets et leur épaisseur est trop faible pour être détectée comme une primitive. Pour reconstruire correctement l'objet, un fantôme parallèle à la primitive détectée et à distance égale à l'épaisseur de l'objet doit être généré. L'épaisseur de l'objet est estimée par extrusion de la primitive dans la direction opposée à sa normale. Plusieurs épaisseurs sont testées, et l'épaisseur retenue est celle pour laquelle il y a violation des conditions de visibilité (intersection avec le cône de visibilité de l'acquisition) pour un nombre suffisant de rayons du centre de l'acquisition vers les points détectés. Les épaisseurs sont testées par ordre croissant avec un pas constant. (Figure 8.8)

8.4.4 Optimisation et ordre d'insertion

En dimension 3, la complexité d'un arrangement de plans est cubique en le nombre de plans insérés. L'algorithme utilisé pour la construction de l'arrangement est celui de Edelsbrunner et al. [1986].

Pour limiter le nombre de cellules construites et ainsi la complexité de la construction, il faut se limiter à l'insertion de plans qui ont une chance raisonnable d'être utilisés dans la reconstruction. De plus, il est possible de limiter l'extension des plans, en utilisant les plans déjà présents dans l'arrangement comme limitation. Cependant, utiliser cette

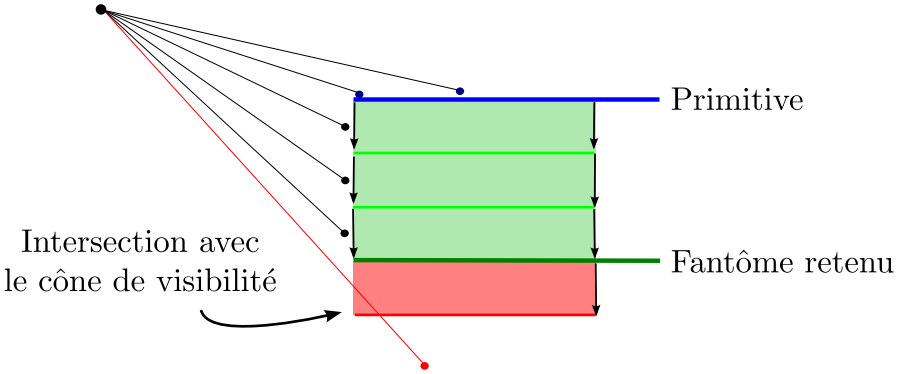
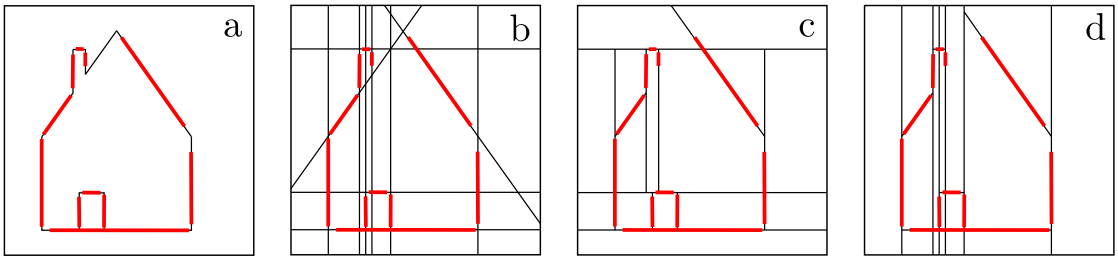


FIGURE 8.8 – Génération d'une primitive fantôme parallèle à la primitive détectée dans le nuage de points.

limitation crée un biais dans l'ordre d'insertion et peut rendre inaccessible la surface souhaitée (figure 8.9).



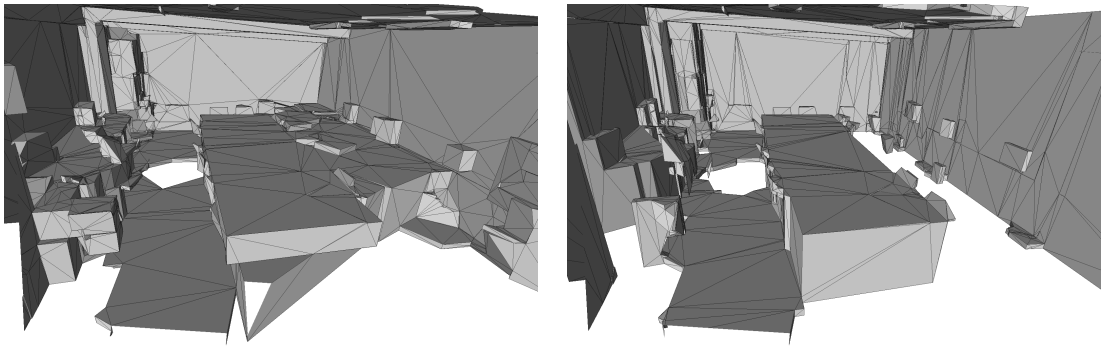
(a) Segments sur un exemple synthétique. (b) arrangement construit sans limitation des plans. (c) arrangement avec limitation, insertion dans l'ordre les segments horizontaux, verticaux et obliques. (d) arrangement avec limitation, insertion dans l'ordre les segments verticaux, horizontaux et obliques

FIGURE 8.9 – Différences d'arrangements suivant l'ordre d'insertion des primitives.

Chauve et al. [2010] proposent une solution dépendante du type de données pour limiter les plans. Pour les données aériennes, les plans horizontaux sont vraisemblablement les primitives détectées avec le plus de précision et en plus grand nombre. Ces plans sont insérés en premier, puis les plans verticaux limités par les premiers. Enfin, tous les autres plans sont ajoutés dans l'arrangement, limités par les primitives horizontales et verticales. Pour les nuages de points créés à partir du sol, ce sont les plans verticaux qui sont insérés en premier, suivis des horizontaux et des obliques.

La figure 8.10 montre la différence entre les deux modes d'insertion dans l'arrangement pour la méthode de Chauve et al. [2010]. On observe que les deux reconstructions sont particulièrement différentes. C'est principalement dans l'invisible que les variations sont observables puisque la reconstruction dans les parties visibles repose essentiellement sur les primitives effectivement détectées, qui sont toujours présentes en totalité dans l'arrangement.

La stratégie d'insertion que nous proposons est moins contraignante. Pour garder une indépendance sur l'ordre d'insertion des primitives observables, les plans générés à partir de celles-ci sont des plans traversants (sans limitation) pour l'arrangement. Seules les primitives fantômes sont limitées, mais uniquement par la primitive qui les a générées. Le seul ordre à respecter est alors de ne pas insérer une primitive après les fantômes qu'elle a créés.



À gauche, les plans horizontaux sont insérés en premier, à droite les primitives verticales ont la priorité.

FIGURE 8.10 – Illustration de l'influence de l'ordre d'insertion des primitives pour l'algorithme de Chauve et al. [2010].

Si nous pouvons nous permettre d'insérer des plans sans limitations pour les primitives, contrairement à ce que font Chauve et al. [2010], c'est parce que nous avons un moins grand nombre (mais de meilleures) d'hypothèses de plans grâce à notre méthode de détermination des bords de primitives, plus régulière et plus robuste que les α-shapes.

8.5 Reconstruction de surface

8.5.1 Objectif

Le but est de reconstruire une surface qui se conforme au mieux aux données dans les parties visibles, tout en étant simple et plausible dans les zones invisibles. La conformité aux données signifie que la surface doit passer près des points observés du nuage (ou tout du moins près des points participant à une primitive extraite) tout en n'intersectant pas le cône de visibilité de l'acquisition.

8.5.2 Définition du problème

La reconstruction de surface est un problème d'étiquetage binaire (plein ou vide) de l'ensemble des cellules de l'arrangement de plans défini à la section 8.4. La surface est l'interface entre les cellules avec l'étiquette vide et celles avec l'étiquette plein. C'est le sous-ensemble des facettes $\mathcal{F}$, tel que chaque élément soit adjacent à une cellule vide et à une cellule pleine. Cette définition garantit une surface étanche et sans auto-intersection.

Soit $x_c \in \{0, 1\}$ la variable discrète associée à la cellule $c \in \mathcal{C}$. 0 représente une cellule vide, 1 une cellule pleine. Soit $\mathbf{x} = (x_c)_{c \in \mathcal{C}}$ le vecteur d'occupation des cellules. Chaque $\mathbf{x}$ représente une surface possible.

La solution du problème est le vecteur $\mathbf{x}$ minimisant une énergie de la forme :

$$E(\mathbf{x}) = E_{data}(\mathbf{x}) + E_{regul}(\mathbf{x}) \tag{8.1}$$

Cette énergie est composée de deux termes, E_{data} et E_{regul}. E_{data} pénalise un écart aux données observées. E_{regul}, quant à elle, traduit les *a priori* sur la scène, à savoir un environnement humain. Elle agit principalement dans les parties cachées.

Notations Les notations suivantes seront utilisées.

Chaque point $p \in \mathcal{P}$, où $\mathcal{P}$ est le nuage de points, est associé à une primitive Π_p. La primitive Π_p est soit un plan P_p, soit un ensemble de points à l'infini, noté Π_∞, soit un ensemble de points à coordonnées finies mais n'appartenant pas à une primitive, noté Π_0. Π_∞ correspond par exemple aux rayons du laser passant à travers des fenêtres. Π_0 inclut entre autres les points échantillonnés sur de petits objets, n'ayant pas généré de primitives visibles. Enfin, il est possible que des points soient considérés comme à l'infini, dû au matériau de l'objet touché. Ces points sont généralement isolés dans le nuage, ils sont exclus de $\mathcal{P}$.

Un point p est associé à une facette f_p du complexe polyédral. C'est la facette issue de P_p, contenant le projeté orthogonal de p sur P_p.

Pour une facette $f \in \mathcal{F}$, sont notés f^+ et f^- les côtés de f, définis par la normale au plan auquel appartient f. Lorsque le nuage de points contient un unique point de vue, il est toujours possible de faire correspondre f^+ avec le côté visible et f^- avec le côté caché.

Les cellules correspondant à f^+ et f^- sont notées respectivement c_{f^+} et c_{f^-} et nous notons $x_{f^+} = x_{c_{f^+}}$, $x_{f^-} = x_{c_{f^-}}$.

8.5.3 Attache aux données

Pour obtenir une surface conforme aux observations, l'attache aux données E_{data} pénalise deux types d'écart aux données. E_{data} est la somme de deux termes, E_{prim}, l'attache aux primitives, et E_{vis} pénalisant les violations de visibilité :

$$E_{data}(\mathbf{x}) = E_{prim}(\mathbf{x}) + E_{vis}(\mathbf{x}) \tag{8.2}$$

Attache aux primitives. E_{prim} pénalise les points qui n'appartiennent pas à la surface reconstruite. Une pénalité est donnée à un point si sa facette correspondante f_p ne fait pas partie de la surface reconstruite, c.-à-d. si $x_{f^+} = x_{f^-}$ (que la valeur soir 0 ou 1).

Comme il peut y avoir du bruit dans les mesures et des erreurs dans l'estimation des points, une tolérance autour de P_p est prise en compte. Soit σ cette tolérance, les cellules contenant les extrémités du segment de droite orthogonal à P de longueur 2σ et centré sur la projection de p sur P_p sont notées $c_p^{\sigma^-}$ et $c_p^{\sigma^+}$, suivant l'orientation de P_p. Soient $x_p^{\sigma^-}$ et $x_p^{\sigma^+}$ les variables correspondantes. La pénalité est définie de la manière suivante :

$$|1 - x_p^{\sigma^+} - x_p^{\sigma^-}|$$

qui vaut 0 lorsqu'il y a une transition, 1 autrement. Il est à noter que les cellules entre $c_p^{\sigma^-}$ et $c_p^{\sigma^+}$ sont ignorées. En pratique, il n'est pas nécessaire d'imposer une unique transition entre $c_p^{\sigma^-}$ et $c_p^{\sigma^+}$, c'est la régularisation qui entre en jeu pour éviter la création de surfaces complexes. La figure 8.11 illustre la construction de E_{prim} pour un exemple avec trois droites.

Pour prendre en compte l'anisotropie et les variations de densité de l'acquisition laser, l'influence de chaque point est pondérée. Le poids est proche de celui défini dans le chapitre 2. Il mesure l'aire relative que représente chaque point du nuage pour la surface sous-jacente par rapport à la densité locale d'acquisition des points. Ce poids est sans dimension, homogène à un nombre d'unités d'aire rapporté à l'échelle σ^2. On peut le voir aussi comme le nombre de σ^2 que représente le point.

Ce poids dépend de la densité locale de points sur la surface, et de l'orientation de la surface au point p considéré. Comme au chapitre 2, l'acquisition est supposée sphérique.

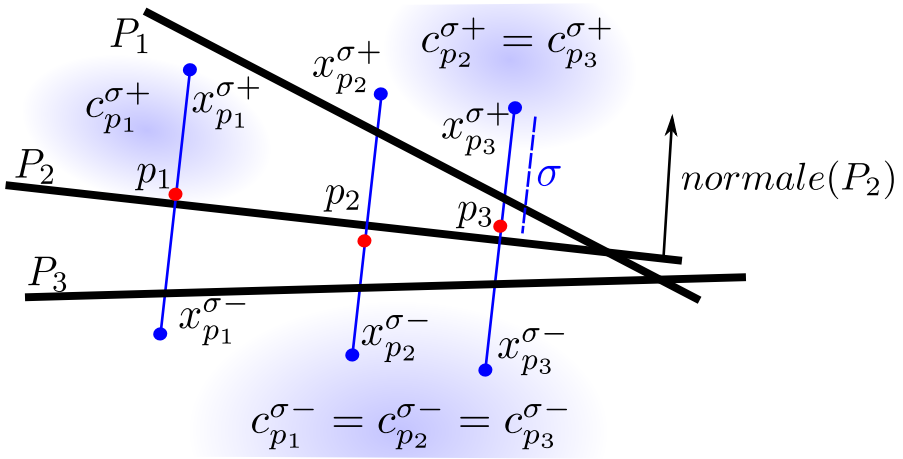


FIGURE 8.11 – Tolérance pour la pénalité des points appartenant à une primitive (ici P_2).

La direction de chaque point est donnée par ses coordonnées polaires (θ, ϕ), avec l'azimut $\theta \in [0, 2\pi[$ et l'angle polaire $\phi \in [0, \pi]$ (l'angle avec la verticale). L'acquisition discrétise les angles en pas constants Δ_θ et Δ_ϕ.

Soit un point p, acquis dans la direction (θ, ϕ), et à distance d du point d'acquisition. p est acquis avec une incidence $\psi \in [0, \pi/2[$, l'angle entre le rayon laser et la surface en p. Cette incidence est accessible après estimation des normales et peut être raffinée si p appartient à une primitive. Comme au chapitre 2, équation (2.2), le poids est relatif à l'aire de la projection du patch (fragment de surface) correspondant à la discrétisation angulaire $\Delta_\theta \times \Delta_\phi$ sur le plan tangent en p, et pondéré par l'angle polaire pour prendre en compte la densité d'échantillonnage (chapitre 2, figure 2.13):

$$w_p^{aniso} = \frac{d^2}{\sigma^2} \Delta_\theta \Delta_\phi \frac{\sin \phi}{\cos \psi} \tag{8.3}$$

Enfin, le terme de l'énergie relatif aux primitives est défini par :

$$E_{prim}(\mathbf{x}) = \sum_{p \in \mathcal{P} \setminus \Pi_0, \Pi_\infty} w_p^{aniso} |1 - x_p^{\sigma^+} - x_p^{\sigma^-}| \tag{8.4}$$

Violations de visibilité. $E_{vis}((x))$ pénalise les surfaces reconstruites sur la trajectoire entre les points du nuage et leur point de vue. Soient p un point du nuage et ω son point de vue, le segment $[\omega, p]$ ne doit pas couper la surface. Premièrement, la cellule contenant ω est étiquetée vide (le dispositif d'acquisition a forcément été placé dans un endroit vide). Ensuite, les facettes de transition intersectées par $[\omega, p]$ sont pénalisées. Une pénalité est payée pour chaque interface traversée par le segment.

Comme dans le cas du terme d'attache aux primitives, une tolérance σ est laissée. Les facettes f de plan P_f ne sont pas pénalisées si la distance orthogonale entre p et P_f est inférieure à σ (figure 8.12).

De même, pour tenir compte de l'anisotropie de l'acquisition et de l'orientation des surfaces, l'influence des points est pondérée par w^{aniso}. Cependant, contrairement au terme d'attache aux primitives, tous les points du nuage sont utilisés :

$$E_{vis}(\mathbf{x}) = \sum_{\substack{p \in \mathcal{P}, f \in \mathcal{F}, \\ [\omega,p] \cap f \neq \emptyset, d(p,P_f) > \sigma}} w_p^{aniso} |x_{f^+} - x_{f^-}| \tag{8.5}$$

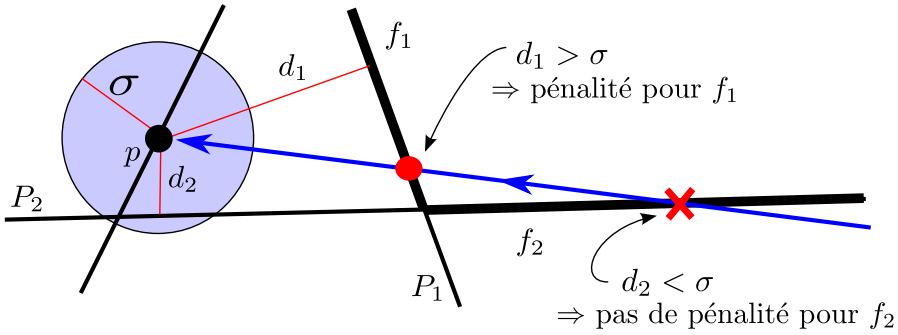


FIGURE 8.12 – Tolérance pour la pénalité de visibilité.

En comparaison de [Chauve et al., 2010], les attaches aux primitives et à la visibilité sont séparées. De plus, ici, pour la visibilité, tous les points sont utilisés et toutes les transitions sont pénalisées dans le cône de visibilité, pas seulement les surfaces mal orientées. Enfin Chauve et al. [2010] utilisent un coût constant pour les points sans prendre en compte l'anisotropie de l'acquisition.

8.5.4 Régularisation

L'objectif étant d'obtenir la surface plausible la plus simple, le terme de régularisation E_{regul} pénalise la complexité de la scène. Cette pénalisation s'applique à l'ensemble de la scène, et pas seulement aux parties cachées. En effet, il est souhaitable d'avoir une surface régulière aussi dans les parties visibles, et d'éviter ainsi la présence d'aliasing dû au bruit d'acquisition. Cependant, le poids associé à l'énergie de régularisation est plus faible que celui attribué à E_{data}, pour assurer la conformité de la surface reconstruite avec les observations.

L'énergie de régularisation est la somme de trois termes qui peuvent être utilisés séparément ou conjointement : un terme d'aire, un terme d'arête et un terme de coin :

$$E_{regul}(\mathbf{x}) = \lambda_{aire}\, E_{aire}(\mathbf{x}) + \lambda_{arete}\, E_{arete}(\mathbf{x}) + \lambda_{coin}\, E_{coin}(\mathbf{x}) \tag{8.6}$$

Terme d'aire

$E_{aire}(\mathbf{x})$ pénalise l'aire de la surface reconstruite. Une notion de simplicité de surface est de vouloir la surface d'aire minimale. Notant a_f l'aire de la facette f, pour f une facette à l'interface entre deux cellules, on définit :

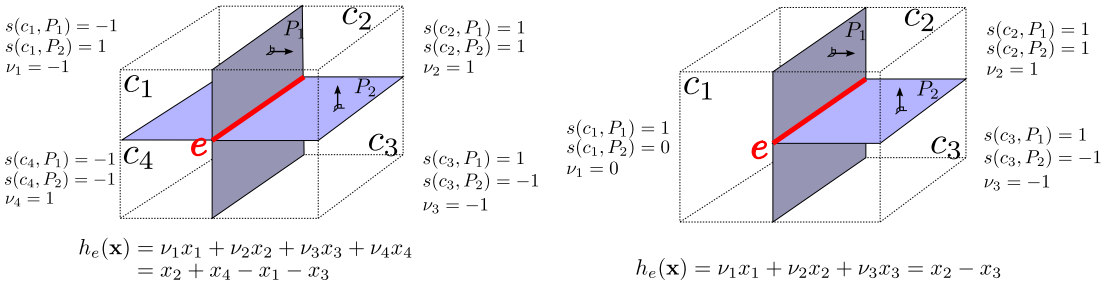
$$E_{aire}(\mathbf{x}) = \sum_{f \in \mathcal{F}} \frac{a_f}{\sigma^2} \, |x_{f^+} - x_{f^-}| \tag{8.7}$$

La pénalité intervient lorsque la facette fait partie de la surface reconstruite, c'est-à-dire lorsque les cellules adjacentes sont de nature différente, l'une pleine l'autre vide. L'énergie est normalisée par le facteur d'échelle σ^2 pour que E_{aire} soit sans dimension.

Cette énergie peut être vue comme une somme de potentiels par paires dans le contexte d'un champ de Markov aléatoire (MRF, Markov Random Field). [Chauve et al., 2010] utilisent aussi ce type de terme. La minimisation de cette énergie peut être résolue de manière optimale et rapidement calculable via une coupe de graphe.

Terme d'arête

$E_{arete}(\mathbf{x})$ pénalise la longueur des arêtes de la surface reconstruite. Une arête $e \in \mathcal{E}$ est un objet de dimension 1, elle correspond au bord d'une facette. Dans la plupart des cas, une arête est adjacente à quatre cellules, mais il arrive qu'elle soit adjacente à seulement trois d'entre elles, lorsqu'il y a limitation d'un plan par un autre (figure 8.13).



Cas d'adjacence à quatre cellules (à gauche) et à trois cellules à droite.

FIGURE 8.13 – Calcul des poids $\nu(c, e)$ et de la fonction $h_e(\mathbf{x})$.

Pour modéliser les deux cas, la fonction s décrit le signe d'une cellule par rapport à un plan donné :

$$s(c, P) = \begin{cases} +1 \text{ si } c \text{ est du côté positif de } P \\ -1 \text{ si } c \text{ est du côté négatif de } P \\ \quad 0 \text{ si } c \text{ est de chaque côté de } P \end{cases} \tag{8.8}$$

Dans le complexe, une arête qui a quatre cellules adjacentes est une arête "matérielle" (une arête pénalisée) de la surface reconstruite si et seulement si :

- une seule des cellules est vide (arête entrante)
- une seule des cellules est pleine (arête saillante)
- chaque cellule est adjacente via une facette uniquement à des cellules d'étiquette opposée

Les configurations sont dessinées sur la figure 8.14. Les possibilités non représentées sont obtenues par inversion des étiquettes volumiques plein/vide. Une arête ayant trois cellules adjacentes est une arête matérielle uniquement si les deux cellules qui sont du même côté du plan ont des étiquettes différentes.

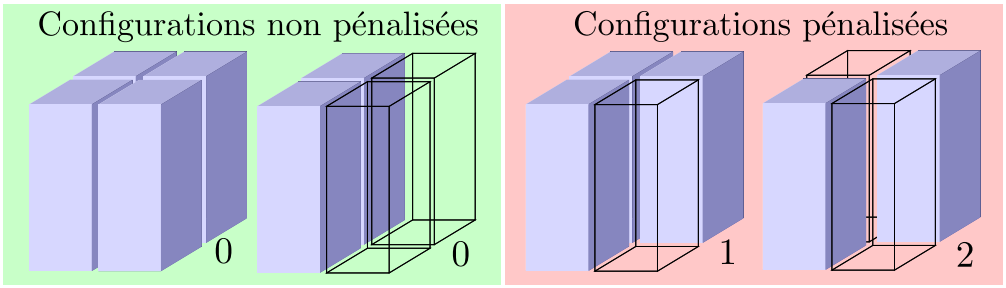
Dans les deux cas, l'existence d'une arête matérielle e s'exprime via :

$$h_e(\mathbf{x}) = \sum_{c \in Adjc(e)} \nu(c, e)\, x_c \quad \text{où} \quad \nu(c, e) = \prod_{P \in AdjP(e)} s(c, P) \tag{8.9}$$

où $Adjc(e)$ est l'ensemble des cellules adjacentes à e et $AdjP(e)$ l'ensemble des plans adjacents à e. Ces ensembles sont accessibles en temps constant, du fait de la mémorisation des graphes d'adjacence (cf section 8.4.1). $\nu(c, e)$ est un signe associé à chaque cellule en fonction des plans adjacents (figure 8.14).

La valeur absolue $|h(e)|$ est nulle lorsqu'il n'y a pas d'arête matérielle et 1 ou 2 lorsque l'arête matérielle existe. $|h(e)|$ est indépendante de l'assignation de l'orientation des plans.

Le cas « quatre-cellules »correspond à un potentiel d'ordre de 4 dans le contexte de MRFs, et dans le cas « trois-cellules »à un potentiel du second ordre (l'une des variables étant toujours nulle du fait d'un facteur nul $\nu(c, e)$).



Les configurations pénalisées sont sur fond rouge, les autres sur fond vert. Les coûts associés sont situés en bas à droite de la figure correspondante. Les configurations manquantes sont obtenues par rotation et inversion des labels.

FIGURE 8.14 – Configurations possibles pour une arête adjacente à quatre cellules, et les coûts associés à chacune d'entre elles.

Nous définissons E_{arete} de la manière suivante :

$$E_{arete}(\mathbf{x}) = \sum_{e \in \mathcal{E}} |h_e(\mathbf{x})|$$

Lemme 1. *Ces potentiels d'ordre 4 ne peuvent pas être régularisés au sens de Kolmogorov and Zabih [2004].*

a	b
c	d

(a) Configuration.

$E($) $+ E($) $- E($) $- E($) > 0

(b) Contre-exemple à la régularité de E.

Démonstration. On considère le cas de l'énergie de la configuration à quatre cellules de la figure 8.15a :

$$E = E(a, b, c, d)$$

E est régulière au sens de Kolmogorov and Zabih [2004] si toutes les projections à deux variables sont régulières ([Kolmogorov and Zabih, 2004], Définition 5.1). Autrement dit, en fixant deux variables parmi a, b, c et d, l'énergie $\tilde{E}$ des deux variables restantes doit être régulière :

$$\tilde{E}(0,0) + \tilde{E}(1,1) - \tilde{E}(0,1) - \tilde{E}(1,0) \leq 0$$

Pour montrer que E n'est pas régulière, il suffit de trouver une projection violant cette régularité. Fixant $b = 0$ et $c = 1$:

$$\tilde{E}(a = 0, d = 0) + \tilde{E}(1,1) - \tilde{E}(0,1) - \tilde{E}(1,0) = 1 + 1 - 0 - 0 = 2 > 0$$

Ce contre-exemple est illustré visuellement sur la figure 8.15b. □

En pratique les arêtes sont pénalisées par :

$$E_{arete}(\mathbf{x}) = \sum_{e \in \mathcal{E}} w_e |h_e(\mathbf{x})| \tag{8.10}$$

où w_e est un poids associé à e. w_e est proportionnel à la longueur de l'arête, l_e, pour rendre la longueur totale d'arêtes de la scène indépendante de l'ajout d'un plan supplémentaire dans l'arrangement. Par ailleurs la longueur l_e est mesurée en nombre de σ pour en faire une quantité sans dimension.

Il est aussi possible de pénaliser l'angle α_e entre les plans constituant e, en fonction d'une distribution souhaitée. Par exemple, dans un environnement créé par la main humaine, il est envisageable de favoriser les angles droits. On considère deux paramètres : A, le coût d'un angle très différent de 90°, et ρ l'écart type de l'angle à pénaliser par rapport à l'angle droit. Le poids angulaire est alors :

$$w_{ang}(\alpha) = A + (1 - A)\exp(-\frac{(\alpha - \pi/2)^2}{2\rho^2}) \tag{8.11}$$

Le poids $w_{ang}(\alpha)$ est égal à 1 si l'angle est droit et est très proche de A dès que la déviation est plus grande que ρ.

Enfin w_e est défini par :

$$w_e = \frac{l_e}{\sigma}\, w_{ang}(\alpha_e) \tag{8.12}$$

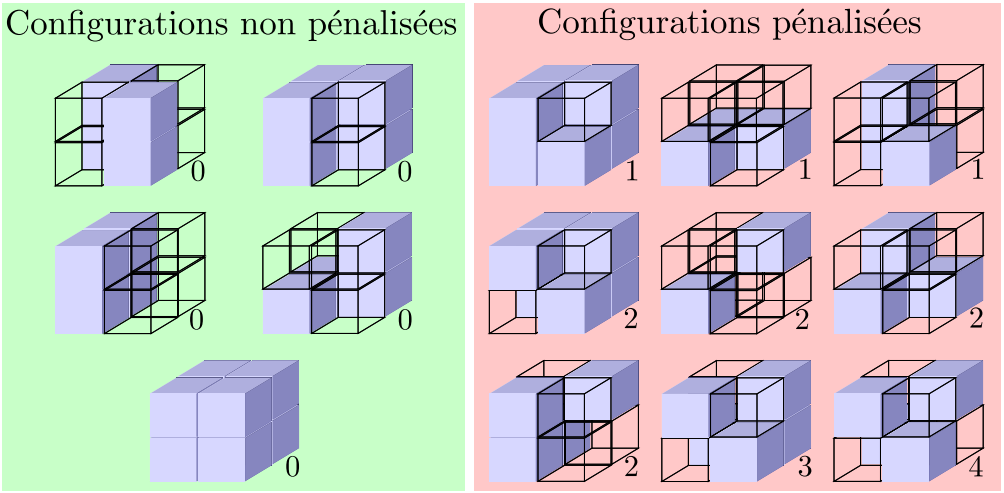
Terme de coin

$E_{coin}(\mathbf{x})$ pénalise le nombre de coins de la surface finale. Un sommet $v \in \mathcal{V}$ dans le complexe est le point d'intersection de trois plans. Il est en général adjacent à huit cellules, mais peut avoir six ou quatre voisins du fait de la limitation des plans fantômes.

Avec une définition similaire à celle de l'arête matérielle, l'existence d'un coin matériel est exprimée par :

$$h_v(\mathbf{x}) = \sum_{c \in Adjc(v)} \nu(c, v)\, x_c \quad \text{où} \quad \nu(c, v) = \prod_{P \in AdjP(v)} s(c, P) \tag{8.13}$$

où $\nu(c, v)$ est un signe utilisé comme poids linéaire associé à chaque cellule. $|h_v(\mathbf{x})|$ est égal à 0 lorsqu'il n'y a pas de coin matériel, à 1 pour un coin simple et jusqu'à 4 dans les cas plus compliqués (figure 8.16).



Les configurations pénalisées sont sur fond rouge, les autres sur fond vert. Les coûts associés sont situés en bas à droite de la figure correspondante. Les configurations manquantes sont obtenues par rotation et inversion des labels.

FIGURE 8.16 – Configurations possibles pour un coin adjacent à huit cellules, et les coûts associés à chacune d'entre elles.

Il faut noter que le point selle (le second cas de la deuxième ligne des cas non pénalisés) n'est pas pénalisé. D'un point de vue de la complexité, cela se justifie. Par exemple,

placer deux parallélépipèdes en contact (le bord d'une table contre un mur) ne doit pas être considéré comme plus compliqué que s'ils ne sont pas en contact. Ainsi le point selle que ce contact peut créer ne doit pas être pénalisé (figure 8.17).

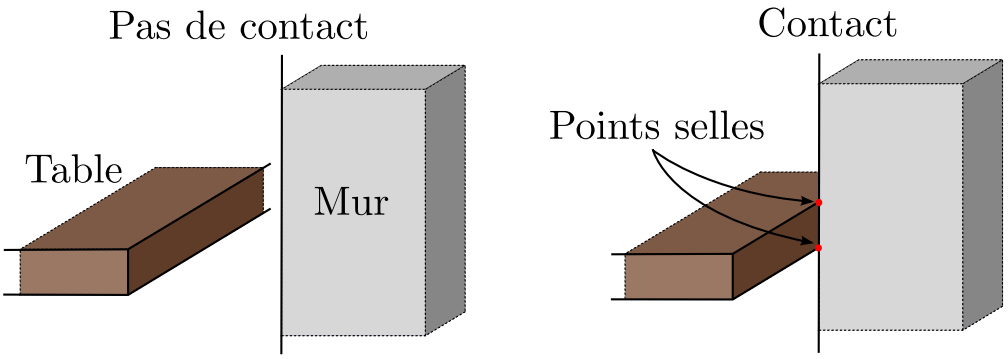


FIGURE 8.17 – Présence ou non de point selle. À gauche, aucun contact entre la table et le mur et à droite, la table est collée au mur, créant ainsi deux points selles.

Nous définissons E_{coin} de la manière suivante :

$$E_{coin}(\mathbf{x}) = \sum_{v \in \mathcal{V}} |h_v(\mathbf{x})|$$

Comme pour les arêtes, les $|h_v(\mathbf{x})|$ ne dépendent pas des conventions d'orientation prises pour les plans. Ils correspondent à des potentiels allant jusqu'à l'ordre 8 (cas sans limitation) dans le contexte des MRFs. Les coins sont en fait pénalisés avec :

$$E_{coin}(\mathbf{x}) = \sum_{v \in \mathcal{V}} w_v |h_v(\mathbf{x})| \tag{8.14}$$

où w_v est un poids associé à l'angle. En effet, il est possible de décourager les coins formés par des plans qui ne sont pas deux à deux orthogonaux. Les trois angles entre les plans sont notés $(\alpha_{v,i})_{i \in \{1,2,3\}}$. Comme dans le cas des arêtes, la pénalité est fonction de deux paramètres A et ρ :

$$w_{ang}(\alpha_1, \alpha_2, \alpha_3) = A + (1 - A) \exp\left(-\frac{\sum_{i \in \{1,2,3\}} (\alpha_i - \pi/2)^2}{2\rho^2}\right) \tag{8.15}$$

qui est indépendant de l'ordre donné aux angles et aux plans. Finalement :

$$w_v = w_{ang}(\alpha_1, \alpha_2, \alpha_3) \tag{8.16}$$

Comme E_{aire} et E_{arete}, E_{coin} est une quantité sans dimension.

8.5.5 Résolution

La surface est estimée par minimisation de l'énergie $E(\mathbf{x})$. Il est possible de le faire avec une méthode basée sur la formulation MRF, avec des potentiels d'ordres supérieurs (jusqu'à 8). Cependant, ces potentiels sont connus pour être difficiles à résoudre pour la plupart des méthodes d'inférence sur les MRFs.

Pour les potentiels d'ordre 2, des méthodes efficaces de résolution telles que les coupes de graphe peuvent être utilisées. Cependant, cela restreint l'utilisation au terme d'attache aux données et à la régularisation sur l'aire, comme pour [Chauve et al., 2010].

Nous avons expérimenté le cas général (avec des potentiels d'ordre jusqu'à 8), grâce à des méthodes de résolutions de la librairie OpenGM [Andres et al., 2012] telles que

« Tree Reweighted Belief Propagation »(TRBP) et « Lazy Flipper ». Ces expériences ont produit de mauvais résultats, tant en terme de temps de calcul qu'en terme de valeur finale de l'énergie et de la qualité de la surface reconstruite.

Optimisation linéaire mixte en nombres entiers

Plutôt que d'utiliser une formulation sous forme de MRF, le problème est exprimé comme un programme linéaire. L'idée principale est de reformuler chaque valeur absolue $|a|$ de l'énergie comme un programme linéaire via l'utilisation d'une variable continue b :

$$|a| = \min_b b \quad s.c. \quad -b \leq a \leq b \tag{8.17}$$

Basé sur cette idée, un ensemble de variables continues $\mathbf{y}$ est introduit, une variable pour chaque facette, arête et sommet de l'arrangement, ainsi qu'un ensemble de contraintes sur ces variables $\mathscr{C}$:

$$(\mathbf{x},\mathbf{y}) \in \mathscr{C} \Leftrightarrow \begin{cases} \forall f \in \mathcal{F}, & -y_f \leq h_f(\mathbf{x}) \leq y_f \\ \forall e \in \mathcal{E}, & -y_e \leq h_e(\mathbf{x}) \leq y_e \\ \forall v \in \mathcal{V}, & -y_v \leq h_v(\mathbf{x}) \leq y_v \end{cases} \tag{8.18}$$

Il est possible de reformuler la minimisation de $E(\mathbf{x})$ comme la minimisation d'une fonction linéaire sur les deux ensembles de variables $\mathbf{x}$ et $\mathbf{y}$. Le problème devient linéaire mixte en nombres entiers. L'énergie à minimiser prend la forme suivante :

$$E'(\mathbf{x},\mathbf{y}) = \zeta + \sum_{c\in\mathcal{C}} \tau_c x_c + \sum_{f\in\mathcal{F}} \tau_f y_f + \sum_{e\in\mathcal{E}} \tau_e y_e + \sum_{v\in\mathcal{V}} \tau_v y_v \tag{8.19}$$

et

$$\min_{\mathbf{x}} E(\mathbf{x}) = \min_{(\mathbf{x},\mathbf{y})} E((\mathbf{x},\mathbf{y})) \quad s.c. \quad \begin{cases} \forall c \in \mathcal{C}, & -x_c \in \{0,1\} \\ (\mathbf{x},\mathbf{y}) \in \mathscr{C} \end{cases} \tag{8.20}$$

Optimisation par utilisation d'une relaxation linéaire

Résoudre un programme en nombres entiers est NP-difficile en général. Une approche classique est de relâcher le problème pour que toutes les variables soient réelles, moyennant des contraintes sur leur valeur. Ici, les x_c sont autorisées à prendre une valeur dans $[0,1]$.

Grâce à cette relaxation, le problème devient linéaire vis-à-vis de $\mathbf{x}$ et peut être résolu efficacement par des solveurs utilisés comme boîte noire :

$$\min_{\mathbf{x}} E(\mathbf{x}) = \min_{(\mathbf{x},\mathbf{y})} E((\mathbf{x},\mathbf{y})) \quad s.c. \quad \begin{cases} \forall c \in \mathcal{C}, & -x_c \in [0,1] \\ (\mathbf{x},\mathbf{y}) \in \mathscr{C} \end{cases} \tag{8.21}$$

Pour les expériences, Mosek© a été utilisé avec succès.

Dans le cas où seule la pénalisation sur l'aire est utilisée, le problème entier est un problème de coupe minimale pouvant être résolu par coupe de graphe. La solution du programme linéaire et la solution du programme entier sont alors confondues. (La matrice définissant le programme linéaire est totalement unimodulaire, le théorème de Hoffman and Kruskal [1956] s'applique.)

Dans le cas des potentiels d'ordres supérieurs à deux, ce qui est le cas pour les pénalisations sur la longueur des arêtes et le nombre de coins, les x_c peuvent prendre après résolution des valeurs fractionnaires et doivent être arrondis. Cet arrondi introduit un biais vis-à-vis de la solution optimale du problème linéaire, et rien ne garantit que la solution entière obtenue est proche de l'optimal du problème entier. La stratégie d'arrondi

utilisée ici est la plus simple, les x_c sont arrondis à l'entier le plus proche sans tenir compte du voisinage de la cellule.

Pour estimer la déviation à l'optimal après arrondi, le problème est résolu à nouveau en forçant les x_c à leur valeur entière pour estimer la fonction objectif. Expérimentalement, l'augmentation due à l'arrondi reste faible, avec une déviation maximale de l'ordre de 8% pour la pénalisation sur les coins uniquement, et de l'ordre de 6% en pénalisant la longueur des arêtes. On peut donc penser que les solutions entières obtenues par arrondi sont presque aussi bonnes que l'optimum.

Pour toutes les expériences, la méthode de résolution utilisée est celle du dual simplexe de Mosek©. La méthode « branch & bound & cut »du même logiciel a aussi été testée, mais l'amélioration est faible, pour un temps de calcul plus important.

8.6 Résultats

La méthode est évaluée sur des scènes d'intérieur acquises par laser. Les acquisitions ont été décimées (les tailles des nuages sont disponibles dans la section 8.6.2).

Elle est comparée à l'algorithme général de reconstruction planaire par morceaux de [Chauve et al., 2010] à la même échelle σ.

La comparaison n'a de sens qu'avec des méthodes planaires par morceaux et essayant de reconstruire dans les parties cachées. C'est pourquoi les reconstructions de Poisson ne sont données qu'à titre indicatif. Les méthodes basées sur l'hypothèse de monde Manhattan ne sont pas non plus utilisées car elles présentent des artefacts évidents.

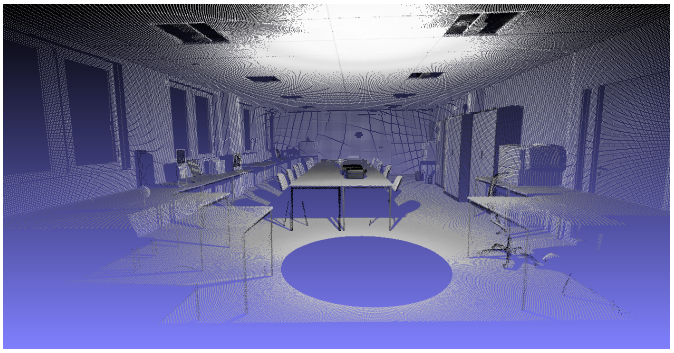
Pour toutes les expériences, les même paramètres ont été utilisés. Premièrement, σ est fixé à 10 cm. Ensuite, lorsque les régularisations sont utilisées séparément, les poids relatifs à E_{data} sont les suivants : $\lambda_{aire} = 10^{-4}$, $\lambda_{arete} = 10^{-3}$ et $\lambda_{coin} = 10^{-2}$. Pour l'utilisation conjointe de la longueur des arêtes et du nombre de coins, λ_{arete} est fixé à 5×10^{-2} et λ_{coin} à 10^{-2}. L'idée, derrière ces poids, est de favoriser en priorité une réduction du nombre de coins, mais pour deux solutions, l'une et l'autre avec le même nombre de coins, c'est la solution qui a la longueur totale d'arêtes la plus faible qui sera préférée. Un coin est alors échangeable contre une réduction de $\sigma\lambda_{coin}/\lambda_{arete} = 2$ m d'arêtes. Ce choix de paramètres pour la régularisation se rapproche d'un ordre lexicographique. Par ailleurs, les coefficients assignés aux régularisations sont négligeables vis-à-vis du terme d'attache aux données. Cela signifie que la régularisation influence peu ou pas la surface reconstruite dans la partie visible ou fait confiance aux observations. Seule la partie invisible est contrôlée par la régularisation.

8.6.1 Visuels

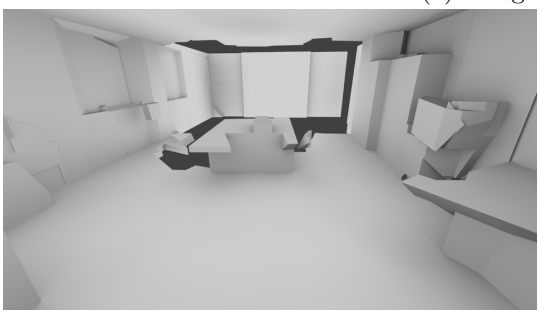
La figure 8.18 montre les reconstructions pour différents modèles et pour la méthode de [Chauve et al., 2010], et avec notre méthode pour les trois termes de régularisation utilisés séparément. On observe sur l'image correspondant à [Chauve et al., 2010] (8.18b), que les primitives distantes du point d'acquisition ne sont pas correctement reconstruites, du fait de l'anisotropie de l'acquisition. Modifier la valeur de σ dégrade les résultats, un σ plus petit augmente l'effet de l'anisotropie, et une échelle plus grande cause une perte de détail. Ce phénomène est présenté figure 8.19 où la scène est reconstruite pour différentes valeurs de σ.

Sur les images 8.18c, 8.18d et 8.18e la reconstruction est robuste à l'anisotropie. Cependant la pénalisation de la surface (8.18c) introduit des artefacts non désirés comme des trous dans les parties invisibles (sous la table ou sous le laser) car ils permettent de

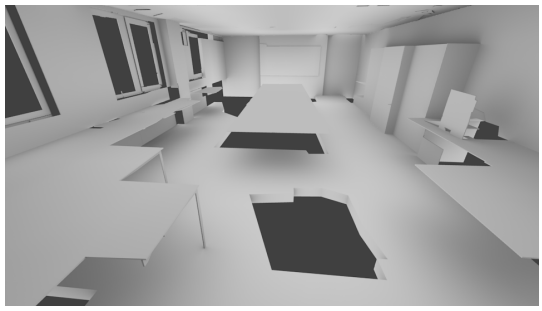
réduire la surface totale. Les régularisations sur les arêtes (8.18d) et sur les coins (8.18e) produisent de bien meilleurs résultats, les trous disparaissent car ils sont constitués de coins et d'arêtes inutiles.



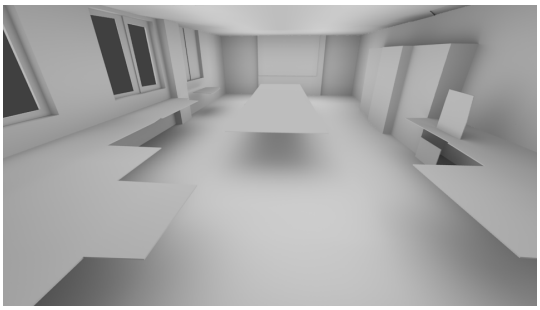
(a) Nuage de points.



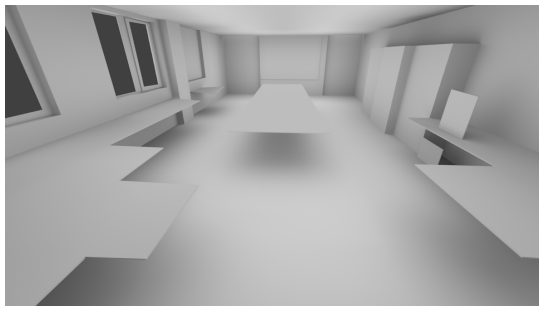
(b) [Chauve et al., 2010].



(c) Aire.



(d) Longueur des arêtes.



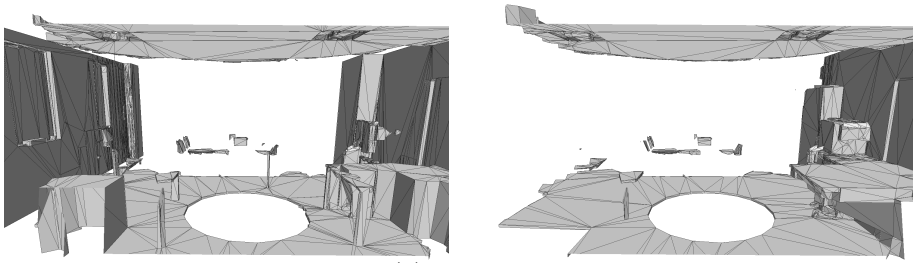
(e) Nombre de coins.

FIGURE 8.18 – Reconstructions pour la salle de réunion 1 (a) avec la méthode de [Chauve et al., 2010] (b), et avec notre méthode en utilisant les pénalisations sur l'aire (c), la longueur des arêtes (d) et le nombre de coins (e).

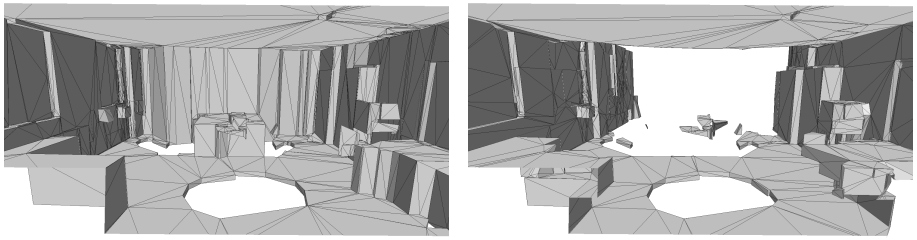
Les pénalisations sur la longueur des arêtes et sur le nombre de coins donnent des reconstructions proches. La différence se joue principalement au niveau des détails. Sur la figure 8.20, les coins de la table sont coupés pour réduire la longueur des arêtes, mais créent des coins supplémentaires. Cet artefact disparaît avec l'objectif de réduire le nombre de coins (à droite).

Il est aussi possible d'utiliser les régularisations conjointement. La figure 8.21 fournit un exemple pour lequel il est utile de limiter la longueur des arêtes et le nombre de coins.

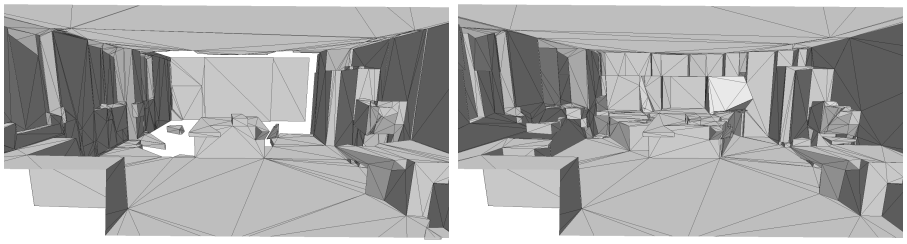
Les figures 8.22 et 8.23 montrent d'autres reconstructions. Malgré quelques erreurs dues à la présence d'objets, les surfaces principales sont bonnes, même pour les éléments



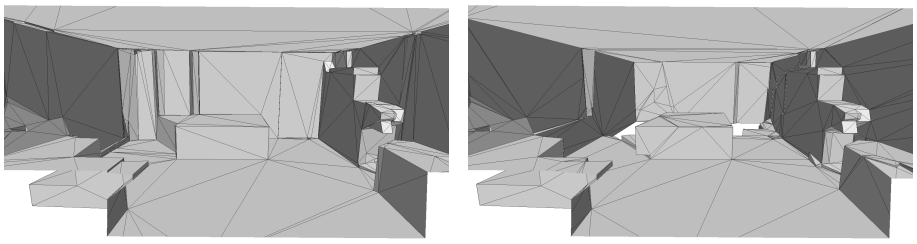
(a) $\sigma = 0.02$



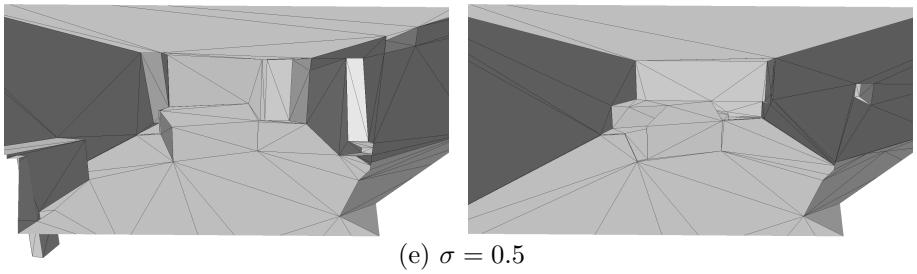
(b) $\sigma = 0.05$



(c) $\sigma = 0.1$



(d) $\sigma = 0.2$



(e) $\sigma = 0.5$

FIGURE 8.19 – Évolution de la reconstruction avec la méthode de Chauve et al. [2010] en fonction de l'échelle de reconstruction σ. Insertion prioritaire des plans verticaux (colonne de gauche) et horizontaux (colonne de droite).

fins tels que les tables et les cadres de fenêtres.

Il faut aussi noter que les acquisitions laser ont été effectuées d'un point de vue différent de celui des images de reconstruction représentées. Par exemple sur la figure 8.22, il n'y a pas de points sous la table. Sur la figure 8.23, il n'y pas de données sur le dessus des

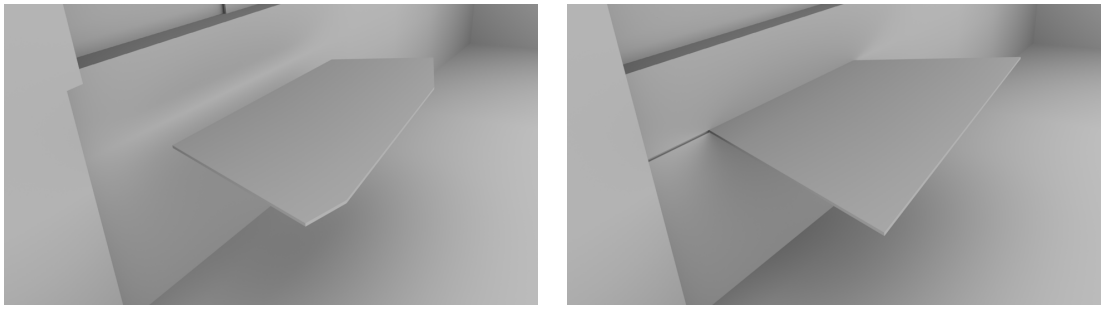


FIGURE 8.20 – Régularisation sur les arêtes (à gauche) et sur les coins (à droite).

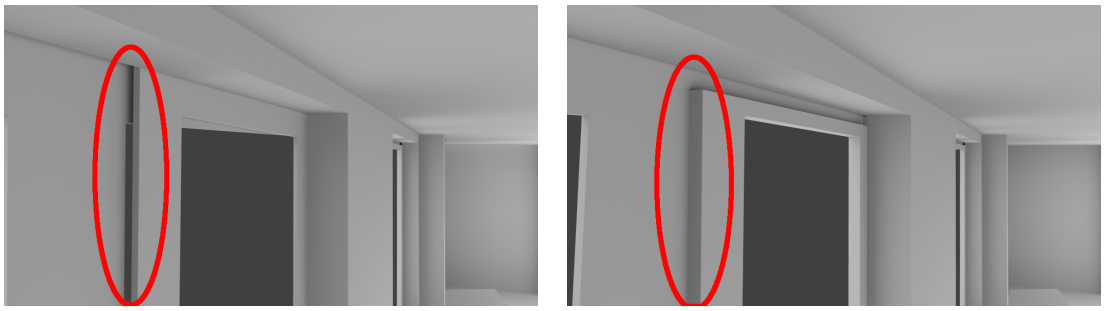


FIGURE 8.21 – Régularisation sur les coins (à gauche), et conjointement arêtes et coins (à droite).

marches dès que celles-ci sont plus hautes que le point d'acquisition du laser. Cependant l'escalier est correctement reconstruit.

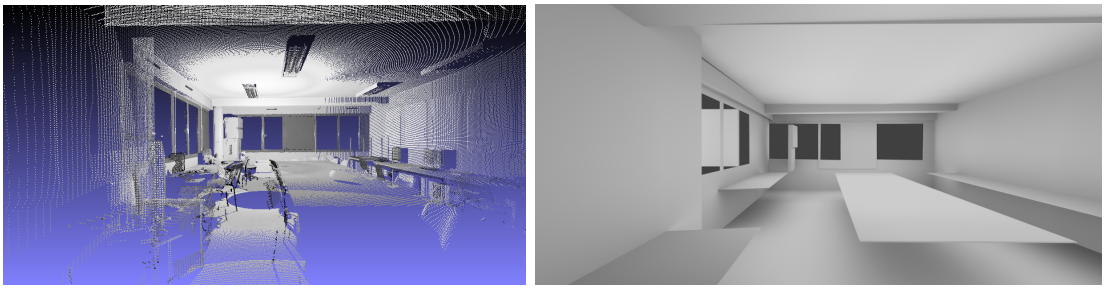


FIGURE 8.22 – Salle de réunion 2 avec pénalisation sur les coins.

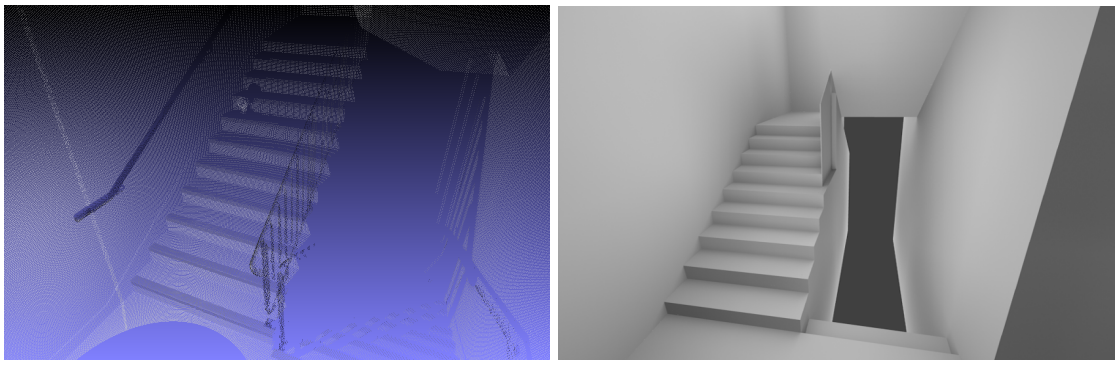


FIGURE 8.23 – Escaliers : nuage de points (à gauche) et surface reconstruite avec pénalisation sur les coins (à droite).

Deux reconstructions de scène extérieure sont proposées figure 8.24. Deux nuages de points laser du même bâtiment ont été acquis depuis le sol. Les nuages de points n'ont pas été fusionnés, nous avons produit une reconstruction pour chacun des nuages (figures 8.24a et 8.24b). On observe que le bâtiment est correctement reconstruit à l'exception de certaines zones décrites ci-dessous. Pour la plupart des fenêtres, les fantômes orthogonaux et parallèles sont générés et utilisés, il en résulte des ouvertures de bonne qualité et même dans certains cas la création d'un intérieur de pièce derrière l'ouverture (figure 8.24b, à gauche).

Les principaux artefacts observables sont d'une part une ouverture verticale allongée (figure 8.24a, à droite) et des fenêtres non dessinées (figure 8.24a, à gauche et figure 8.24b, à droite).

L'ouverture allongée est due à la présence d'un lampadaire lors de l'acquisition. Celui-ci a été retiré lors de la sélection des points correspondant au bâtiment à reconstruire. Cette étape de nettoyage du nuage n'est pas de notre fait, nous avons utilisé les données telles que fournies. Dans notre implémentation actuelle, les points manquants dans l'image laser sont considérés comme étant à l'infini, ce qui explique la création d'une ouverture à cet endroit.

Les bords inexistants sur certaines fenêtres sont en fait cachés par l'avancée du bâtiment au centre de la façade. Or, pour générer un plan fantôme, il faut avoir détecté une arête. Ce cas de figure traduit un nouveau besoin : la détection de régularité dans le modèle. La présence de fenêtres proches et visibles peut être une source d'informations pour la reconstruction des ouvertures partiellement visibles.

8.6.2 Temps de calcul

La taille des nuages de points traités et les temps d'exécution sont présentés dans la table 8.1. Les surfaces sont reconstruites en quelques minutes sur un ordinateur muni de 2 processeurs Intel(R) Xeon(R) X5472 3.00GHz. Bien que le système puisse gérer plusieurs millions de points, il est plus pratique de réduire la taille des images laser (d'un facteur de l'ordre de 10 à 25). En effet, la qualité des reconstructions varie peu pour un temps de calcul moindre (pour le σ choisi). Les opérations concernant les points individuellement (l'estimation de normales, l'assignation des points aux facettes ou les tests de visibilité) sont rapides mais doivent être effectuées pour chaque point. Ainsi, un nuage de taille raisonnable améliore le temps de calcul.

De plus, le goulot d'étranglement de la méthode est la construction de l'arrangement. L'insertion d'un plan supplémentaire et plus coûteuse en temps et en espace que l'insertion du précédent, ce qui impose de limiter le nombre de plans à insérer. A ceci s'ajoute le fait que les tests de visibilité sont aussi influencés par la complexité de l'arrangement (dans la table 8.1 le temps des tests de visibilité est inclus dans le temps de calcul de l'arrangement).

Le nombre de variables dans l'optimisation est lui aussi indépendant de la taille du nuage et ne dépend que de la structure de l'arrangement.

Scène	#points	Normales	Détection		Fusion		Fantômes		Arrang.	Optimisation		Total
Salle réunion 1 st. 1	658 k	18 s	0 s	72 prim.	2 s	39 prim.	27 s	344 plans	54 s	coin	52 s	153 s
Salle réunion 1 st. 2	642 k	24 s	1 s	48 prim.	1 s	27 prim.	23 s	211 plans	57 s	coin	50 s	156 s
Salle réunion 1 st. 3	663 k	22 s	1 s	52 prim.	1 s	26 prim.	23 s	276 plans	32 s	coin	16 s	95 s
Salle réunion 1 st. 4	667 k	23 s	2 s	56 prim.	1 s	26 prim.	24 s	230 plans	52 s	coin	42 s	144 s
Salle réunion 2	1054 k	49 s	2 s	36 prim.	1 s	21 prim.	34 s	232 plans	64 s	coin	18 s	168 s
Escaliers	680 k	18 s	0 s	51 prim.	1 s	40 prim.	26 s	160 plans	47 s	coin	45 s	137 s

TABLE 8.1 – Temps de calcul des différentes étapes de reconstruction pour différentes scènes.

8.7 Conclusion et perspectives

Ce chapitre a présenté une méthode efficace pour la reconstruction de surfaces planaires par morceaux. L'algorithme complète les parties manquantes de la surface en utilisant des a priori ordinaires d'orthogonalité et de simplicité pour des scènes d'intérieur créées par l'homme.

En comparaison de [Chauve et al., 2010], l'approche présentée n'a pas les artefacts dus à l'utilisation d'une grille de voxels. De plus, elle gère la présence d'anisotropie dans les données ainsi que des objets fins dans la scène.

Les expériences montrent que les régularisations sur le nombre de coins et sur la longueur des arêtes matérielles sont supérieures à la régularisation basée sur la minimisation de l'aire dans les parties invisibles. La formulation du problème pour ces régularisations est résolue efficacement via une relaxation linéaire pour atteindre un résultat proche de l'optimum.

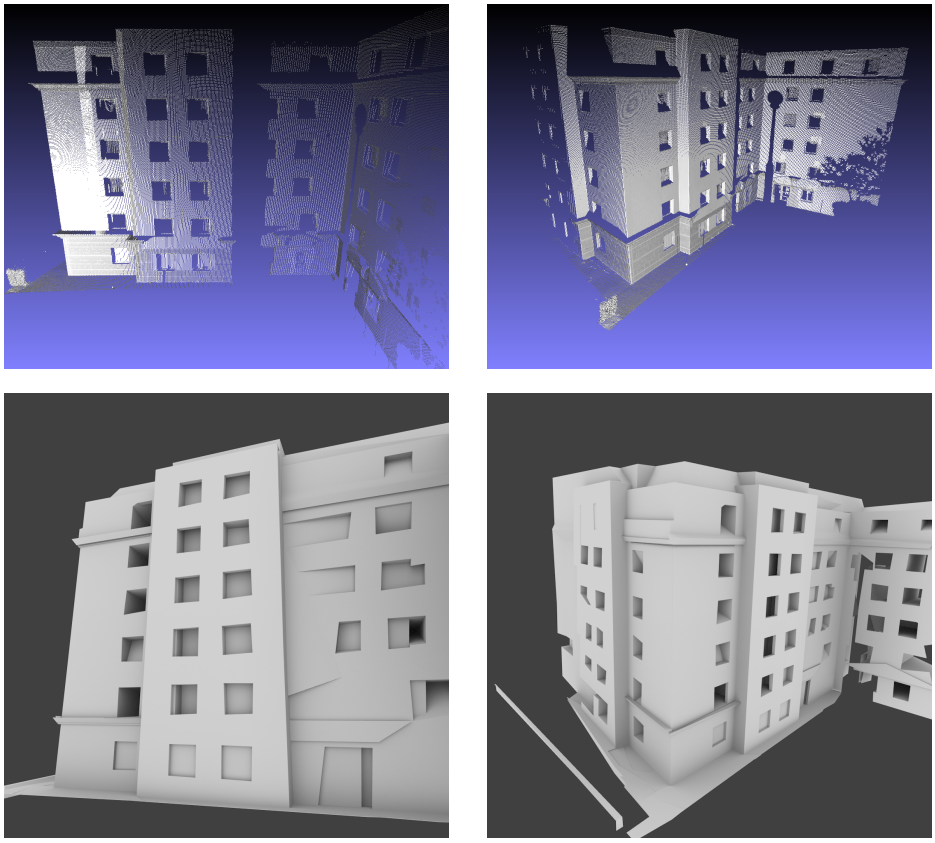
Il faut aussi noter que ces régularisations et leur optimisation ne se limitent pas aux images de profondeur acquises par laser, elles peuvent être utilisées dans d'autres contextes, comme par exemple, en photogrammétrie [Chauve et al., 2010]. Seul notre pré-traitement concernant les bords des primitives nous contraint à des expériences sur un scan laser plutôt qu'un nuage de points issu de photogrammétrie. (Notre implémentation actuelle ne gère pas non plus de multiples points de vue.)

Bien que la méthode soit capable de gérer des nuages de plusieurs millions de points, elle n'est pas adaptée au traitement des centaines de scans nécessaires à la reconstruction d'un bâtiment complet. Le goulot d'étranglement se situe au niveau de l'arrangement de plans qui fonctionne pour quelques centaines de plans, mais ne passe pas à l'échelle de plusieurs milliers. Le défi est maintenant d'être en mesure de combiner les reconstructions partielles de chaque point de vue, tout en garantissant la consistance géométrique et en restant proche de l'optimum global.

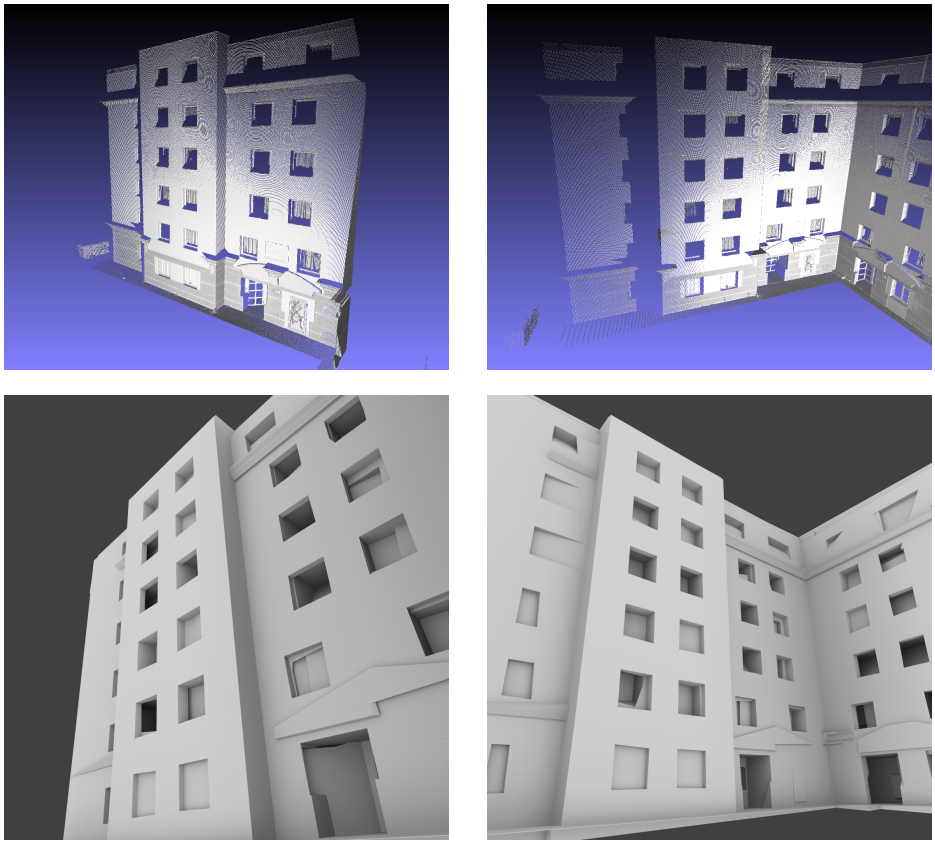
Publication

Ce travail a été présenté au Symposium On Geometry Processing 2014, à Cardiff, Pays de Galles. Il a été publié dans la revue Computer Graphics Forum :

Alexandre Boulch, Martin de La Gorce, Renaud Marlet.
Piecewise-Planar 3D Reconstruction with Edge and Corner Regularization.
Comput. Graph. Forum 33(5): 55-64 (2014)



(a) Station 1.



(b) Station 2.

FIGURE 8.24 – Reconstruction de façade, à partir de 2 nuages laser différents.

Bibliographie

Andres, B., Beier, T., and Kappes, J. H. (2012). OpenGM: A C++ library for discrete graphical models. *CoRR*, abs/1206.0111.

Arikan, M., Schwärzler, M., Flöry, S., Wimmer, M., and Maierhofer, S. (2013). O-Snap: Optimization-Based Snapping for Modeling Architecture. *ACM Transactions on Graphics*, 32(1):1–15.

Avron, H., Sharf, A., Greif, C., and Cohen-Or, D. (2010). l_1-sparse reconstruction of sharp point set surfaces. *ACM Transactions on Graphics*, 29(5):135.

Berger, M., Tagliasacchi, A., Seversky, L. M., Alliez, P., Levine, J. A., Sharf, A., and Silva, C. (2014). State of the art in surface reconstruction from point clouds. *Eurographics STAR (Proc. of EG'14)*.

Budroni, A. and Böhm, J. (2010). Automatic 3D modelling of indoor Manhattan-world scenes from laser data. In *ISPRS Symp. Close Range Image Measurement Techniques*.

Castellani, U. (2002). Improving environment modelling by edge occlusion surface completion. In *Proceedings of 3DPVT*, pages 1–4.

Chauve, A.-L., Labatut, P., and Pons, J.-P. (2010). Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *CVPR*, pages 1261–1268.

Chazal, F., Cohen-Steiner, D., and Mérigot, Q. (2011). Geometric inference for probability measures. *Foundations of Computational Mathematics*, 11(6):733–751.

Chen, J. and Chen, B. (2008). Architectural modeling from sparsely scanned range data. *International Journal of Computer Vision*, pages 1–22.

Dey, T. K. and Goswami, S. (2006). Provable surface reconstruction from noisy samples. *Computational Geometry*, 35(1-2):124–141.

Edelsbrunner, H., O'Rourke, J., and Seidel, R. (1986). Constructing arrangements of lines and hyperplanes with applications. *SIAM Journal on Computing*, 15(2):341–363.

Fayolle, P.-A. and Pasko, A. (2013). Segmentation of discrete point clouds using an extensible set of templates. *Visual Computer*, 29(5):449–465.

Furukawa, Y., Curless, B., Seitz, S., and Szeliski, R. (2009a). Manhattan-world stereo. In *CVPR*, pages 1422–1429. Ieee.

Furukawa, Y., Curless, B., Seitz, S., and Szeliski, R. (2009b). Reconstructing building interiors from images. In *ICCV*, pages 80–87.

Giraudot, S., Cohen-Steiner, D., and Alliez, P. (2013). Noise-adaptive shape reconstruction from raw point sets. *Comput. Graph. Forum*, 32(5):229–238.

Hoffman, A. and Kruskal, J. (1956). Integral boundary points of convex polyhedra. *Kuhn, H.W. ; Tucker, A.W., Linear Inequalities and Related Systems, Annals of Mathematics Studies*, 38:223–246.

Jenke, P., Krückeberg, B., and Straßer, W. (2008). Surface reconstruction from fitted shape primitives. In *Proceedings of the Vision, Modeling, and Visualization Conference*.

Kazhdan, M. M., Bolitho, M., and Hoppe, H. (2006). Poisson surface reconstruction. In *Symposium on Geometry Processing*, pages 61–70.

Kolmogorov, V. and Zabih, R. (2004). What energy functions can be minimized via graph cuts ? *PAMI*, 26:147–159.

Lafarge, F. and Alliez, P. (2013). Surface reconstruction through point set structuring. *Comput. Graph. Forum*.

Li, Y., Wu, X., Chrysanthou, Y., Sharf, A., Cohen-Or, D., and Mitra, N. J. (2011). GlobFit: consistently fitting primitives by discovering global relations. *ACM Transactions on Graphics*.

Mellado, N., Guennebaud, G., Barla, P., Reuter, P., and Schlick, C. (2012). Growing least squares for the analysis of manifolds in scale-space. *Comput. Graph. Forum*, 31(5):1691–1701.

Schnabel, R., Degener, P., and Klein, R. (2009). Completion and reconstruction with primitive shapes. *Comput. Graph. Forum*, 28:503–512.

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient RANSAC for point-cloud shape detection. *Comput. Graph. Forum*.

Schoenemann, T., Kahl, F., and Cremers, D. (2009). Curvature regularity for region-based image segmentation and inpainting: A linear programming relaxation. In *ICCV*.

Shalom, S., Shamir, A., Zhang, H., and Cohen-Or, D. (2010). Cone carving for surface reconstruction. *ACM Transactions on Graphics*, 29(6):150.

Shekhovtsov, A., Kohli, P., and Rother, C. (2012). Curvature prior for MRF-based segmentation and shape inpainting. In *DAGM/OAGM Symposium*.

Song, Y. (2010). Boundary fitting for 2D curve reconstruction. *The Visual Computer*, 26(3):187–204.

Sotoodeh, S. (2006). Outlier detection in laser scanner point clouds. In *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences XXXVI-5*, pages 297–302.

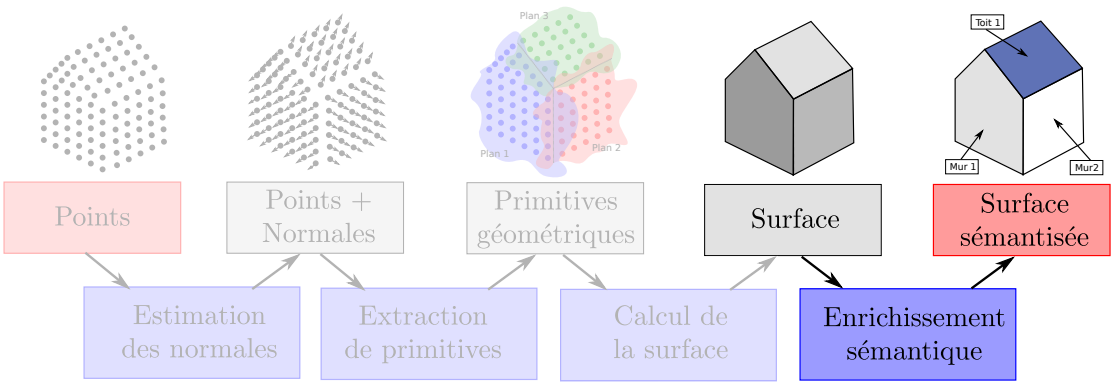
Unnikrishnan, R., Lalonde, J.-F., Vandapel, N., and Hebert, M. (2010). Scale selection for geometric fitting in noisy point clouds. *Int. J. Comput. Geometry Appl.*, 20(5):543–575.

Vanegas, C. A., Aliaga, D. G., and Benes, B. (2012). Automatic extraction of Manhattan-world building masses from 3D laser range scans. *IEEE TVCG*.

Wang, H., Scheidegger, C. E., and Silva, C. T. (2009). Bandwidth selection and reconstruction quality in point-based surfaces. *IEEE Transactions in Visualization and Computer Graphics*, 15(4):572–582.

Quatrième partie

Sémantisation de modèles géométriques à l'aide de grammaires



Chapitre 9

Sémantisation de modèles géométriques

Sommaire

9.1 Introduction

Ce chapitre est consacré à l'enrichissement de modèles géométriques. L'objectif est d'ajouter des informations sémantiques aux polygones qui forment la surface du modèle.

Avec les avancées dans le domaine de la photogrammétrie et le récent développement de moyens d'acquisitions bon marché de données trois dimensions, tel que le Kinect [site] ou dans une moindre mesure les lasers, la production de données 3D s'est démocratisée. Les modèles géométriques résultant de ces acquisitions sont devenus nombreux et très variés. Les algorithmes disponibles pour reconstruire des surfaces sont nombreux (cf partie III). À ces modèles s'ajoutent des géométries purement synthétiques. Il est désormais accessible et aisé de construire des modèles 3D, grâce notamment à des logiciels de conception assistée par ordinateur (CAO) comme SketchUp [site] ou FreeCAD [site].

La compréhension de modèles 3D est un problème ancien en géométrie, toujours considéré comme difficile. L'ajout d'informations sémantiques sur les géométries nues s'avère utile pour des applications variées.

Dans de larges dépôts de modèles comme aim@shape [2006] ou 3D-Warehouse [site], les informations sémantiques pourraient être utiles pour augmenter les performances et la pertinence des recherches.

Parmi les applications de la sémantisation de géométrie, on compte aussi le jeu vidéo, le design d'intérieur, le design de produit pour la fabrication [Attene et al., 2009], etc. L'annotation sémantique manuelle est longue, fastidieuse et sujette à erreur. De plus, la taille de certaines scènes ou dépôts est trop grande pour envisager des actions manuelles, l'automatisation de la tâche en ce cas est requise.

L'industrie de la construction utilise de plus en plus de maquettes numériques sémantisées durant tout le cycle de vie du bâtiment, depuis les esquisses et la conception jusqu'à la construction mais aussi pendant l'exploitation et le démantèlement. La sémantique permet un travail plus rapide et moins coûteux. Si des bâtiments nouveaux sont maintenant construits à l'aide d'une maquette numérique créée au préalable, pour les bâtiments plus anciens, les maquettes sont inexistantes. Avec le développement du marché de la rénovation, le besoin de construction de modèles numériques basés sur l'existant est de plus en plus pressant. Les maquettes de bâtiments existants, quand elles sont produites, sont reconstruites sur la base d'acquisitions laser du bâtiment. Le processus de reconstruction est principalement manuel, long, fastidieux et coûteux.

Le challenge est donc de reconstruire des modèles 3D de bâtiments, contenant à la fois la géométrie et les informations sémantiques, comme les sols, les murs, les fenêtres, etc. Nous proposons dans cette partie une méthode innovante pour retrouver la sémantique d'un géométrie 3D. La sémantique produite reflète la décomposition hiérarchique du modèle. L'entrée est un modèle géométrique, constitué de polygones, et la sortie est un ensemble d'informations sémantiques attachées à ces mêmes polygones.

Les architectes peuvent aussi faire usage d'un tel outil après avoir créé une esquisse rapide d'un bâtiment pour présentation à des clients, avant de produire des plans et des maquettes détaillés. Bien que les outils de CAO permettent souvent de faire un rendu graphique, ils s'avèrent souvent inadaptés aux grands projets. Les architectes passent par une étape de graphisme pour obtenir des vues spectaculaires de leur projet. Pour ce faire, ils doivent fournir l'ensemble des informations sémantiques au graphiste. Ceci est généralement fait en associant aux éléments géométriques une couleur qui sera ensuite

remplacée par les textures adéquates. L'objet de ce chapitre est de faire cet étiquetage automatiquement.

A une autre échelle, dans l'industrie du jeu vidéo, les concepteurs de niveaux ont tendance à se focaliser sur la géométrie, transmettant peu d'informations aux graphistes, qui doivent souvent peindre séparément chaque élément de la scène.

9.2 Travaux précédents

Nombre de travaux ont été menés sur la segmentation sémantique et la classification. La plupart des méthodes concernant les objets complexes sont basées sur l'analyse de graphes.

Les graphes peuvent être générés soit à partir des squelettes volumiques [Biasotti et al., 2003] ou de décompositions de surfaces, soit à partir d'une géométrie « parfaite »[El-Mehalawi and Miller, 2003], soit de nuages de points [Schnabel et al., 2007] ou encore de maillages [Pascucci et al., 2007]. Ils peuvent être automatiquement appris à partir d'exemples [Tangelder and Veltkamp, 2008] ou créés « à la main »[Schnabel et al., 2008].

Les algorithmes de mise en correspondance de graphes (graph matching) sont ensuite utilisés pour retrouver des objets précis dans une scène, avec la possibilité d'autoriser une mise en correspondance incomplète : sous-graphe commun de taille maximale, mise en correspondance avec une distance d'édition [Gao et al., 2010], etc.

Bien que ces problèmes de mise en correspondance de graphes soient NP-difficiles, des algorithmes basiques et des heuristiques sont utilisables, pourvu que le graphe ne soit pas trop grand.

Or, nous nous intéressons ici à des objets, par exemple des bâtiments, qui sont complexes :

1. Ce sont des objets à haut degré de structuration hiérarchique, c-à-d qu'ils se décomposent hiérarchiquement en un grand nombre de composants.

2. Le nombre d'éléments constituant les objets est variable et a priori inconnu.

3. Les relations entre les différentes parties peuvent être complexes, elles ne se limitent pas à l'adjacence ou à la position relative.

L'objectif est ici de reconnaître non seulement des objets mais aussi leur décomposition. Pour ce genre d'objets, la structure sous-jacente adéquate semble être une grammaire de forme :

1. Chaque règle de production représente un niveau de décomposition différent dans la hiérarchie.

2. Des règles récursives expriment un nombre arbitraire d'éléments.

3. Les grammaires peuvent exprimer des contraintes représentant les relations complexes entre les composants.

De plus, les grammaires sont naturellement modulaires et se veulent intuitives. Elles permettent la conception et la maintenance de larges spécifications, contrairement à des approches qui codent des règles en dur et qui requièrent une connaissance en programmation [Martinović et al., 2012].

Les grammaires de formes ont été utilisées pour la génération procédurale de modèles [Müller et al., 2006], mais rarement inversées pour analyser la scène, ou alors en se

limitant à la 2D, notamment pour l'analyse de façades [Müller et al., 2007; Ripperda and Brenner, 2007; Teboul et al., 2011].

Les modèles génératifs en 3D ont peu servi pour l'analyse, et ont été restreints à des tâches spécifiques et codées en dur telles que la détection de toitures [Huang et al., 2011] et de bâtiments avec hypothèse de monde Manhattan [Vanegas et al., 2010, 2012].

Récemment, la modélisation procédurale a été utilisée pour l'analyse en multi-vues 3D [Simon et al., 2012]. Cependant, l'utilisation de la 3D se limite à l'ajout d'un terme de profondeur dans une énergie 2D. La 3D n'est pas calculée à partir de l'analyse. De plus l'espace de recherche est particulièrement grand. Le passage à l'échelle de modèles volumineux ainsi qu'à la 3D pure n'est pas immédiat.

Contrairement aux méthodes citées précédemment qui ont une approche de haut en bas (top-down), du modèle abstrait vers les données, les approches de bas en haut (bottom-up) partent des données pour construire le modèle abstrait correspondant. Han and Zhu [2009]; Wu and Zhu [2011] s'en sont servis avec l'ajout de composants top-down pour prédire les parties manquantes ou cachées. Cependant, ils se sont limités à l'utilisation d'images et de primitives simples, par exemple, des projections de rectangles et de cercles en 3D.

Mathias et al. [2011] ont utilisé une variante multi-vues de l'approche top-down, avec des parties spécifiques à certains types d'objets dont une partie est codée en dur, et ont employé des recherches exhaustives de compatibilité entre formes, sans propagation de contrainte. Ils obtiennent de bons résultats sur la reconstruction de temples doriques.

En 3D, une approche grammaticale a été utilisée par Toshev et al. [2010]. Les règles sont appliquées dans un ordre spécifique ce qui rend la spécification difficile à écrire, à mettre au point et peu modulaire. Elle ne peut pas être généralisée à n'importe quel jeu de règles grammaticales.

9.3 Approche proposée

La méthode d'étiquetage présentée dans cette partie est entièrement bottom-up bien qu'il soit possible d'y inclure des éléments de prédiction des parties manquantes (top-down).

Cette approche se base sur l'utilisation de grammaires attribuées avec contraintes, associées à des prédicats géométriques spécifiques (section 9.4).

Les terminaux, composants de base, sont des primitives 3D (polygones). Ils sont combinés via des règles de production pour créer des non-terminaux, éléments de complexité plus élevée (figure 9.1).

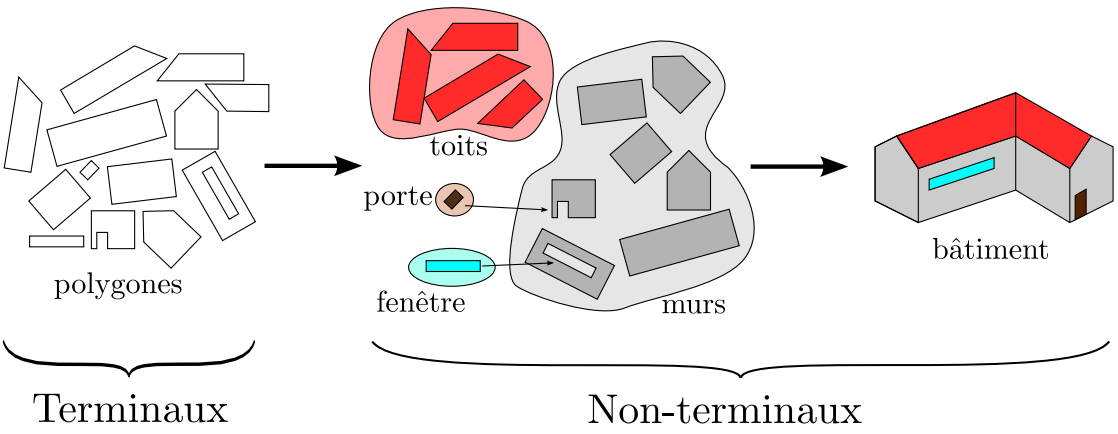


FIGURE 9.1 – Terminaux et non-terminaux

Ce mode de production, qui consiste à envisager les différents non-terminaux qu'il est possible de créer, peut mener à une explosion combinatoire. Pour endiguer ce phénomène, notre algorithme utilise une propagation de contraintes pour réduire l'espace de recherche au maximum. Il repose en particulier sur l'utilisation de prédicats inversibles. Les grammaires attribuées avec contraintes ont été utilisées par le passé, mais dans une version beaucoup plus restreinte, sans propagation de contraintes et avec une unique interprétation, pour l'analyse de formules mathématiques [Pagallo, 1998].

Les contributions de ce chapitre sont :

- La définition des grammaires attribuées avec contraintes pour des objets complexes, y compris en 3D.
- L'utilisation d'un formalisme pratique, aisément paramétrable, et accessible à des personnes non issues du domaine de l'informatique, par opposition aux descriptions bas niveau de méthodes basées sur les graphes ou codées en dur qui nécessitent des connaissances en programmation. Ceci permet la conception et la modification de spécifications de gros ensembles.
- L'introduction d'opérateurs maximaux, pour réduire l'espace de recherche.
- Le calcul efficace d'une forêt d'analyse (ensemble d'arbres d'analyse) pour une grammaire et un ensemble de terminaux (polygones) donnés.
- L'évaluation de la méthode sur un ensemble de modèles de bâtiments, synthétiques ou reconstruits à partir de données réelles.

L'objet de la thèse étant la reconstruction automatique de maquettes numériques de bâtiments, les tests sont effectués sur des données de ce type. Cependant, il faut noter que ce formalisme peut s'appliquer à d'autres objets, tant que ceux-ci sont décomposables de manière hiérarchique.

9.4 Grammaires attribuées avec contraintes

Les grammaires décrivent des objets complexes qui se décomposent hiérarchiquement. Un des principaux domaines d'application des grammaires formelles est la segmentation et la compréhension de textes. La partie gauche de la figure 9.2 montre la décomposition d'une phrase en un sujet, un verbe et un complément. Les éléments constitutifs du sujet sont aussi représentés : un article, un adjectif et un nom. Cette décomposition est hiérarchique, elle peut se représenter sous la forme d'un arbre dont la racine est le nœud de plus grande complexité, *phrase*.

La partie droite de la figure 9.2 illustre comment une telle décomposition hiérarchique peut être adaptée à l'analyse de bâtiments. Ici, le bâtiment (très sommaire) est constitué de façades, d'un toit et d'un plancher. Une des façades se décompose elle-même en un mur et une fenêtre.

Cependant, une décomposition telle que présentée précédemment n'est pas suffisante pour rendre entièrement compte de la complexité de la phrase. En effet, ici les sous-arbres issus du nœud *phrase* sont indépendants les uns des autres, ce qui n'est pas vrai. Comme le montre la figure 9.3, le verbe s'accorde avec le sujet, ce qui traduit une interaction entre le nœud *sujet* et le nœud *groupe verbal*. Il en est de même pour le bâtiment où le mur et le toit doivent être adjacents.

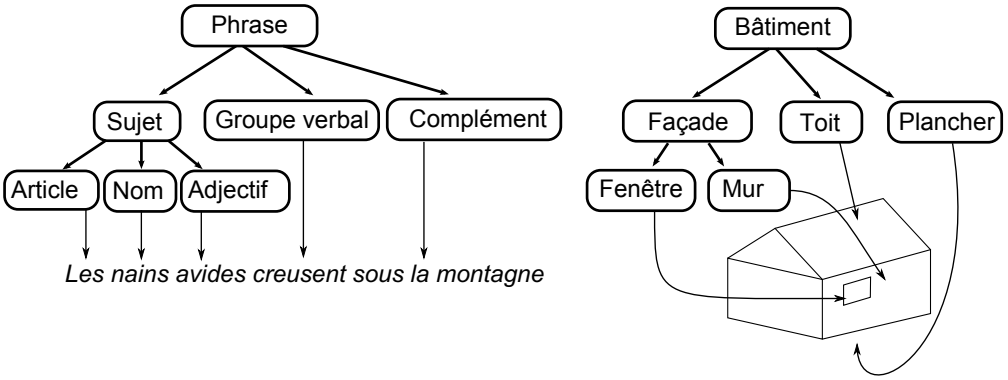


FIGURE 9.2 – Décomposition hiérarchique d'une phrase et d'un bâtiment.

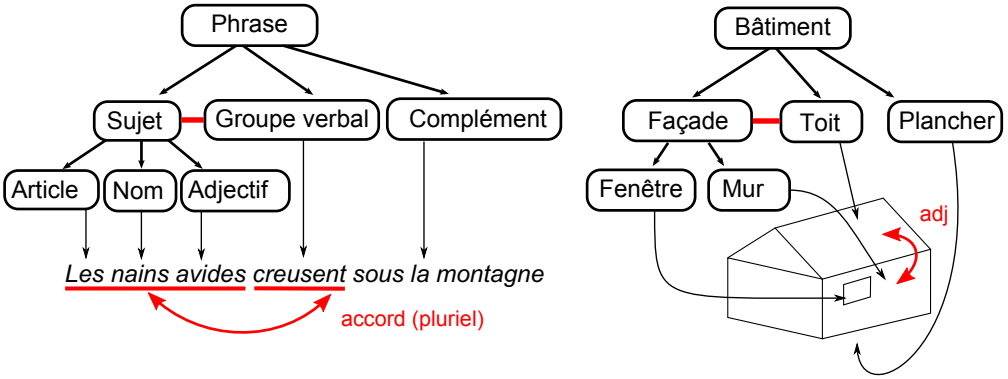


FIGURE 9.3 – Décomposition avec contraintes.

Si une décomposition hiérarchique simple peut être exprimée sous la forme d'une grammaire attribuée, elle ne suffit pas à rendre compte des relations complexes entre les éléments. C'est pourquoi, nous utilisons des grammaires attribuées avec contraintes, qui régissent les interactions entre éléments.

9.4.1 Définitions

Une grammaire G est un quadruplet :

$$G = (T, N, P, S) \tag{9.1}$$

où :

- T est un ensemble de terminaux. Les terminaux correspondent aux primitives géométriques de la scène, par exemple, des polygones à trous.
- N est un ensemble de non-terminaux. Cet ensemble est disjoint de T. Un non-terminal représente un objet complexe, qui se décompose par G en un ensemble de non-terminaux et/ou de terminaux.
- P est un ensemble de règles de production. Une règle de production définit la décomposition d'un non-terminal en ses sous-éléments.
- S est un ensemble de symboles de départ, ou racines de la décomposition hiérarchique. Ce sont les objets finaux souhaités. On a $S \subset N$.

Les sections suivantes décrivent le formalisme des règles de l'ensemble P, en partant des règles simples jusqu'aux formes plus complexes qui manipulent des collections d'objets.

9.4.2 Règle basique

La forme la plus simple d'une règle de production r est :

$$r \; = \; Y \rightarrow X_1, \ldots, X_k \tag{9.2}$$

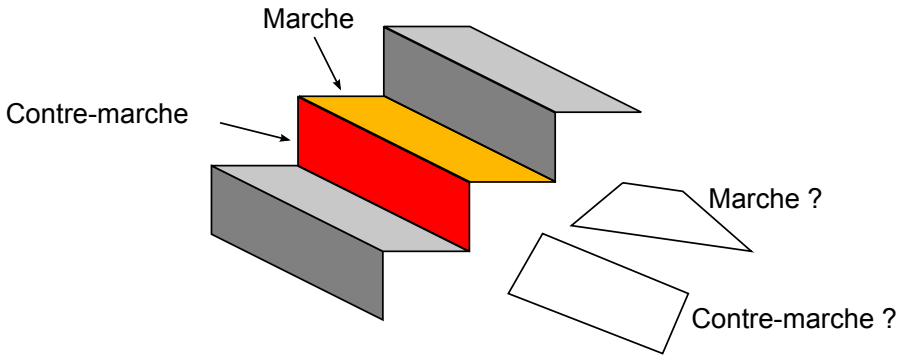
r exprime la décomposition de $Y \in N$ en $\{X_1, \ldots, X_k\} \in (N \cup T)^k$, pour $k \geq 1$. En pratique dans le cadre d'utilisation de ce chapitre, nous appliquons les règles de bas en haut, dans le sens inverse de la production ordinaire : étant donnés $X_1, \ldots, X_k$, nous créons Y.

Par exemple, si les données en entrée incluent des polygones, nous créons un terminal de type *Polygon* pour chacun d'eux. Par exemple, la règle :

$$\mathit{Tread} \rightarrow \mathit{Polygon}$$

définit la création des éléments non-terminaux du type *Tread* (une marche) à partir d'un élément de type *Polygon* (un polygone).

La figure 9.4 donne un autre exemple simple pour la construction d'un bloc-marche d'escalier. Ici le bloc-marche est généré à partir d'une marche et d'une contre-marche. Un des blocs-marche possibles est coloré en orange et rouge. Cependant d'après la définition de la règle de production, toutes les paires contre-marche/marche peuvent former un bloc-marche. Le bloc-marche coloré est donc un bloc-marche dans un grand ensemble de possibilités.



Règle : *Step* $\rightarrow$ *Tread*, *Riser*

FIGURE 9.4 – Exemple de construction de bloc-marche.

La barre verticale est utilisée pour la disjonction :

$$Y \rightarrow \; X_1 | X_2 \Longleftrightarrow \begin{cases} Y & \rightarrow \; X_1 \\ Y & \rightarrow \; X_2 \end{cases}$$

9.4.3 Contraintes

Pour obtenir des blocs-marche plausibles, il est nécessaire d'ajouter des contraintes. Par exemple il est naturel d'imposer une condition d'adjacence ou d'orthogonalité entre le plateau et la contre-marche.

Contrairement aux grammaires 1D et aux grammaires de découpage de forme (split grammars), une règle $r \; = \; Y \rightarrow X_1, \ldots, X_k$ n'exprime que la domination de Y sur les X_i et non un ordre sur les X_i. Pour gérer la 3D, les contraintes entre les X_i pour

$i \in \{1, \ldots, k\}$ sont regroupées dans une condition C attachée à la règle r. La notation utilisée est $r \ \langle C \rangle$. De plus comme il peut y avoir plusieurs occurrences du même élément de grammaire parmi les X_i (cas où $X_i = X_j$), chaque occurrence reçoit un nom de variable différent :

$$Y \ y \rightarrow X_1 \ x_1, \ldots, X_k \ x_k \ \ \langle C \rangle \tag{9.3}$$

La condition agit sur les variables ainsi définies.

Par exemple, sur la figure 9.4, si on impose l'adjacence via une arête entre la contremarche et la marche, la règle devient :

$$\mathit{Step} \ s \rightarrow \mathit{Tread} \ t, \mathit{Riser} \ r \ \langle \mathsf{edgeAdj}(t, r) \rangle$$

9.4.4 Attributs

Chaque élément de grammaire possède des attributs qui décrivent la géométrie de l'élément, qu'il soit un polygone ou un objet complexe. Un attribut peut être une valeur : un entier, un réel, un booléen... ou encore un élément de la grammaire. Par exemple, il est possible d'accéder à l'aire de chaque polygone, ou le volume de la boîte englobante de chaque objet, ou encore son orientation. Lorsqu'un attribut est un élément de grammaire, c'est en général un des constituants de l'objet, dont il est alors possible d'atteindre à son tour les attributs.

Les grammaires considérées ici sont en fait des grammaires attribuées :

$$G = (T, N, P, S, A) \tag{9.4}$$

où A est un ensemble de noms d'attributs disponibles pour être utilisés dans une condition. Une condition peut donc porter sur les $(x_i)_{i \in \{1,\ldots,k\}}$ mais aussi sur les $(x_i.a)_{i \in \{1,\ldots,k\}, a \in A}$. Les attributs peuvent être des scalaires, des vecteurs ou des objets représentés par un terminal ou un non-terminal.

Sur l'exemple du bloc-marche d'escalier, il est possible d'écrire une règle du type :

$$\mathit{Step} \ s \rightarrow \mathit{Tread} \ t, \mathit{Riser} \ r \ \langle \mathsf{edgeAdj}(t, r), t.\mathit{len} == r.\mathit{len} \rangle$$

où *len* désigne la plus grande dimension de la boîte englobante orientée.

Les attributs considérés ne sont que des attributs hérités. Les attributs d'un non-terminal en partie gauche de règle de production sont les noms des variables indiquées en partie droite, qui permettent d'accéder aux attributs spécifiques de chaque élément en partie droite, ainsi que les des attributs géométriques généraux qui donnent des informations sur l'objet composite complexe. En pratique, ces attributs généraux peuvent être calculés à la volée pour toute application de règle. Dans une règle $Y \rightarrow X_1, \ldots, X_k$, les attributs de Y sont déterminés par les attributs de $X_1, \ldots, X_k$ mais les attributs des X_i ne dépendent pas de Y ni des autres $(X_j)_{j \neq i}$.

Certains attributs sont calculés pour chaque forme, comme par exemple la boîte englobante minimale orientée, la largeur, la longueur et la profondeur. Ces attributs sont résumés dans la table 9.1.

Plus généralement, une règle peut s'écrire alors de la forme :

$$Y \ y \rightarrow X_1 \ x_1, \ldots, X_k \ x_k \ \ \langle C \rangle \ \ \{E\} \tag{9.5}$$

où E, optionnel, est un ensemble de règles pour la création de nouveaux attributs à partir de ceux des x_i.

Attribut	Type	Signification
area	Scalaire	Aire (pour un polygone)
nbHoles	Scalaire	Nombre de trous du polygones (pour un polygone)
length	Scalaire	Longueur de la boîte englobante orientée
lengthVector	Vecteur	Direction principale de la boîte englobante orientée
breadth	Scalaire	Largeur de la boîte englobante orientée
breadthVector	Vecteur	Second direction de la boîte englobante orientée
depth	Scalaire	Profondeur de la boîte englobante orientée
depthVector	Vecteur	Plus petite direction de la boîte englobante orientée
bbx	Scalaire	Dimension de la boîte englobante non orientée suivant x
bby	Scalaire	Dimension de la boîte englobante non orientée suivant y
bbz	Scalaire	Dimension de la boîte englobante non orientée suivant z
size	Scalaire	Taille (pour une collection)

TABLE 9.1 – Liste des attributs.

9.4.5 Prédicats

Une condition C est une conjonction de prédicats s'appliquant sur les variables x_i de la règle. Un prédicat pose une question dont la réponse est un booléen.

Les prédicats expriment notamment des conditions géométriques, mais peuvent plus généralement représenter n'importe quelle contrainte sur les objets ou leurs attributs. La plupart des attributs concernent à la fois les terminaux et les non-terminaux. Les principaux prédicats utilisés dans ce chapitre sont :

- **Adjacence** : l'adjacence générale entre deux instances est le prédicat $\mathsf{adj}(x, y)$. L'adjacence via une arête $\mathsf{edgeAdj}(x, y)$, via une arête et par un trou $\mathsf{intEdgeAdj}(x, y)$, via une arête et un bord extérieur $\mathsf{extEdgeAdj}(x, y)$ ou via un sommet $\mathsf{vertAdj}(x, y)$ sont des spécialisations de $\mathsf{adj}(x, y)$.

  En pratique, le graphe d'adjacence est construit pour les terminaux. L'adjacence des non-terminaux est calculée en observant celle des terminaux dont il est composé.

- **Orientation** : pour un système de coordonnées donné, les prédicats d'horizontalité $\mathsf{horizontal}(x)$ et de verticalité $\mathsf{vertical}(x)$ sont définis. Ils s'appliquent aux vecteurs.

- **Orientation relative.** : des prédicats permettent de définir la configuration géométrique d'un objet par rapport à un autre. Les prédicats $\mathsf{parallel}(x, y)$, respectivement $\mathsf{orthog}(x, y)$, défini pour des vecteurs, indiquent si x et y sont parallèles, respectivement orthogonaux.

  $\mathsf{above}(x, y)$, respectivement $\mathsf{under}(x, y)$, indiquent si x est au-dessus, respectivement en-dessous, de y. Ces deux prédicats ne sont pas inversibles et agissent sur des non-terminaux ou des terminaux.

- **Opérateurs de comparaison sur les attributs** : l'égalité $x == y$ et la différence $x! = y$ pour des scalaires et des vecteurs, les inégalités inférieures et supérieures $x < y$, $x \leq y$, $x > y$, $x \geq y$ pour des scalaires.

Parce que la géométrie en entrée ne peut pas être parfaite, les prédicats exacts peuvent être relâchés. L'adjacence comme la position relative sont définies à une distance près et une erreur angulaire est autorisée pour l'orientation (voir section 9.7.1).

En outre, dans notre implémentation, des prédicats supplémentaires peuvent être définis par l'utilisateur en en fournissant le code source, mais leur utilisation n'est pas optimisée pour l'ordre d'application des prédicats (sections suivantes).

9.4.6 Collections maximales

Les collections décrivent des ensembles d'éléments dont on ne connaît pas la taille a priori.

C'est le cas pour un escalier, défini comme un ensemble de blocs-marche adjacents. Le nombre de blocs-marche k est inconnu, il est donc impossible d'écrire la règle de création de cet escalier sous la forme :

$$\mathit{Stairway}\ sw \rightarrow \mathit{Step}\ s_1, \ldots, \mathit{Step}\ s_k\ \langle C \rangle$$

Une collection Y d'éléments de type X peut être définie récursivement sous la forme :

$$Y \rightarrow X, Y \mid \emptyset$$

Il est alors possible de définir des opérateurs qui permettent de construire des objets composites tels que des ensembles, des composantes connexes, des séquences, etc. Cependant l'utilisation de ce genre d'opérateur est difficile voire inappropriée dans le cadre d'une approche bottom-up. Par exemple, s'il y a m instances de X, il y a jusqu'à 2^m instances possibles de Y, ce qui devient vite incalculable.

Abandonner les collections n'est pourtant pas envisageable. Les cas où le cardinal des ensembles est inconnu sont nombreux. En pratique, dans la plupart des cas, l'ensemble ciblé par l'utilisateur n'est pas n'importe quelle instance de Y, mais la collection de cardinal maximal. Par exemple, pour l'escalier, l'objectif est de désigner la séquence maximale de marches, et non pas toutes les séquences possibles.

Rechercher les collections maximales limite alors grandement le nombre d'instances possibles de Y.

Une collection maximale est de la forme :

$$Y \rightarrow \mathsf{maxcol}(X, c_u, c_b)\ mc \tag{9.6}$$

Elle désigne une collection maximale particulière d'instances de X, avec des contraintes $c_u : X \rightarrow \mathit{bool}$ unaires agissant sur des x_i de type X dans la collection et des contraintes $c_b : X \times X \rightarrow \mathit{bool}$ binaires agissant sur des paires (x_i, x_j) d'éléments de type X dans la collection. La mention des contraintes unaires ou binaires est optionnelle. Si l'une d'elles est absente, elle est supposée satisfaite par défaut.

Les opérateurs de collections maximales sont :

- **L'ensemble maximal** : $\mathsf{maxset}(X, c_u, c_b)$ est un ensemble de cardinal m, maximal, tel que pour tout $1 \leq i \leq m$, $c_u(x_i)$ soit remplie et que pour $1 \leq i, j \leq m$, $c_b(x_i, x_j)$ le soit également.
- **La composante connexe maximale** : $\mathsf{maxconn}(X, c_u, c_b)$ est un ensemble de cardinal m, maximal, tel que pour tout $1 \leq i \leq m$, $c_u(x_i)$ soit remplie et pour tout $1 \leq i < j \leq m$, il existe une séquence $1 \leq e_1, \ldots, e_l \leq m$ tels que $e_1 = i$ et $e_l = j$ et pour chaque $1 \leq k < l$, les conditions $c_b(e_k, e_{k+1})$ et $c_b(e_{k+1}, e_k)$ soient remplies.
- **La séquence maximale** : $\mathsf{maxseq}(X, c_u, c_b)$ est un ensemble de cardinal m, maximal, tel que pour tout $1 \leq i \leq m$, $c_u(x_i)$ soit remplie et pour tout $1 \leq i < m$, la condition $c_b(x_i, x_{i+1})$ soit remplie.
- **Le cycle** : $\mathsf{cycle}(X, c_u, c_b)$ est un ensemble de cardinal m, maximal, tel que pour tout $1 \leq i \leq m$, $c_u(x_i)$ soit remplie et pour tout $1 \leq i \leq m$, la condition $c_b(x_i, x_{i+1})$ soit remplie (en considérant que $x_m + 1 = x_1$).

Reprenant l'exemple de l'escalier, la règle de création de l'escalier est une séquence maximale :

$$Stairway\ sw \rightarrow \mathsf{maxseq}(Step\ s, \mathsf{edgeAdj})$$

Les opérateurs de collections représentent une instance d'un objet complexe. Chacune de ces instances a les attributs ordinaires définis pour tous les objets, auxquels s'ajoute la notion de taille : attribut *size*. Il est ainsi possible de contraindre la taille des séquences voulues. Dans le cas précédent, si on souhaite définir un escalier comme étant une séquence maximale de marches d'au moins trois marches :

$$Stairway\ sw \rightarrow \mathsf{maxseq}(Step\ s, \mathsf{edgeAdj})\ ms\ \ \langle ms.size \geq 3 \rangle$$

9.4.7 Contexte

Il est souvent nécessaire de faire référence à d'autres variables qui ne sont pas des constituants de Y. Par exemple, une fenêtre est adjacente à un mur, mais le mur n'est pas un élément de la fenêtre.

Pour traiter de tels cas, des variables de contexte sont introduites. Par exemple :

$$Y\ y\ \rightarrow\ X\ x,\ \mathsf{context}(Z\ z)\ \ \langle \mathsf{adj}(x, z) \rangle$$

Pour cette règle, une instance de Y est uniquement constituée d'un élément de type X, mais celui-ci doit être adjacent à une instance de Z. z est alors une variable contextuelle.

9.4.8 Instances optionnelles

Une instance optionnelle traduit le fait qu'elle n'est pas nécessaire à la résolution de la règle, mais si celle-ci est présente, elle doit être prise en compte. Dans une règle, cela est indiqué de la manière suivante :

$$Y\ y\ \rightarrow\ X_1\ x_1,\ \mathsf{optional}(X_2\ x_2) \tag{9.7}$$

Cette règle est équivalente à :

$$Y\ y\ \rightarrow\ X_1\ x_1,\ \mathsf{maxset}(X_2)\ x_2\ \langle x_2.size \leq 1 \rangle$$

En revanche, elle n'est pas équivalente à la règle :

$$Y\ y\ \rightarrow\ X_1\ x_1 | X_1\ x_1,\ X_2\ x_2 \tag{9.8}$$

En effet, lorsque X_2 est présent, la règle (9.7) produit un seul élément dont les fils sont de type X_1 et X_2. La règle (9.8) quant à elle produit deux éléments, l'un dont les fils sont de type X_1 et X_2, et l'autre avec un seul fils de type X_1.

Un exemple de grammaire pour l'extraction d'un escalier est donné figure 9.5. Les éléments mentionnés sont représentés sur la figure 9.6, à gauche. La partie droite de cette dernière est une vue colorée d'une reconnaissance d'escalier dans un modèle CAO. Les contre-marches sont colorées en rouge, les marche en marron clair.

$$
\begin{array}{lll}
\textit{Tread } t & \rightarrow \textit{Polygon } p & \langle \mathsf{horizontal}(p),\ p.width \leq 2.0 \rangle \\
\textit{Riser} - \textit{Base } rb & \rightarrow \textit{Polygon } p & \langle \mathsf{vertical}(p),\ 0.05 \leq p.height \leq 0.25 \rangle \\
\textit{Nose} - \textit{Elt } ne & \rightarrow \textit{Polygon } p & \langle \mathsf{horizontal}(p.lengthVec),\ p.height \leq 0.05 \rangle \\
\textit{Riser } r & \rightarrow \textit{Riser} - \textit{Base } rb,\ \mathsf{maxseq}(\textit{Nose} - \textit{Elt } ne, \mathsf{edgeAdj})\ n & \\
 & & \langle \mathsf{edgeAdj}(rb, n),\ \mathsf{above}(n, rb) \rangle \\
\textit{Step } s & \rightarrow \textit{Tread } t,\ \textit{Riser } r & \langle \mathsf{edgeAdj}(t, r),\ \mathsf{above}(t, r) \rangle \\
\textit{Stairway } sw & \rightarrow \mathsf{maxseq}(\textit{Step } s, \mathsf{edgeAdj})\ ms,\ \textit{Riser } r & \\
 & & \langle \mathsf{edgeAdj}(ms, r),\ \mathsf{above}(r, ms) \rangle
\end{array}
$$

FIGURE 9.5 – Grammaire pour un escalier

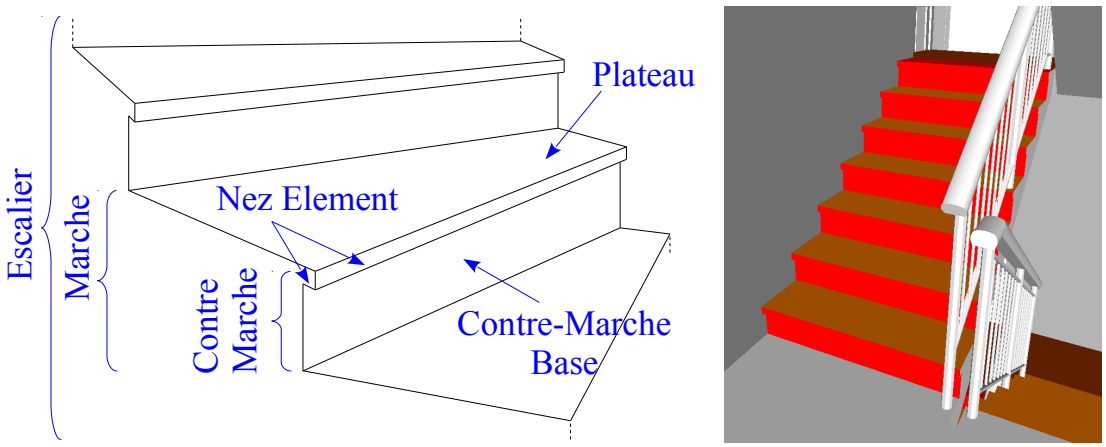


FIGURE 9.6 – Les éléments de la grammaire d'escalier (à gauche) et un exemple de reconnaissance automatique sur un modèle CAO (à droite).

9.5 Interprétation de la scène

Interpréter une scène, avec en entrée une grammaire et un ensemble de primitives, consiste à construire un ou plusieurs arbres d'analyse. L'arbre d'analyse reflète une structuration possible de la scène vis-à-vis de la grammaire. C'est une représentation permettant de décrire simplement la décomposition hiérarchique de la scène, les étiquettes sémantiques, les primitives et les relations complexes entre elles.

9.5.1 Arbre d'analyse

Un arbre d'analyse t est un arbre qui reflète des instanciations des règles de la grammaire.

Chaque nœud n de t correspond soit à un terminal, si n est une feuille, soit à un non-terminal.

Un nœud terminal est associé à chaque primitive géométrique détectée dans la scène. Dans ce chapitre, les primitives sont des polygones à trous en 3D, mais le formalisme autorise l'utilisation d'autres types de primitives : cylindres, sphères, etc.

Chaque nœud non-terminal correspond à une instanciation d'une règle de la grammaire r :

$$Y\ y \rightarrow X_1\ x_1, \ldots, X_k\ x_k\ \ \langle C \rangle$$

en ce sens que :

- Le type du nœud n, noté $\tau(n)$, est Y.

- Les enfants de n sont des nœuds $(n_i)_{1\leq i\leq k}$ de type $(\tau(n_i) = X_i)_{1\leq i\leq k}$.
- Associer $(n_i)_{1\leq i\leq k}$ aux variables $(x_i)_{1\leq i\leq k}$ satisfait la condition $\langle C\rangle$ de r.

L'ensemble des terminaux du sous-arbre de racine n est appelé une forme de n, ou plus généralement une forme de type $\tau(n)$.

La figure 9.7 donne un exemple de représentation d'arbre d'analyse. Les feuilles (les terminaux) sont les polygones et la racine (symbole de départ) est le bâtiment.

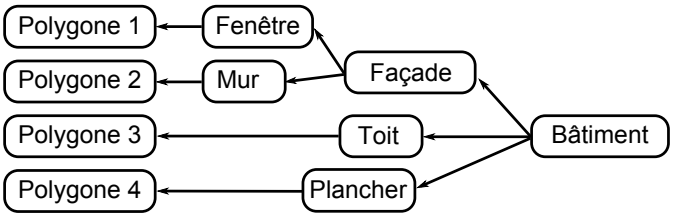
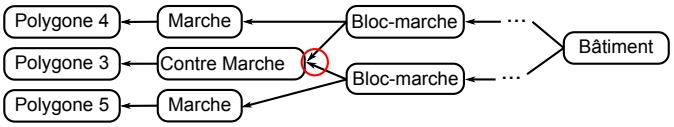


FIGURE 9.7 – Exemple d'arbre d'analyse.

Contrainte d'exclusivité Dans le cas des grammaires 1D, par exemple appliquées à l'analyse de textes, les terminaux (les mots) en entrée sont ordonnés. De ce fait, la séquence des terminaux est égale à la chaîne de caractères en entrée. Les mots sont utilisés une fois, et une fois seulement. En 3D, il n'est pas possible de définir un ordre sur les primitives. Pour assurer que les arbres d'analyse soient valides, il est nécessaire que les objets ne partagent pas de polygones, sauf entre parents et enfants. Cette contrainte sur les arbres d'analyse est la contrainte d'exclusivité. Cela se traduit par le fait qu'une primitive n'a qu'une interprétation : un polygone qui appartient à un escalier n'appartient pas à une fenêtre. La figure 9.8 montre un exemple de graphe pouvant être produit lorsque la contrainte d'exclusivité est absente. En pratique, pour qu'une instance de Y associée à n_Y soit valide, il suffit que les ensembles de polygones, dont les formes de ses nœuds fils sont issues, soient disjoints.



Ici une contre-marche appartient à deux instances de marche dans le même bâtiment. Le résultat n'est pas un arbre.

FIGURE 9.8 – Exemple de construction sans contrainte d'exclusivité.

9.5.2 Forêt d'analyse

Dans le cas d'un texte, l'analyse doit en général fournir une explication pour l'ensemble de la phrase. Pour l'analyse de scènes 3D, l'approche est différente. La complexité et la diversité des objets de la scène ne permettent pas de donner de manière exhaustive une interprétation de tous les éléments. L'objectif est plutôt de localiser un maximum d'objets au milieu d'un ensemble de données sans étiquette.

Contrairement à la plupart des autres approches basées sur les grammaires qui produisent un unique arbre d'analyse, la sortie de notre algorithme est une forêt partagée F, représentant de manière compacte l'ensemble des arbres d'analyse possibles.

Lors de l'analyse, toutes les instances valides de chaque élément sont créées. Du fait de la contrainte d'exclusivité présentée dans la section précédente, il est possible de générer plusieurs interprétations mutuellement exclusives de la scène. La figure 9.9 montre deux interprétations générées à partir des mêmes polygones. Ces deux arbres sont mutuellement exclusifs car ils reposent sur les même polygones. Dans l'un, le polygone 1 est considéré comme une fenêtre. Dans l'autre, c'est une porte.

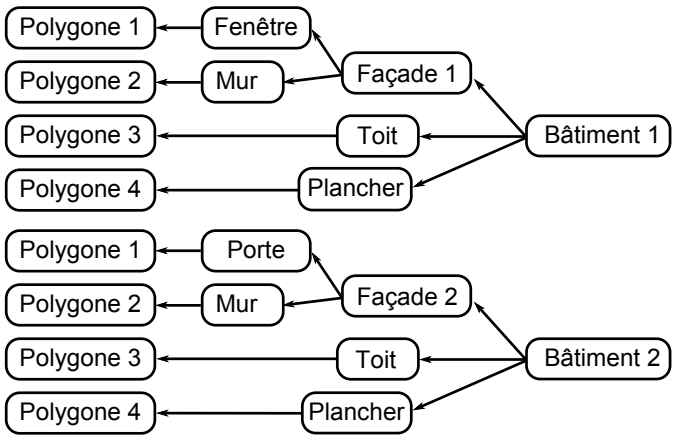


FIGURE 9.9 – Deux interprétations mutuellement exclusives.

Le nombre d'interprétations est lié au degré de contrainte de la grammaire. Plus la grammaire est contrainte, plus le nombre d'interprétations sera réduit. En pratique, dans les expériences menées, les contraintes grammaticales utilisées suffisent à obtenir une unique, quelques ou juste un petit nombre d'interprétations.

Pour représenter F, plutôt que d'énumérer tous les arbres, un graphe orienté acyclique (Directed Acyclic graph, DAG) est utilisé. Un nœud n de F peut appartenir à plusieurs arbres. Cette représentation est compacte, la figure 9.10 montre la compression des deux arbres de la figure 9.9 en un DAG.

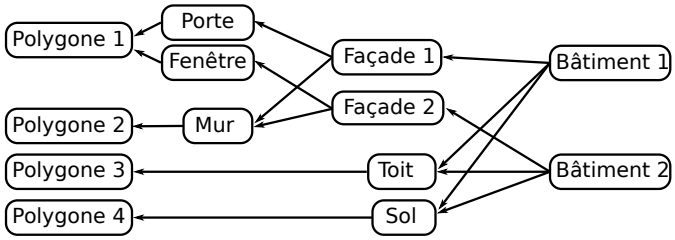


FIGURE 9.10 – Représentation compacte d'une forêt sous la forme d'un DAG.

La notion de partage de nœuds introduite par la représentation sous forme de DAG, n'est pas incompatible avec la notion d'exclusivité. Pour cela, il suffit que deux formes partageant un terminal ne soient pas associées pour créer une nouvelle forme.

Par exemple, si une marche appartient à un escalier, elle ne peut pas appartenir à un autre escalier de la même instance (interprétation) de bâtiment. Par contre, rien n'interdit qu'elle appartienne à un autre escalier, dans une instance (interprétation) différente de bâtiment. Les deux instances de bâtiment sont dites mutuellement exclusives.

En raison de cette contrainte, les différentes interprétations de la scène sont exactement tous les sous-ensembles de nœuds racines (inclus dans les symboles de départ) tels que les arbres issus de ces racines soient deux à deux disjoints.

9.6 Analyse Bottom-Up

Cette partie présente comment l'analyse de la scène est en pratique effectuée, étant donnés une grammaire et un ensemble de polygones.

9.6.1 Calcul de la forêt d'analyse

L'algorithme est purement bottom-up. Il part des primitives puis applique itérativement les règles de la grammaire pour créer de nouveaux nœuds non-terminaux. Au départ la forêt ne contient que les terminaux (les polygones) puis celle-ci est enrichie au fur et à mesure de l'analyse.

Appliquer une règle consiste à chercher dans la forêt courante les nœuds ayant le type des variables spécifiées dans le membre droit de la règle, et à générer un nouveau nœud de type correspondant au membre gauche de la règle, ainsi que ses attributs.

Pour assurer la terminaison du processus ainsi que le partage nécessaire à la représentation de la forêt :

1. Tous les sous-arbres identiques sont fusionnés dès leur construction.

2. Les nœuds de type X dont les descendants forment une chaîne 1D contenant un nœud de type X sont rejetés pour éviter d'entrer dans une boucle infinie. De tels cas peuvent apparaître avec des règles de grammaire récursives, qui sont rendues inutiles en pratique, grâce à l'existence d'opérateurs de collection.

3. Lorsqu'aucune règle ne peut être appliquée, l'analyse s'arrête.

En l'absence de récursivité dans la grammaire, la contrainte d'exclusivité implique que l'application des règles finira par ne plus être possible. Ceci est dû au fait que l'ensemble des terminaux est fini (de même que celui des règles).

9.6.2 Cas des opérateurs maximaux

Les opérateurs maximaux imposent de prendre des précautions. Le fait qu'une collection de X soit de cardinal maximal n'est garanti que si toutes les instances de X sont connues.

Sur l'exemple de la figure 9.11, la création de la séquence maximale de marches est appliquée avant la découverte de toutes les marches. Lors de la création de la dernière marche, la collection précédente n'est plus maximale, et donc invalide. Il faut empêcher ce cas de se produire.

Nous considérons le graphe de dépendance de la grammaire. Si celui ci contient un cycle incluant un opérateur maximal, nous sommes dans le cas de la figure 9.11 et la grammaire est considérée par notre algorithme comme inutilisable. La figure 9.12 montre deux exemples de graphes de dépendance. Le graphe de gauche ne contient pas de cycle incluant un opérateur maximal, il est donc valide. Ce n'est pas le cas du graphe de droite.

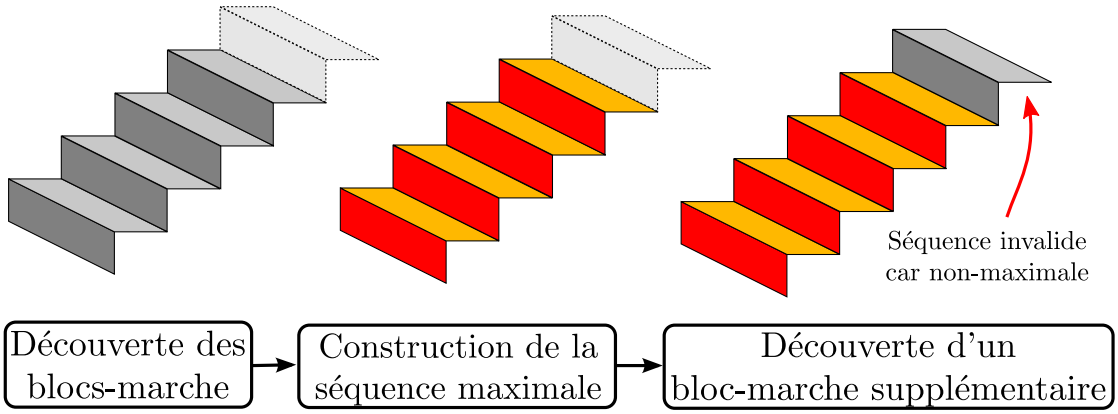


FIGURE 9.11 – Exemple d'application invalide de l'opérateur maximal.

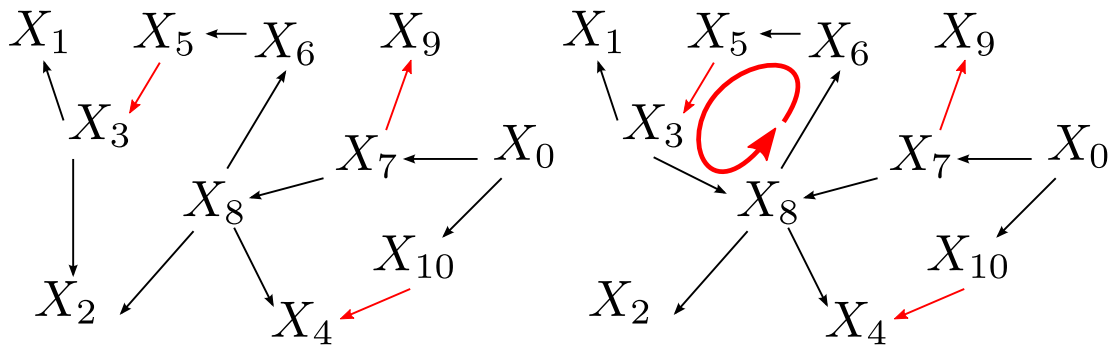


FIGURE 9.12 – Graphes de dépendance valide (à gauche) et invalide (à droite). Les opérateurs maximaux sont représentés par des arêtes rouges.

9.6.3 Ordre d'application des règles

Lorsque la grammaire est considérée comme valide par notre algorithme, il est tout de même nécessaire d'observer un ordre sur l'application des règles pour ne pas tomber dans le cas de la figure 9.11.

La grammaire est décomposée en ensembles de règles pouvant être appliquées ensemble. L'application de ces ensembles est ordonnée : tous les éléments correspondant à un ensemble doivent être calculés avant de passer à l'ensemble suivant.

Les ensembles sont produits en contractant le graphe de dépendance jusqu'à ce que qu'aucune arête ne puisse être contractée.

Contracter une arête $a \to b$ est remplacer $a \to b$ par un sommet ab. Le degré du sommet $deg(ab)$ est égal $deg(a) + deg(b) - 2$. La figure 9.13 illustre une contraction d'arête.

Une arête peut être contractée sauf si elle correspond à un opérateur maximal et si la contracter ne crée pas un cycle incluant un opérateur maximal. Les sommets résultants de la contraction correspondent aux ensembles de règles pour l'ordre d'application. La

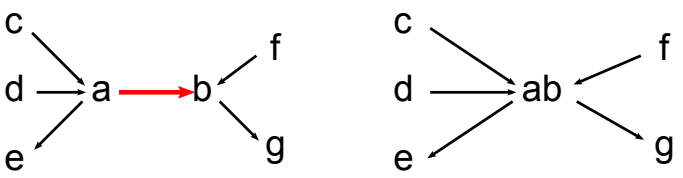


FIGURE 9.13 – Contraction d'un arête.

figure 9.14 montre un exemple de graphe de dépendance et le graphe réduit qui en résulte.

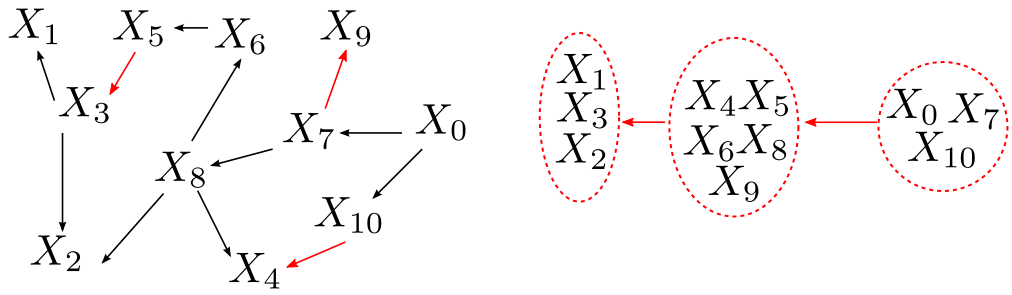


FIGURE 9.14 – Exemple de graphe de dépendance (à gauche) et le graphe de dépendance réduit correspondant (à droite).

L'ordre sur les ensembles est obtenu en utilisant un ordre topologique inverse sur le graphe de dépendance réduit. Cette partition des règles de la grammaire est notée, dans l'ordre, $R(i)_{1\leq i\leq d}$ où d est la taille de la partition.

L'algorithme 5 résume la création de la forêt. Pour chaque élément de la partition $(R_i)_{1\leq i\leq d}$, l'ensemble des règles de R_i est appliqué jusqu'à ce qu'aucune règle ne crée de nouveaux nœuds.

Algorithme 5 Calcul de la forêt d'analyse

$F \leftarrow \{terminal(p) \mid p \text{ primitive géométrique}\}$
$(R_i)_{1\leq i\leq d} \leftarrow$ partition ordonnée de P (voir texte)
pour $i = 1$ **à** d **faire**
  répéter
    inchangé $\leftarrow$ *vrai*
    pour chaque règle $r \in R_i$ **faire**
      $F' \leftarrow appliquerRègle(r, F)$
      si $F' \neq \varnothing$ **alors**
        inchangé $\leftarrow$ *faux*
        $F \leftarrow F \cup F'$
      fin si
    fin pour
  tant que *inchangé*
fin pour

9.6.4 Application d'une règle

Appliquer une règle :

$$Y\ y \rightarrow X_1\ x_1 \ldots X_k\ x_k\ \langle C\rangle\ \{E\}$$

c'est construire de nouveaux nœuds n de type $\tau(n) = Y$ dans la forêt d'analyse F. L'idée de base consiste à récupérer les ensembles $D(x_i)$ de nœuds de F tels que $D(x_i) = \{n \in F, \tau(n) = X_i\}$.

Les $D(x_i)$ peuvent être calculés en une passe sur les éléments de F.

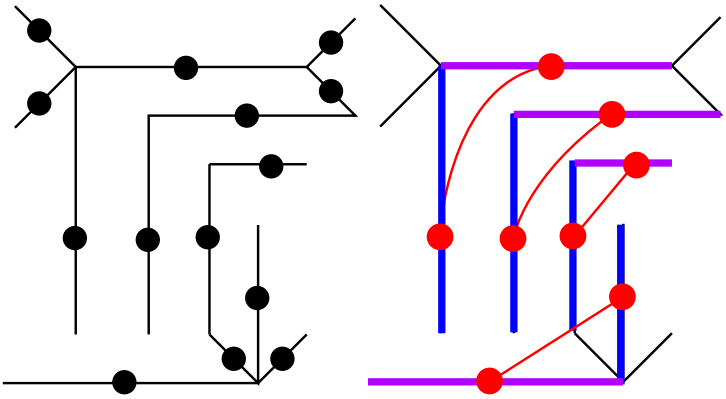
Si un seul des $D(x_i)$ est vide alors la règle ne s'applique pas. Une fois chaque $D(x_i)$ construit, toutes les combinaisons $(n_i)_{1\leq i\leq k} \in \prod_{1\leq i\leq k} D(x_i)$ peuvent être testées par rapport à la condition $\langle C\rangle$. Si chacune des contraintes de $\langle C\rangle$ est satisfaite, un nouveau nœud de type Y est créé ainsi que ses attributs spécifiques tels que définis par $\{E\}$.

9.6.5 Exemple : Influence de l'ordre d'application des contraintes

L'utilisation de la partition $(R_i)_{1\leq i\leq d}$ et l'application des règles comme défini précédemment (section 9.6.4) assurent la prise en compte de la maximalité et de l'exclusivité. Cependant la performance de l'algorithme peut grandement varier avec la manière d'appliquer les contraintes. Cette section présente un exemple pour illustrer les possibles différences. La figure 9.15, à gauche, montre un exemple de scène 2D sur laquelle on souhaite appliquer la règle r :

$$\textit{pair } p \;\rightarrow\; \textit{segment } s_1, \textit{segment } s_2, \quad \langle \mathsf{orthog}(s_1, s_2),\ \mathsf{adj}(s_1, s_2),\ \mathsf{vertical}(s_1) \rangle$$

La partie droite de la même figure, montre le résultat après application de r. La règle cherche à construire les paires répondant à trois contraintes : les segments doivent être orthogonaux, adjacents et l'un de ces segments doit être vertical.



$$\textit{pair } p \;\rightarrow\; \textit{segment } s_1, \textit{segment } s_2, \quad \langle \mathsf{orthog}(s_1, s_2),\ \mathsf{adj}(s_1, s_2),\ \mathsf{vertical}(s_1) \rangle$$

FIGURE 9.15 – Exemple 2D, avec à gauche l'ensemble des segments terminaux, et à droite les paires recherchées qui satisfont la contrainte.

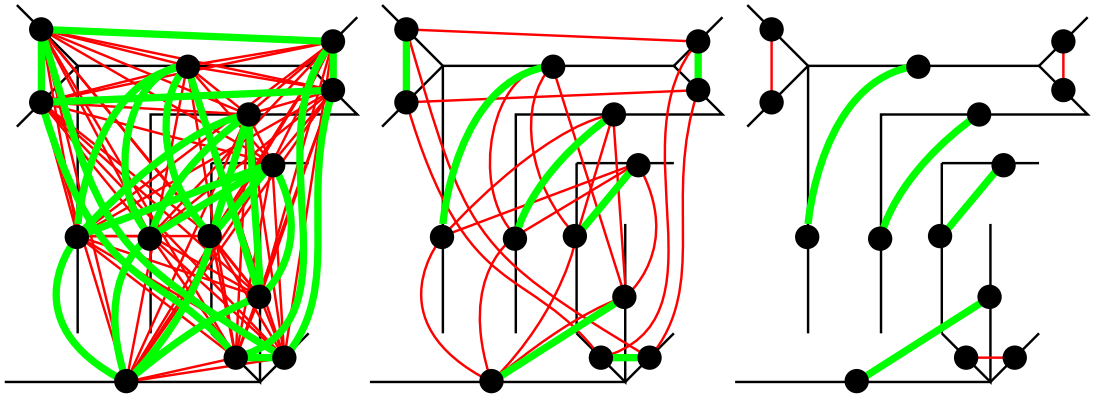
Deux ordres d'application des contraintes sont envisagés. Le premier (figure 9.16) cherche d'abord les paires orthogonales, puis vérifie l'adjacence, et enfin la verticalité. Le second (figure 9.17) sélectionne d'abord les segments verticaux avant de construire les paires adjacentes et enfin, parmi celles-ci, cherche celles qui satisfont la relation d'orthogonalité.

Sur la figure 9.16, commencer par la recherche des paires orthogonales implique de tester toutes les paires. Si N est le nombre de segments, la complexité de l'opération est $O(N^2)$. Ensuite, si le graphe d'adjacence a été maintenu, tester l'adjacence et la verticalité s'effectue en temps constant. La complexité est donc de l'ordre de $O(N^2)$.

Sur la figure 9.17, la verticalité est le premier test effectué. Cette opération nécessite un passage sur tous les segments, $O(N)$. Récupérer les paires qui sont adjacentes dépend ensuite du nombre de voisins des extrémités des segments. Si $maxDeg$ est le degré maximal des extrémités, alors la complexité devient : $O(N\ maxDeg)$. Enfin, le test d'orthogonalité laisse la complexité inchangée.

9.6.6 Ordre sur les prédicats

L'ordre d'application des contraintes est donc très important pour la performance de l'algorithme. Il faut que les combinaisons impossibles soient éliminées le plus tôt



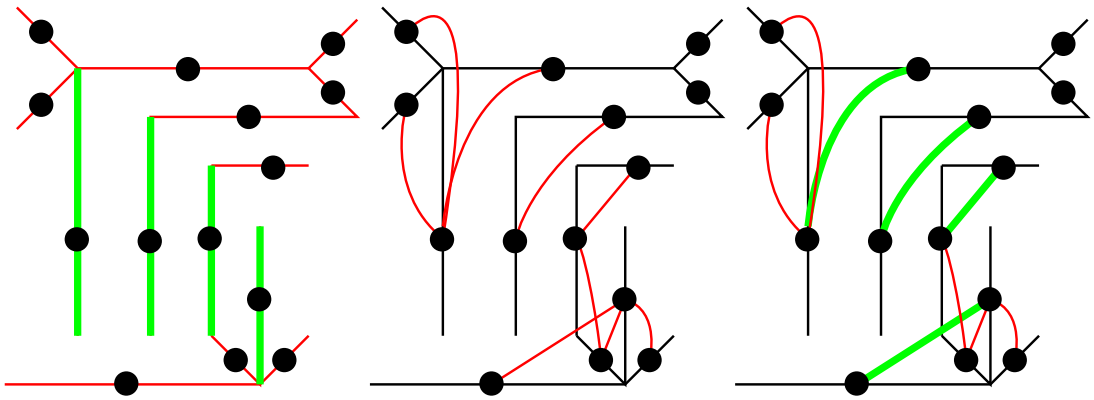
Test de toutes les paires. Complexité par paire : Complexité par paire :

$O(N^2)$ $O(1)$ $O(1)$

(a) Orthogonalité (b) Adjacence. (c) Verticalité.

Les éléments colorés, rouges et verts, représentent les éléments testés (paires ou segments). Ceux qui réussissent le test sont colorés en vert, les autres en rouge.

FIGURE 9.16 – Premier ordre d'application de contraintes, menant à une complexité en $O(N^2)$, N étant le nombre de segments.



Test de tous les segments : Complexité par segment : Complexité par paire :

$O(N)$ $O(maxDeg)$ $O(1)$

(a) Verticalité. (b) Adjacence. (c) Orthogonalité

Les éléments colorés, rouges et verts, représentent les éléments testés (paires ou segments). Ceux qui réussissent le test sont colorés en vert, les autres en rouge.

FIGURE 9.17 – Second ordre d'application de contraintes, menant à une complexité en $O(N\ maxDeg)$, N étant le nombre de segments.

possible pour limiter la taille de l'espace de recherche. Premièrement, plus un $D(x_i)$ a un cardinal faible, plus celui-ci a des chances de participer à des prédicats invalides, et donc de générer peu de nouveaux nœuds. Ensuite, les variables impliquant un grand nombre de prédicats ont plus de chances de ne pas vérifier un des prédicats que celles apparaissant dans peu de prédicats. Enfin, comme illustré par l'exemple de la section précédente, certains prédicats sont moins complexes à calculer que d'autres.

Nous allons évaluer deux degrés de complexité pour les variables de la condition $\langle C \rangle$. Puis nous combinerons ces degrés pour définir un ordre d'instanciation.

Prédicats unaires

Les prédicats unaires sont généralement simples et rapides à calculer. Ils sont directement évalués à la création des $D(x_i)$. Lors de la passe sur les éléments de F, un nœud n_i de type X_i est ajouté à $D(x_i)$ si $C_{unaire,\,x_i}(n_i)$ est remplie ($C_{unaire,\,x_i}$ étant l'ensemble des prédicats unaires de $\langle C \rangle$ agissant sur x_i).

Les prédicats unaires ayant déjà été satisfaits à la création des $D(x_i)$, le degré unaire x est le cardinale de $D(x)$:

$$deg_{unaire}(x) = |D(x)| \tag{9.9}$$

Prédicats inversibles

Cette nouvelle expression du degré peut encore être améliorée. Certains prédicats sont plus faciles à calculer que d'autres, et les espaces de recherche en jeu peuvent être réduits. Ces prédicats sont appelés prédicats *inversibles*. Plus précisément, un prédicat binaire $p(z_i, z_j)$ où z_i esr de la forme x_i ou $x_i.a$ est dit inversible si et seulement si, pour une valeur ν_i de z_i (et de même pour z_j), l'ensemble $N_j(\nu_i)$ des valeurs n_j satisfaisant $p(\nu_i, n_j)$ est petit et peut-être aisément énuméré, et de même pour $p(\nu_i, x_j.a_j)$ si z_j est de la forme $x_j.a_j$, .

N_j est utilisé pour réduire la taille de l'espace de recherche. En itérant sur les valeurs de ν_i, on construit D_j :

$$D_j \leftarrow D_j \cap \bigcup_{\nu_i} N_j(\nu_i)$$

Nous définissons le degré d'inversibilité des prédicats, pour une variable x:

$$deg_{inv}(x) = \frac{\text{nombre de prédicats inversibles sur } x \text{ dans } C}{|D(\tau(x))|} \tag{9.10}$$

Les degrés sont recalculés après chaque traitement de prédicat, car cela peut modifier le nombre d'éléments dans les D_j.

Par exemple, une primitive ou une forme n_i est généralement adjacente à un nombre limité d'autres primitives ou formes n_j. De plus l'accès au n_j est rapide dès qu'un graphe d'adjacence est précalculé. Pour une forme, calculer les formes adjacentes de type τ est effectué en récupérant les primitives qui la constituent, puis l'adjacence de celles-ci, pour enfin en remonter dans la forêt pour obtenir les formes de type τ. Pour limiter le temps de calcul, il est possible de mettre en cache les opérations effectuées afin d'éviter de les répéter.

L'adjacence est un prédicat inversible.

De même l'égalité $z_i == x_j$ (et $z_i == x_j.a$) est aussi inversible, lorsqu'on connaît z_i (hypothèse d'inversibilité).

Pour une valeur ν_i de z_i, il y a en général peu de nœuds n_j tels que $\nu_i = n_j.x_j$ (c'est aussi vrai pour $\nu_i = n_j.a$). La recherche des nœuds n_j peut être effectuée efficacement via des tables de hachage. Avant l'application des règles, une table de hachage est construite associant à chaque valeurs ν_i possible de l'argument z_i l'ensemble des nœuds n_j tels que $\nu_i = n_j$. L'utilisation d'une telle méthode n'est valide que si z_i est à valeurs discrètes, seul cas où l'égalité stricte est définie.

Autres prédicats

Les autres prédicats, comme above(,) (au-dessus) ou orthog(,) (orthogonal) ne sont pas considérés comme inversibles. Un grand nombre de primitives ou de formes peuvent se situer au-dessus d'une autre, de même pour l'orthogonalité entre primitives.

Le degré de contrainte deg_{autre} d'une variable x est :

$$deg_{autre}(x) = \frac{\text{nombre de prédicats sur } x \text{ dans } C}{|D(\tau(x))|} \tag{9.11}$$

Ordre d'instanciation des variables

Pour construire un ordre pour l'instanciation des variables, nous combinons les degrés définis précédemment. Soient x et y, nous instancions x en premier si l'une des propositions suivantes est valide :

- $deg_{inv}(x) < deg_{inv}(y)$
- $deg_{inv}(x) = deg_{inv}(y)$ et $deg_{unaire}(x) < deg_{unaire}(y)$
- $deg_{inv}(x) = deg_{inv}(y)$ et $deg_{unaire}(x) = deg_{unaire}(y)$ et $deg_{autre}(x) < deg_{autre}(y)$

Dans le cas où tous les degrés de x et de y sont égaux, c'est l'ordre d'écriture dans la grammaire qui prévaut.

Cet ordre a été établi de manière empirique, en observant que c'est le degré d'inversibilité qui tend à influencer le plus la complexité de l'algorithme.

Ordre d'application des prédicats

Les variables sont instanciées selon l'ordre défini précédemment. Pour instancier une variable x, l'ordre d'application des prédicats est le suivant :

- Les prédicats unaires, lors de la création de $D(x)$.
- Les prédicats inversibles.
- Les autres prédicats.

9.6.7 Instanciation des opérateurs maximaux

Les opérateur de collections maxcol$(X\ x, c_u, c_b)$ sont traités comme tous les terminaux et non-terminaux. La différence vient du calcul du domaine $D(x)$, qui n'est pas construit via une passe linéaire sur F.

La contrainte d'exclusivité est un problème majeur ici. Dans le cas général, l'ensemble des collections maximales *valides* (au sens de la contrainte d'exclusivité) ne peut pas être aisément déduit de l'ensemble des collections maximales.

La figure 9.18 illustre cette difficulté. La règle de la grammaire est la création d'un escalier défini comme une séquence maximale de blocs-marche adjacents à une contre-marche. Cependant, deux lignes orthogonales ont deux interprétations : soit un bloc-marche (s_1), soit une contre-marche (r_1) et un plancher. La création de la séquence maximale de blocs-marche (s_1, s_2, s_3, s_4), sans tenir compte de la seconde interprétation, conduit à l'impossibilité de construire l'escalier car une contre-marche supplémentaire va manquer. La séquence maximale souhaitée (s_2, s_3, s_4) n'est maximale que si r_1 est pris en compte séparément.

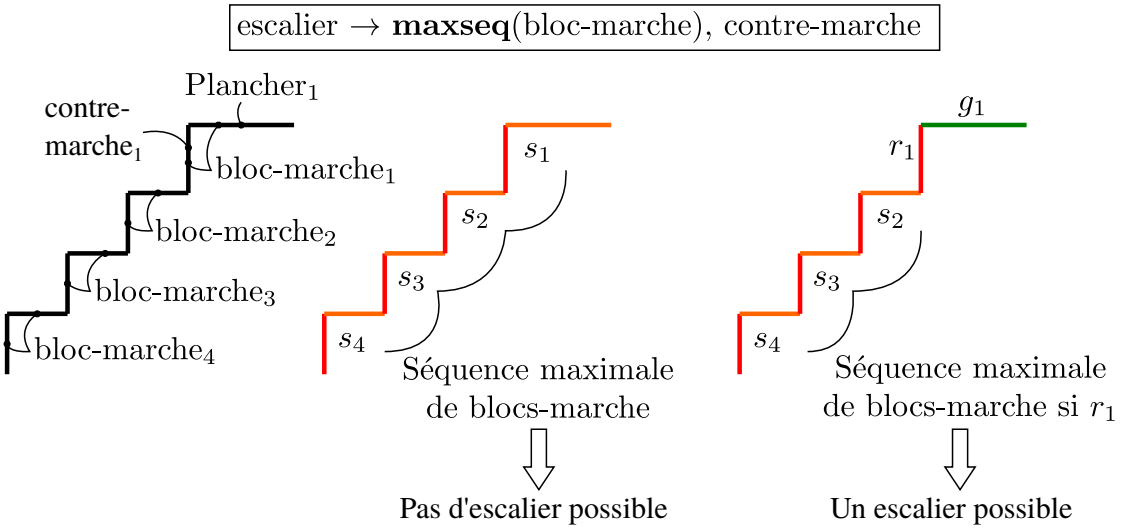


FIGURE 9.18 – Exemple où la collection maximale valide est difficilement déductible de l'ensemble des collections maximales.

L'idée retenue ici est d'approcher l'ensemble des collections valides en garantissant que les éléments retenus sont valides, c.-à-d. en construisant un sous-ensemble de collections valides.

Pour produire $D(x_i)$ dans le cas d'un opérateur sans variable de contexte, le graphe $G = (V, E)$ est construit de la manière suivante :

V, l'ensemble des sommets de G, est l'ensemble des nœuds de F vérifiant les contraintes unaires :

$$V = \{n \in F | \tau(n) = X,\ c_u(n) = vrai\}$$

E, l'ensemble des arêtes de G, est l'ensemble des paires $(n, n') \in V^2$ vérifiant les contraintes binaires, et exclusives entre elles :

$$E = \left\{(n, n') \in V^2 | c_b(n, n') = vrai,\ excl(n, n') = vrai\right\}$$

où $excl(n, n')$ est un prédicat qui représente la vérification de la contrainte d'exclusivité entre n et n'.

Dans le cas maxconn et maxset, les arêtes de G sont non orientées. Dans les autres cas, on préserve l'orientation.

Les collections maximales sont ensuite calculées en utilisant les algorithmes décrits dans la table 9.2. Pour le maxset, l'ensemble calculé est exhaustif, car il n'y a pas d'arête entre deux nœuds non-exclusifs ; ce n'est pas le cas pour les autres opérateurs. Pour maxseq et maxconn la contrainte d'exclusivité est vérifiée à la construction de la collection, pour cycle, après construction. Dans le pire des cas, malgré les opérateurs maximaux, le nombre de collections peut-être exponentiel. En pratique, les graphes ont un degré de connectivité faible, et les énumérations sont rapides.

Par exemple pour la construction d'un escalier, une séquence maximale de blocs-marche est théoriquement construite en temps quasi-linéaire en partant de n'importe quel bloc-marche et en faisant deux croissances exclusives jusqu'à ce qu'aucun bloc-marche ne puisse être ajouté. Lorsqu'ils sont considérés pour générer une nouvelle séquence, les blocs-marche qui appartiennent à cette séquence maximale sont mémorisés de sorte qu'aucune nouvelle croissance n'est effectuée car la séquence a déjà été calculée.

Operateur	Collections	Algorithme
maxseq	Chemins acycliques maximaux	Recherche en profondeur d'abord (DFS) pour trouver les sommets extrémaux, puis DFS pour trouver les chemins
cycle	Cycles élémentaires	Tarjan
maxset	Cliques maximales	Bron-Kerbosch
maxconn	Composantes connexes	DFS (le graphe étant non-orienté dans ce cas)

TABLE 9.2 – Opérateurs de collections maximales et algorithmes pour calculer les instances de ceux-ci.

Opérateurs dans une règle avec des variables de contexte

Lorsque des variables de contexte sont utilisées dans les conditions d'une règle contenant un opérateur maximal, des problèmes de complexité peuvent apparaître (découverte a posteriori de nouvelles variables pouvant servir de contexte). Pour traiter ce cas, les variables contextuelles sont toutes instanciées avant de considérer l'opérateur. Ceci exclut de fait les règles avec des dépendances circulaires entre les opérateurs mais, en pratique, ce type de règle n'a jamais été requis dans nos expériences.

Les opérateurs avec variables de contexte sont même en général plus facile à calculer. Par exemple, pour un escalier dont on impose une extrémité adjacente à un sol :

$$Stairway \rightarrow \mathsf{maxseq}(Step)\ ms, \mathsf{context}(Ground)\ g,\ \langle \mathsf{adj}(ms, g) \rangle$$

il est plus efficace de partir du sol et d'étendre aux marches que de partir au hasard de ce qui ressemble à une marche, de créer la séquence et enfin de vérifier l'adjacence à un sol.

9.7 Expériences

Les fonctions précédentes ont été implémentées au sein d'un outil d'analyse à l'exception du « sucre syntaxique ». Cet outil permet à l'utilisateur de spécifier la grammaire d'une manière simple.

Cette section présente les résultats obtenus sur des modèles CAO (conception assistée par ordinateur) mais aussi sur des scènes réalistes acquises par laser et par photogrammétrie.

Dans toutes les expériences, les terminaux sont des surfaces plutôt que des volumes. Cependant, l'approche proposée n'est pas restreinte au seul enrichissement sémantique de surfaces.

9.7.1 Modèles CAO

5 modèles de bâtiments ont été utilisés pour les expériences. Ces maquettes BIM (Building Information models) ont été créées avec des outils de CAO (Conception Assistée par Ordinateur) au format IFC qui est un standard pour le BIM. L'utilisation de IfcObj [site] a permis d'en extraire les informations purement géométriques, c'est-à-dire une soupe de triangles. Ces triangles ont été fusionnés pour créer un ensemble de polygones à trous et le graphe d'adjacence correspondant. Les modèles, ainsi que les informations de tailles correspondantes, sont listés dans la table 9.3.

Nom	# de triangles	# de polygones
LcG	48332	9705
LcA	111979	26585
LcC	385541	111732
LcD	313012	75257
LcF	286996	84347

TABLE 9.3 – Modèles : nombre de triangles et de polygones.

Création des polygones

La construction des polygones sur des modèles idéaux est en théorie simple. Il suffit de fusionner les triangles coplanaires partageant une arête.

En pratique, bien que les modèles aient été créés via des outils de CAO, nous avons constaté qu'ils n'étaient pas bien triangulés. En effet, deux triangles peuvent être adjacents (par une arête ou par la face) sans pour autant avoir un sommet en commun. A ceci s'ajoute le fait que les triangles n'étaient pas exactement coplanaires. Pour cette raison nous construisons ici un polygone comme un ensemble de triangles quasiment coplanaires (différence angulaire inférieure à 3°). Ceci est dû au mode de création de ces modèles. Les éléments sont généralement conçus séparément, puis assemblés, empilés et répétés, produisant ainsi un ensemble de maillages plutôt qu'un unique maillage rendant compte de la surface.

Créer les polygones de manière robuste nécessite, en pratique, de fusionner des triangles qui peuvent se recouvrir les uns les autres. Il faut de plus utiliser une adjacence approchée (à une certaine distance) par opposition à une adjacence exacte (partage de sommets et d'arêtes).

Le processus de création des polygones est le suivant.

- Un graphe d'adjacence approchée est calculé de la manière suivante. L'ensemble des triangles du modèle est inséré dans un arbre AABB [Alliez et al., 2010] pour le calcul des intersections. Ensuite, pour chaque triangle T, une boîte englobante est construite. Les triangles de cette boîte servent de primitives de test d'intersection dans l'arbre, les 8 triangles intersectés représentant l'adjacence de T. Lorsque deux sommets d'un triangle sont inclus dans la boîte, il y a adjacence via une arête (figure 9.19)

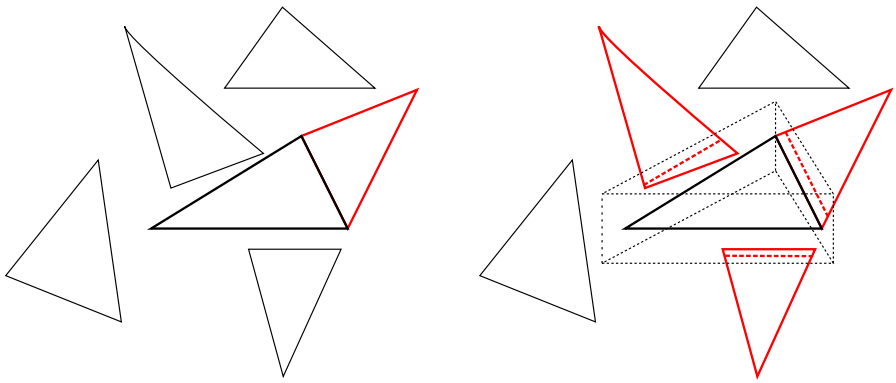
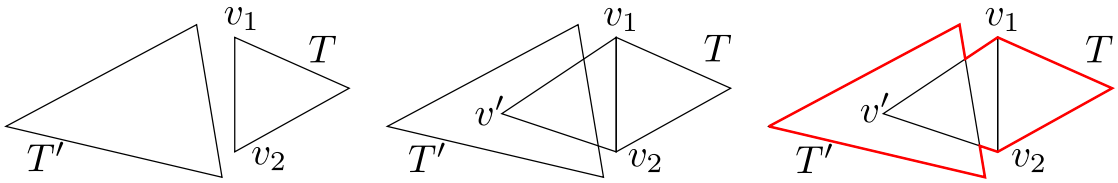


FIGURE 9.19 – Adjacence exacte (à gauche), et approchée (à droite). Les triangles rouges sont considérés comme adjacents au triangle noir.

- Des ensembles de triangles sont alors construits à la manière d'une croissance de régions. Initialisant l'ensemble avec un triangle T, les triangles du voisinage de celui-ci sont ajoutés si l'angle entre leur normale et celle de T est faible (inférieur à 3°).

- Le polygone est construit en projetant sur le plan de régression tous les triangles et en fusionnant ces triangles avec la classe 2D-polygon de CGAL. Il est nécessaire d'éviter la création de plusieurs composantes connexes lors de la fusion de deux triangles T et T', adjacents via l'arête (v_1, v_2). Pour ce faire, le triangle (v', v_1, v_2), où v' est le centre de gravité de T', est ajouté. Ce triangle permet de lier les composantes connexes (figure 9.20).



Deux triangles disjoints (à gauche), le triangle (v', v_1, v_2) est ajouté pour les lier (au centre) et le polygone résultant à droite.

FIGURE 9.20 – Création de polygones pour des triangles disjoints.

De meilleures heuristiques de « réparation »de maquettes sont sans doute possibles, mais cette solution simple s'est avérée suffisante pour les besoins de la validation de la méthode d'analyse sémantique présentée dans ce chapitre.

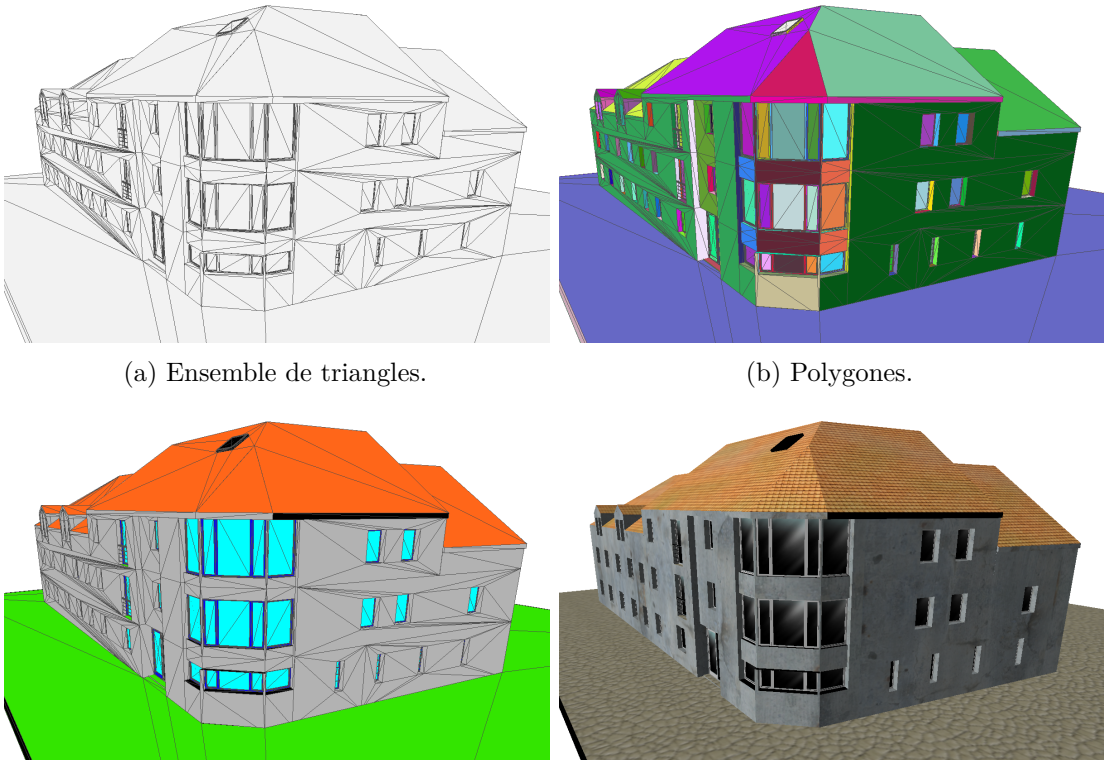
Les figures 9.21b et 9.22 (seconde colonne) montrent les polygones créés à partir des ensembles des triangles des modèles CAO (une couleur différente par polygone).

Cette étape de « réparation »est optionnelle car les attributs utiles pour les polygones sont calculables directement sur l'ensemble des triangles seulement. De plus nous considérons qu'attribuer une étiquette à un polygone est équivalent à étiqueter les triangles qui le composent.

Précision de l'analyse

L'outil d'analyse sémantique (le parseur) que nous avons développé est un outil spécifique que nous avons expérimenté pour l'analyse de bâtiments. Nous l'avons nourri avec des grammaires, l'une pour l'analyse des escaliers, l'autre pour les autres éléments constituant le bâtiment : murs, sols, plafonds et ouvertures (portes et fenêtres). Les figures 9.21 et 9.22 présentent les résultats de l'analyse pour la détection des ouvertures avec la grammaire de la figure 9.26. Le code couleur est donné par la table 9.4. Les escaliers retrouvés dans les différentes scènes sont, quant à eux, illustrés par la figure 9.23. Ces escaliers sont correctement détectés avec la grammaire présentée figure 9.25. La grammaire n'est pas modifiée suivant les modèles malgré la présence ou non de nez de marche et ce, même lorsque les escaliers forment un coude. La figure 9.24 présente trois types de bloc-marches gérés par la grammaire. Le troisième est spécifique aux modèles CAO, et reflète le cas où la marche a été construite à l'aide d'un parallélépipède posé sur une contre-marche.

Le tableau 9.5 fournit une évaluation chiffrée pour les escaliers et les ouvertures pour les différents modèles utilisés. D'après ce tableau, comme visuellement d'après les illustrations, la méthode permet de reconnaître correctement une grande majorité des



(a) Ensemble de triangles.

(b) Polygones.

(c) Résultat après analyse.

(d) Résultat avec texture.

FIGURE 9.21 – Résultat de l'analyse pour la recherche d'ouvertures, de murs, de toits... pour le modèle LcG.

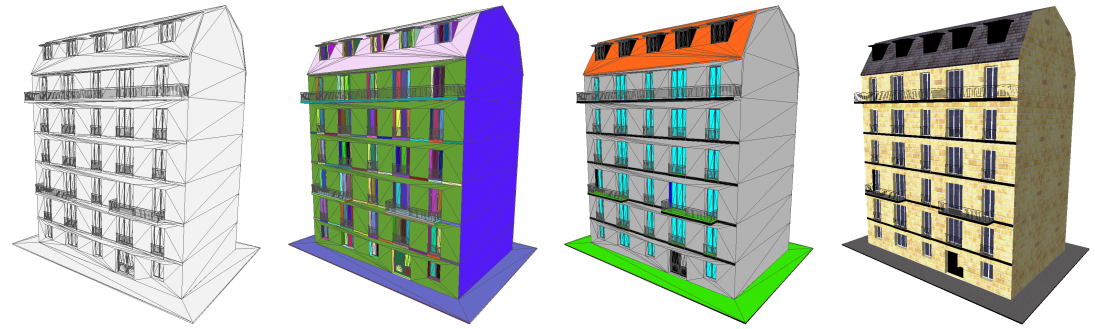
Élément	Couleur	Élément	Couleur
Mur	gris	Ouverture	cyan, bleu
Toit	terracotta	Escalier	rouge, orange
Sol	vert, marron	Non assigné	noir

TABLE 9.4 – Couleurs pour les éléments sémantisés.

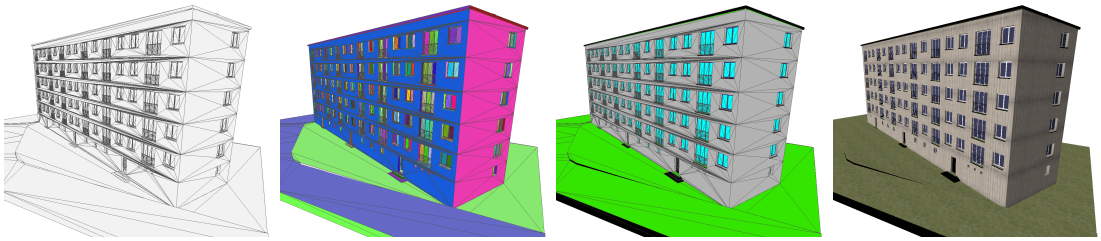
éléments. Le taux de faux positifs (mauvaises détections) comme le taux de faux négatifs (détections manquées) est faible. Le premier s'explique généralement par une détection trop étendue, comme par exemple un escalier auquel a été ajouté une contre-marche non souhaitée. Les détections manquées, quant à elles, s'expliquent majoritairement par des défauts dans la géométrie et/ou des erreurs dans la création des polygones.

En effet, la géométrie des modèles CAO reflète souvent leur processus de création. Les éléments sont généralement créés par blocs et répétés avec des ajustements de qualité variable pour former le bâtiment. De ce fait, les volumes peuvent s'intersecter, créant de fausses adjacences entre triangles (figure 9.27a). À l'inverse, des surfaces théoriquement adjacentes peuvent se retrouver disjointes (figure 9.27b). Des éléments de taille anormale sont aussi présents dans les scènes. C'est par exemple le cas de la figure 9.27 où un escalier intersecte le sol. La première contre-marche est beaucoup plus haute que la normale, la rendant impossible à détecter.

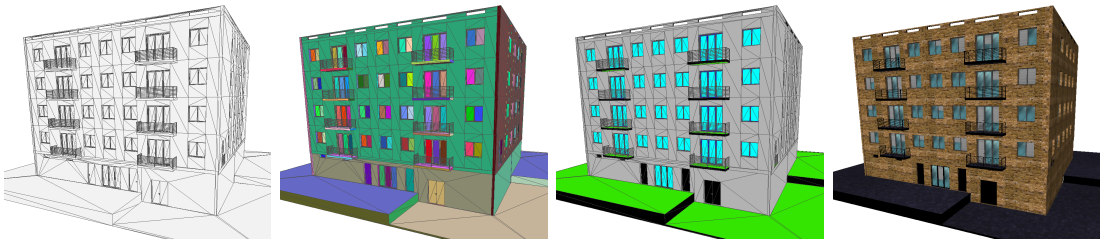
De plus, lors de la création des polygones, les triangles adjacents sont fusionnés s'ils sont approximativement coplanaires. Or, comme l'illustre la figure 9.28, des objets



(a) Modèle LcA.



(b) Modèle LcC.



(c) Modèle LcD.

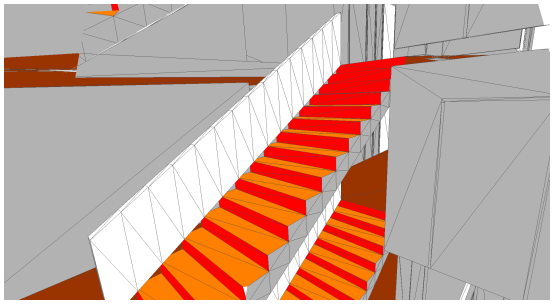
De gauche à droite, les triangles d'entrée, les polygones créés, le modèle après analyse et le résultat avec texture.

FIGURE 9.22 – Résultat de l'analyse pour la recherche d'éléments de construction.

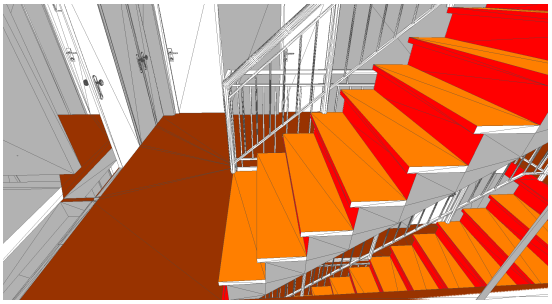
différents peuvent être inclus dans un même polygone ; ici, une porte dont le cadre n'a pas d'épaisseur significative et un mur forment un même polygone.

Enfin, certaines ouvertures manquées sont dues à des limitations de la grammaire. Par exemple, dans le modèle LcA, les fenêtres du toit sont adjacentes à de petits morceaux de murs, trop petits pour être considérés comme tels par la grammaire. Or dans notre grammaire, une ouverture est verticale et adjacente à un mur. De même sur LcG, les fenêtres du toit sont obliques, rendant ainsi impossible la détection de ces ouvertures par la grammaire telle qu'elle avait été écrite a priori.

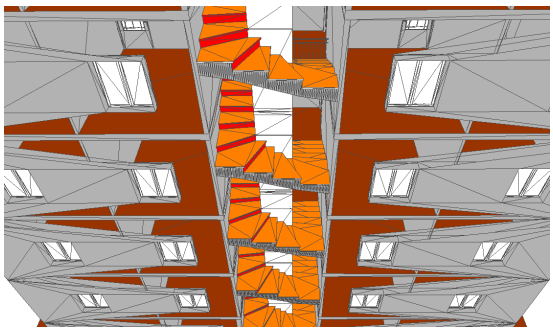
Il est à noter que la grammaire ne fait pas la différence entre les fenêtres et les portes mais les désigne par le terme générique d'ouverture. Ce regroupement sémantique existe déjà dans le format IFC. Mais, ici, ce choix vient du fait que les modèles utilisés ont été réalisés par des personnes différentes, avec des conventions différentes (figure 9.29). Des portes vitrées sont décrites comme fenêtres dans certains modèles et comme portes dans d'autres ce qui fausse complètement les scores de détection.

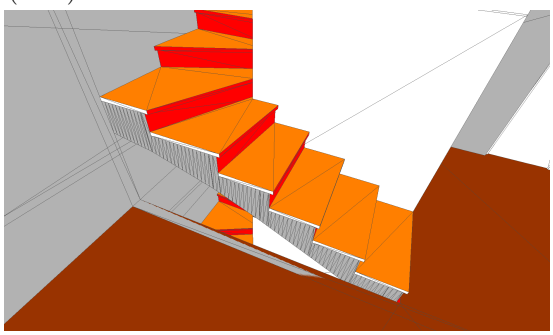


(a) Escaliers linéaires (LcG).



(b) Escaliers linéaires avec des nez de marche (LcD).





(c) Escaliers avec nez de marche et rotation (LcA).

FIGURE 9.23 – Résultats de l'analyse pour la recherche d'escaliers.

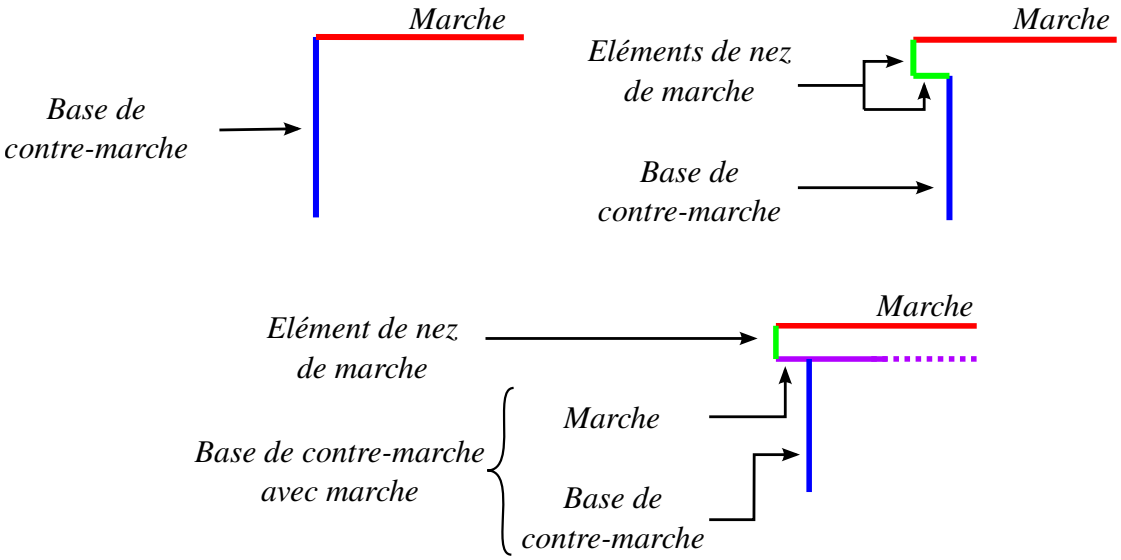


FIGURE 9.24 – Trois types de bases de contre-marche et leur décomposition en éléments de grammaire.

Temps de calcul

Les temps de calcul sont donnés par le tableau 9.6. Il faut noter que l'implémentation relève du prototype est n'a pas été optimisée pour accroître les performances. Comme les résultats présentés précédemment, et afin de rendre compte des temps de calcul liés à chaque partie de l'analyse, la grammaire a été divisée en deux parties : l'une pour la reconnaissance des ouvertures, l'autre pour la recherche des escaliers. Les temps présentés ici sont ceux de l'analyse. Ils ne tiennent pas compte du temps de création des polygones.

```
// Une contre-marche est un polygone vertical dont la taille est contrainte.
        riserBase b  →  Polygon p
                            ⟨ vertical(p),  0.1 <= p.breadth <= 0.3,
                              0.5 <= p.length <= 2 ⟩

// Un élément de nez de marche est un polygone fin,
// dont la direction de longueur est horizontale.
    nosingElement e  →  Polygon p
                            ⟨ horizontal(p.lengthVector),  p.breadth <= 0.08,
                              0.5 <= p.length <= 2 ⟩

// Une contre-marche avec un nez optionnel est une contre-marche
// coiffée d'une séquence possiblement vide d'élément de nez de marche
// (voir figure 9.24).
   riserOptNose ron  →  riserBase b,  maxseq(nosingElement, edgeAdj) n
                            ⟨ edgeAdj(b, n),  above(b, n) ⟩

// Une marche est un polygone horizontal dont la taille est contrainte.
              tread t  →  Polygon p
                            ⟨ horizontal(p),  0.1 <= p.breadth <= 2,
                              0.5 <= p.length <= 2 ⟩

// Une contre-marche coiffée d'une marche et d'un nez de marche (see figure 9.24).
// Ceci est utile pour les modèles CAO.
   riserBotTrBase bt  →  riserBase b,  tread t
                            ⟨ adj(b, t),  above(t, b) ⟩
riserBotTrNose rbtn  →  riserBotTrBase bt,  maxseq(nosingElement, edgeAdj) n
                            ⟨ 1 <= n.size,  edgeAdj(bt, n),  above(n, bt) ⟩

// Une contre-marche est de n'importe quel type de contre-marche.
              riser r  →  riserOptNose ron  |  riserBotTrNose rbtn


// Un bloc-marche est une contre-marche recouvert d'une marche.
               step s  →  riser r,  tread t
                            ⟨ edgeAdj(r, t),  above(t, r) ⟩

// Un escalier est une séquence de marche associé à une contre-marche optionnelle.
        stairway w  →  maxseq(step, edgeAdj) s,  optional(riser) r
                            ⟨ edgeAdj(s, r),  above(r, s) ⟩
```

FIGURE 9.25 – Grammaire pour les escaliers (unités en mètres)

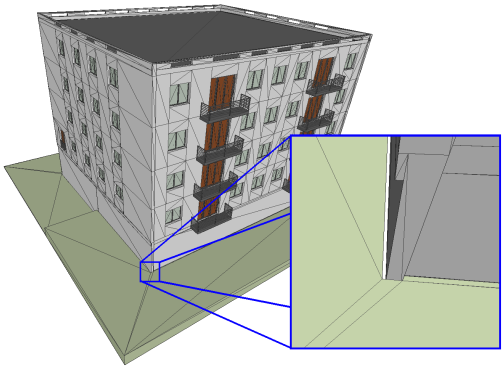
Modèle	nb d'escaliers	nb de marches	Escaliers (%) Préc.	Rapp.	Ouvertures #	Préc.	Rapp.
LcG	3	45	100	93	83	100	90
LcA	6	84	100	100	62	98	83
LcC	30	210	100	100	196	100	98
LcD	5	61	93	100	74	100	93
LcF	7	98	100	50	99	100	96

TABLE 9.5 – Évaluation de l'étiquetage des escaliers et des ouvertures : nombre d'éléments (nb), précision (Préc., %) et rappel (Rapp., %).

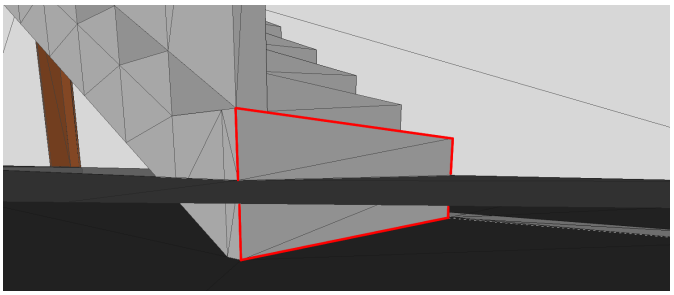
```
// Un plancher est grand polygone horizontal.
            floor f  →  Polygon p
                            ⟨ horizontal(p),
                              3 <= p.breadth, 3 <= p.length, 7 <= p.area ⟩
// Un panneau mural est un polygone vertical haut et large,
// adjacent à deux planchers.
      main_wall w  →  Polygon p, context(floor) f1, context(floor) f2
                            ⟨ vertical(p), 1 <= p.bbz, 0.4 <= p.breadth,
                              adj(p, f1), adj(p, f2) ⟩
// Une arête de mur (un petit mur autour des ouvertures) est un polygone fin
// et sans trou, ajdacent à deux murs (intérieur et extérieur).
      wall_edge we  →  Polygon p, context(main_wall) w1, context(main_wall) w2
                            ⟨ horizontal(p) or vertical(p), p.length <= 2.7,
                              0.05 <= p.breadth <= 0.5, p.nbHoles == 0,
                              adj(p, w1), adj(p, w2) ⟩
// Une pré-ouverture (une ouverture potentielle) est un polygone
// de la taille d'une ouverture.
   pre_opening po  →  Polygon p
                            ⟨ vertical(p), 0.5 <= p.bbz,
                              0.2 <= p.length <= 2.7, 0.2 <= p.breadth <= 2.7 ⟩
// Un pré-cadre (un cadre d'ouverture potentiel) est une pré-ouverture avec un
// ou plus grand trou, ou aucun trou et une aire faible comparée à l'aire de son
// rectangle englobant (un cadre de porte par exemple).
      pre_frame pf  →  pre_opening po ⟨ po.nbHoles >= 1,
                              po.area <= 0.5 * po.length * po.breadth ⟩
                     |  pre_opening po ⟨ po.nbHoles == 0,
                              po.area <= 0.25 * po.length * po.breadth ⟩
// Un panneau ( vitre ou porte) est une pré-ouverture sans trou,
// avec une aire similaire à celle de son rectangle englobant.
          panel pn  →  pre_opening po ⟨ po.nbHoles == 0,
                           0.95 * po.area <= po.length * po.breadth <= 1.05 * po.area ⟩
// Une arête de cadre est un polygone fin de longueur limitée à celle de l'ouverture.
      frame_edge fe  →  Polygon p ⟨ horizontal(p) or vertical(p),
                              horizontal(p.lengthVector) or vertical(p.lengthVector),
                              p.nbHoles == 0,
                              p.length <= 2.7, 0.05 <= p.breadth <= 0.5 ⟩
// Un élément d'ouverture est un panneau, une pré-ouverture ou une arête de cadre.
opening_element oe  →  panel pn | pre_frame pf | frame_edge fe
// Un cadre est un pré-cadre adjacent à une arête de mur.
          frame f  →  pre_frame pf, context( )wall_edge we ⟨ adj(pf, we) ⟩
// Une ouverture est un ensemble connecté d'éléments d'ouverture adjacent à un
// mur, implémentée comme un ensemble d'éléments d'ouverture basé sur un cadre.
        opening o  →  frame f, maxconn(opening_element)soe, ⟨ adj(soe, f) ⟩
```

FIGURE 9.26 – Grammaire pour les ouvertures (unités en mètres)

(a) Porte intersectant le sol (passant en fait dans la dalle).



(b) Non adjacence entre le sol et le bâtiment.



(c) Contre-marche de taille anormale pénétrant dans le sol.

FIGURE 9.27 – Exemples de problèmes de géométrie dans les modèles CAO.

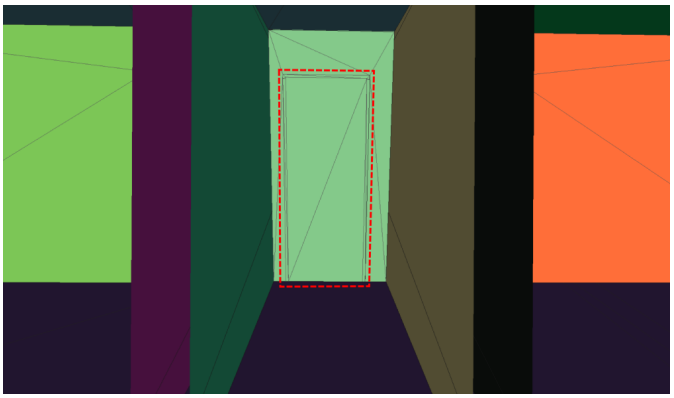


FIGURE 9.28 – Une porte et un mur ayant fusionné à la création des polygones.

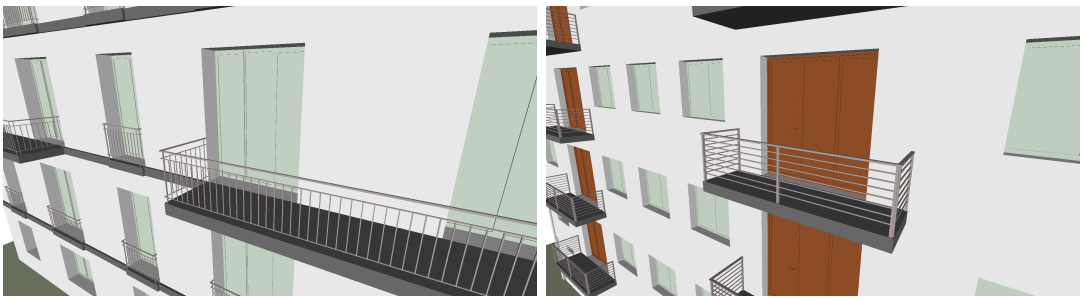


FIGURE 9.29 – Exemple d'ouvertures, géométriquement identiques, mais annotées différemment dans le modèle IFC original.

Nom	nb de escaliers	nb de marches	Temps d'analyse (s)	nb de ouvertures	Temps d'analyse (s)
LcG	3	45	5	83	15
LcA	6	84	14	62	42
LcC	30	210	33	196	306
LcD	5	61	25	74	111
LcF	7	98	39	99	322

TABLE 9.6 – Temps d'analyse (sans la création des polygones).

On observe grâce au bâtiment LcC, notablement plus gros que les autres, que les performances asymptotiques sont bonnes. L'un des points clés de cette étude est que, malgré l'approche bottom-up, le temps de calcul n'est pas exponentiel en le nombre de polygones ou d'objets effectivement présents dans la scène. Ce résultat est obtenu grâce aux opérateurs maximaux.

Le comportement théoriquement quasi-linéaire de l'extraction des séquences maximales est en réalité quadratique dans l'implémentation actuelle, qui n'a pas été optimisée. Cela peut se deviner dans la table 9.6 où les temps de calculs sont mis en perspective du nombre d'éléments à rechercher dans les modèles. Pour le modèle le plus complexe, LcC, une approche exponentielle ne serait pas capable de retrouver les 210 marches et de les regrouper en 30 escaliers différents en seulement 33 secondes. En pratique, le temps d'analyse est surtout lié à la rigidité des contraintes imposées dans les règles, et au degré moyen d'un sommet dans le graphe d'adjacence. Le nombre de règles et le nombre de variables présentes dans cette règle influent peu. Une explosion combinatoire est cependant théoriquement possible, mais elle ne s'est jamais produite dans les modèles testés.

Pour accélérer l'algorithme, il serait possible de paralléliser les applications des règles. En effet, l'application des règles est actuellement effectuée séquentiellement, mais lorsque deux règles sont indépendantes, il est possible de les effectuer simultanément en parallèle. L'ordre des règles doit cependant être respecté pour éviter la création d'éléments invalides (cf. section 9.6). Par exemple, les recherches des murs et des escaliers sont indépendantes, mais ce n'est pas le cas de la recherche des escaliers et des sols. Un escalier devant être adjacent à un sol, les sols doivent être découverts avant.

9.7.2 Nuages de points

La méthode a été évaluée sur deux types de nuages de points : d'un côté des points laser et de l'autre des données photogrammétriques.

Nuages de points laser

Nuage de points seul. La figure 9.30 présente le résultat de la reconnaissance d'escaliers dans une scène laser simulée. Après estimation des normales (partie I), des segments sont produits par croissance de régions dans l'image de profondeur. Les polygones considérés sont les rectangles englobants des points dans le plan de régression du segment. Comme on peut l'observer sur la figure, les marches visibles sont correctement extraites. Cependant l'escalier complet n'est pas extrait. En effet l'algorithme ne crée pas les

marches dont il manque les plateaux (invisibles depuis le laser), et par conséquent, ne peut étendre l'escalier.

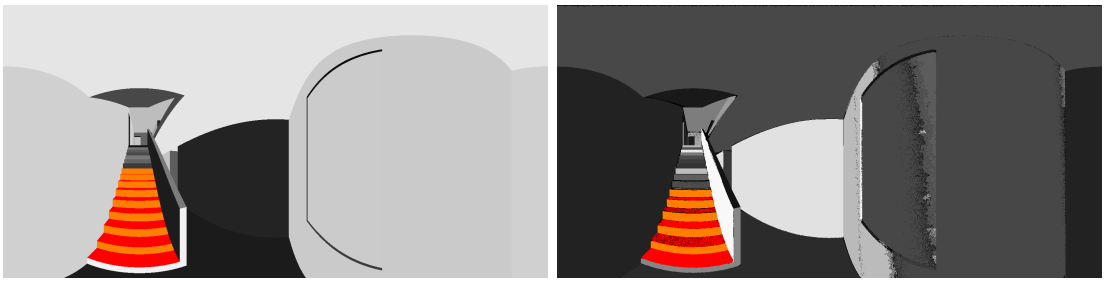


FIGURE 9.30 – Détection d'escalier dans un nuage laser simulé, sans bruit dans les données (à gauche) et avec du bruit gaussien (écart type de 0.5 cm)(à droite).

Surface reconstruite par le processus de reconstruction. La figure 9.31 présente le résultat de la détection d'escaliers dans une scène laser préalablement reconstruite à partir de l'algorithme de la partie III. La complétion des surfaces manquantes dans les zones invisibles permet de détecter correctement la majorité des blocs-marches, y compris lorsque les marches de ceux-ci n'ont pas été échantillonnées par le laser. L'escalier est cependant coupé en deux. La non détection du bloc-marche central est due à la contremarche, dont le polygone, qui inclut l'arête de la rampe d'escalier, a une hauteur largement supérieure à celle d'une contre-marche.

Photogrammétrie

Le nuage de points photogrammétriques utilisé est une reconstruction multi-vues de la cour d'un château [Vu et al., 2012]. Les photos appartiennent au jeu de données de Strecha et al. [2008]. Ce type de données est le « pire scénario »envisagé : le bruit présent dans les données est de l'ordre de 5 à 10 cm, ce qui est considérable en comparaison de la largeur des bords de fenêtres. Cependant, le maillage 3D est assez précis pour détecter certains de ces bords.

Le nuage de points est d'abord segmenté en zones planaires selon la méthode de Schnabel et al. [2007], puis, pour chaque segment, l'α-shape est calculée dans le plan de régression. 1817 polygones sont ainsi obtenus (figure 9.32, à gauche). Certains polygones ne sont pas trouvés lors de cette étape, comme les bords de fenêtres (primitives supportant peu de points), d'autres au contraire sont coupés en plusieurs morceaux, produisant une

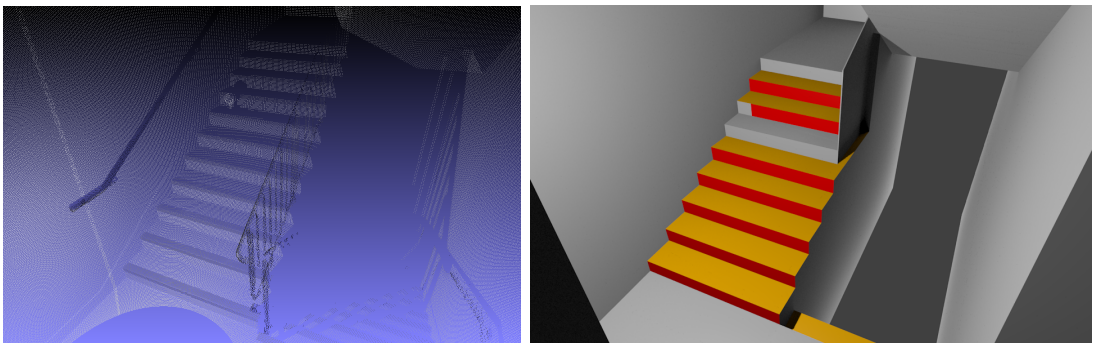


FIGURE 9.31 – Détection d'escaliers dans une scène laser après reconstruction de la surface.

sur-segmentation et une mauvaise estimation des propriétés des objets. Enfin un graphe d'adjacence approché est calculé.

La grammaire pour la détection des ouvertures n'est pas directement applicable à ce type de données. Celle-ci est relâchée pour s'accommoder du bruit et des données manquantes. Une tolérance plus grande est utilisée pour l'orientation des polygones (verticalité, horizontalité) et sur la taille des éléments. Une fenêtre est désormais une vitre adjacente à une séquence d'au moins deux petits éléments, et non plus à un cadre complet comme précédemment. Cette relaxation est heuristique et appelle à la création d'une méthodologie pour décider sur quels paramètres doit agir la relaxation.

Sur la figure 9.32, à droite, on observe que les murs et les ouvertures sont raisonnablement bien détectés, 22 des 31 fenêtres de la façades sont reconnues. Ces résultats sont satisfaisants compte tenu du haut niveau de bruit de la scène. Il faut de plus noter que même en 2D, la détection de fenêtres n'est pas un problème considéré comme résolu [Simon et al., 2012; Teboul et al., 2011].

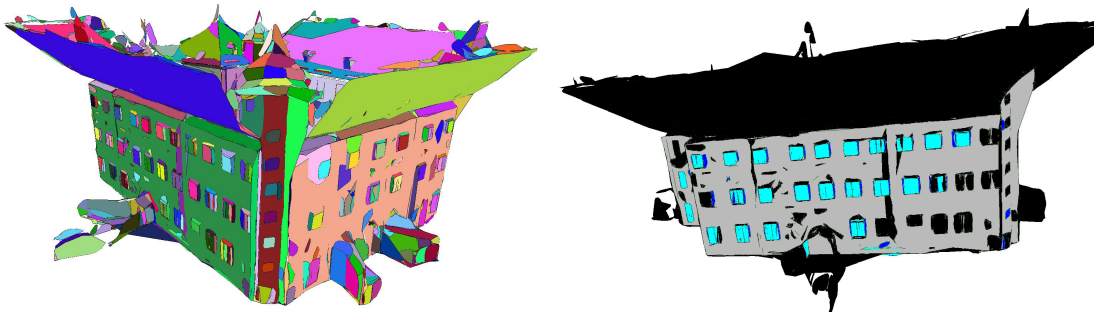


FIGURE 9.32 – Polygones (à gauche) et résultat de la détection des ouvertures (à droite) pour une reconstruction photogrammétrique [Strecha et al., 2008].

9.8 Discussion

9.8.1 Bruit et mauvaises détections

Les résultats présentés sur les modèles CAO sont encourageants tant du point de vue de la précision que de la complexité pratique de l'algorithme. Cependant, en l'état, l'utilisation de l'algorithme présenté dans ce chapitre montre ses limites dès que des primitives sont manquantes ou que le bruit est particulièrement important.

Du fait du bruit, certaines primitives peuvent être mal estimées (mauvaise orientation, mauvaise forme...), coupées en plusieurs composantes ou même non détectées. C'est aussi le cas avec les données certes précises mais souvent incomplètes des acquisitions laser où les occlusions empêchent la détection de certaines primitives. Il est nécessaire de rendre l'approche plus robuste. Une possibilité serait d'introduire plusieurs détections pour les mêmes triangles ou points, associées à un niveau de confiance ou une probabilité d'existence. Cette notion n'entre pas en conflit avec les contraintes d'exclusivité : une seule de ces détections pourrait participer à la création d'une forme. Ce serait aussi compatible avec les primitives coupées en plusieurs morceaux, ces morceaux pouvant être interprétés soit séparément soit conjointement (fusion des primitives).

De plus, actuellement, une primitive fausse produit en cascade des détections partielles ou mauvaises dans les arbres d'analyses (cf. figure 9.31). Plutôt que des règles absolues sur la composition des éléments, il faudra être capable de relâcher les contraintes pour

autoriser une correspondance approximative entre la forme idéale (celle décrite pas la règle) et la forme créée.

Le problème des portes fusionnées avec les murs, ou plus généralement de la sous-segmentation, pourrait être résolu en permettant de couper les formes en plusieurs éléments alternatifs. La division pourrait s'effectuer à la détection des primitives en couplant, par exemple, les informations 3D avec des informations photométriques (par exemple couleur de la porte différente de celle du mur).

Les détections alternatives comme la relaxation des contraintes se traduisent par un espace de recherche beaucoup plus grand, et donc un temps de calcul et une consommation de mémoire plus importante. Une idée pour pallier à ce problème est de prendre en compte la vraisemblance des formes créées et les distances des instanciations d'objets à leur forme idéale. Ainsi, les nouvelles instanciations se verraient associées à une probabilité d'existence. La forêt d'analyse serait alors décimée à la volée en fonction de ce score, permettant ainsi d'explorer un espace très grand, tout en gardant une forêt de taille raisonnable.

9.8.2 Formes manquantes

Notre implémentation actuelle suppose une géométrie de bonne qualité en entrée. Les éléments manquants dégradent le résultat de l'analyse. Il est nécessaire de coupler l'approche proposée avec un système de traitement des informations partielles et manquantes. Cette fonction est aujourd'hui assurée, en amont, par la méthode de reconstruction de surfaces de la partie III qui complète les surfaces invisibles. Il est aussi possible d'inclure cette fonctionnalité au niveau de l'analyse sémantique.

Le fait que la structure et les régularités soient encodées formellement dans les grammaires est ici encore une de leur force. Pour compléter les éléments incomplets ou créer des éléments manquants, il est plus aisé et plus général de raisonner sur la base d'une grammaire que sur la géométrie seule [Han and Zhu, 2009]. Pour « halluciner »de nouvelles instances d'éléments lors de l'analyse, il est nécessaire de créer des contraintes globales et générales exprimant la répétition ou la symétrie [Pauly et al., 2008]. Cependant la question de savoir où, quand et comment doivent être exprimées ces contraintes reste en suspens. Faut-il les encoder comme des opérateurs explicites ou comme des méta-règles influant sur l'ensemble des règles de la grammaire ?

En pratique, sans chercher à compléter géométriquement le modèle de départ, formuler la structure et la régularité des modèles, comme le permettent les grammaires, est crucial pour l'analyse sémantique. Avec des données réelles (bruitées, incomplètes), la structure est essentielle pour prévenir les fausses détections et halluciner les détections manquantes. Par exemple, une grille d'alignement est nécessaire pour retrouver les fenêtres manquantes du modèle de la figure 9.32. Même sur les données CAO, de mauvaises marches sont détectées dans l'ensemble des polygones, et la régularité est nécessaire pour composer des escaliers valides, rejetant ainsi les mauvaises détections.

Enfin l'avantage des grammaires est que les règles sont simples et que la complexité vient de leur composition. Les 9 règles de la grammaire pour la détection des escaliers données figure 9.25 sont suffisantes pour obtenir les reconnaissances précises résumées dans la table 9.5.

9.8.3 Les grammaires, un langage pour les experts

La structure et la hiérarchie peuvent être codées en dur dans le code effectuant l'analyse sémantique. Cependant, une telle manière de procéder est difficile à mettre en œuvre pour des scènes complexes, sans compter le problème de maintenance et d'évolution, avec un manque de flexibilité qui imposerait de ré-implémenter l'algorithme à chaque changement de règle. De plus, un formalisme pour l'expression de la structure est nécessaire, pour d'une part permettre l'optimisation systématique (ordre sur l'application des contraintes et des règles) et d'autre part pour faciliter l'écriture des contraintes par les humains. La grammaire est ici vue comme un langage de spécification de haut niveau, propre au domaine d'application, pour exprimer des régularités. En fait, nous considérons que les règles doivent être écrites par des experts, par exemple des architectes, non des informaticiens. C'est pourquoi l'expression des grammaires doit rester simple et ne pas nécessiter de connaissances en programmation.

9.8.4 Apprendre les règles

Un aspect qui n'a pas été abordé dans cette étude est la possibilité d'apprendre les règles de la grammaire à partir d'une base de données contenant des exemples annotés. Ceci permettrait, tout comme l'utilisation de formes approximatives, de se prémunir contre une description exhaustive des règles de la grammaire, tout en gagnant en robustesse. Par exemple, même si la grammaire actuelle pour les escaliers est capable de gérer des nez de marches complexes (mais réguliers), il serait plus simple pour l'utilisateur de spécifier un nez « générique »et d'utiliser un algorithme s'adaptant à la forme du nez.

9.9 Conclusion

Ce chapitre a présenté un formalisme grammatical de haut niveau pour spécifier des objets complexes. Une procédure d'analyse a aussi été proposée et mise en pratique sur des exemples de modèles CAO, d'acquisition laser et de nuage de points photogrammétrique.

Bien que la méthode repose sur des algorithmes efficaces de théorie des graphes, la machinerie de bas niveau reste cachée, permettant à des utilisateurs non informaticiens d'écrire des spécifications et de les maintenir.

Grâce aux alternatives, aux opérateurs maximaux et à la récursion, l'expressivité de la grammaire est au moins celle de la mise en correspondance de graphes.

Les résultats présentés dans ce chapitre sont encourageants et prouvent que l'utilisation de grammaires dans des approches bottom-up est digne d'intérêt. Cette étude constitue une première étape et il reste à gérer correctement le bruit et les données incomplètes. La section 9.8 recense différentes pistes d'études pour améliorer le procédé et traiter plus efficacement les données réelles.

Publication

Ce travail a été présenté au Symposium On Geometry Processing 2013, à Gênes, Italie. Il a été publié dans la revue Computer Graphics Forum :

Alexandre Boulch, S. Houllier, Renaud Marlet, Olivier Tournaire.
Semantizing Complex 3D Scenes using Constrained Attribute Grammars.
Comput. Graph. Forum 32(5): 33-42 (2013)

Bibliographie

3D-Warehouse, T. (s.i.t.e.). SketchUp repository. https://3dwarehouse.sketchup.com/.

aim@shape (2006). Aim@Shape repository. http://www.aimatshape.net/.

Alliez, P., Tayeb, S., and Wormser, C. (2010). Aabb tree. *CGAL User and Reference Manual. CGAL Editorial Board*.

Attene, M., Robbiano, F., Spagnuolo, M., and Falcidieno, B. (2009). Characterization of 3D shape parts for semantic annotation. *Computer-Aided Design*, 41(10):756 – 763.

Biasotti, S., Marini, S., Mortara, M., and Patané, G. (2003). An overview on properties and efficacy of topological skeletons in shape modeling. In *Shape Modeling Int'l*.

El-Mehalawi, M. and Miller, R. A. (2003). A database system of mechanical components based on geometric and topological similarity. *Computer-Aided Design*, 35(1).

FreeCAD (s.i.t.e.). http://www.freecadweb.org/.

Gao, X., Xiao, B., Tao, D., and Li, X. (2010). A survey of graph edit distance. *Pattern Analysis and Applications*, 13(1):113–129.

Han, F. and Zhu, S. (2009). Bottom-up/top-down image parsing with attribute grammar. *Transactions on Pattern Analysis and Machine Intelligence*, 31(1):59–73.

Huang, H., Brenner, C., and Sester, M. (2011). 3D building roof reconstruction from point clouds via generative models. In *Advances in Geographic Information Systems*. ACM.

IfcObj (s.i.t.e.). Ifcopenshell, open source ifc geometry engine. http://ifcopenshell.org/ifcobj.html.

Kinect, M. (s.i.t.e.). Kinect sensor. http://http://www.xbox.com/fr-FR/Kinect.

Martinović, A., Mathias, M., Weissenberg, J., and Van Gool, L. (2012). A three-layered approach to facade parsing. In *ECCV*, LNCS 7578, pages 416–429. Springer.

Mathias, M., Martinovic, A., Weissenberg, J., and Gool, L. V. (2011). Procedural 3D building reconstruction using shape grammars and detectors. In *IEEE International Conference 3DIMPVT*, pages 304–311.

Müller, P., Wonka, P., Haegler, S., Ulmer, A., and Van Gool, L. (2006). Procedural modeling of buildings. *ACM Transactions on Graphics*, 25(3):614–623.

Müller, P., Zeng, G., Wonka, P., and Van Gool, L. (2007). Image-based procedural modeling of facades. *ACM Transactions on Graphics*, 26(3).

Pagallo, G. M. (1998). Constrained attribute grammars for recognition of multi-dimensional objects. In *Advances in Pattern Recognition*, LNCS 1451, pages 359–365. Springer.

Pascucci, V., Scorzelli, G., Bremer, P.-T., and Mascarenhas, A. (2007). Robust on-line computation of Reeb graphs: simplicity and speed. *ACM Transactions on Graphics*, 26(3):58.

Pauly, M., Mitra, N. J., Wallner, J., Pottmann, H., and Guibas, L. (2008). Discovering structural regularity in 3D geometry. *ACM Transactions on Graphics*, 27(3):1–11.

Ripperda, N. and Brenner, C. (2007). Data driven rule proposal for grammar based facade reconstruction. In *Photogrammetric Image Analysis (PIA)*, pages 1–6.

Schnabel, R., Wahl, R., and Klein, R. (2007). Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226.

Schnabel, R., Wessel, R., Wahl, R., and Klein, R. (2008). Shape Recognition in 3D Point-Clouds. In Skala, V., editor, *The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision'2008*. UNION Agency-Science Press.

Simon, L., Teboul, O., Koutsourakis, P., Van Gool, L., and Paragios, N. (2012). Parameter-free/Pareto-driven procedural 3D reconstruction of buildings from ground-level sequences. In *CVPR*, pages 518–525.

SketchUp, T. (s.i.t.e.). SketchUp software. http://www.sketchup.com/.

Strecha, C., von Hansen, W., Van Gool, L., Fua, P., and Thoennessen, U. (2008). On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *CVPR*.

Tangelder, J. W. and Veltkamp, R. C. (2008). A survey of content based 3D shape retrieval methods. *Multimedia Tools Appl.*, 39(3):441–471.

Teboul, O., Kokkinos, I., Simon, L., Koutsourakis, P., and Paragios, N. (2011). Shape grammar parsing via reinforcement learning. In *CVPR*.

Toshev, A., Mordohai, P., and Taskar, B. (2010). Detecting & parsing architecture at city scale from range data. In *CVPR*.

Vanegas, C., Aliaga, D., and Benes, B. (2010). Building reconstruction using Manhattan-world grammars. In *CVPR*.

Vanegas, C. A., Aliaga, D. G., and Benes, B. (2012). Automatic extraction of Manhattan-world building masses from 3D laser range scans. *IEEE Trans. Vis. Comput. Graph.*, 18(10):1627–1637.

Vu, H., Labatut, P., Pons, J., and Keriven, R. (2012). High accuracy and visibility-consistent dense multiview stereo. *IEEE Trans. Pattern Anal. Mach. Intell.*, 34(5):889–901.

Wu, T. and Zhu, S. (2011). A numeric study of the bottom-up and top-down inference processes in and-or graphs. *IJCV*, 93(2):226–252.

Conclusion

Chapitre 10

Conlusion et Perspectives

10.1 Contributions

Ce travail de thèse avait pour objet la reconstruction automatique de maquettes numériques à partir de nuages de points. Le sujet englobait à la fois la reconstruction de la géométrie et l'analyse sémantique de la scène. Ce document s'est articulé autour de quatre parties correspondant chacune à une brique de la chaîne de traitement, rappelée figure 10.1. Les contributions de ces travaux de thèse sont résumés pour chacune des parties.

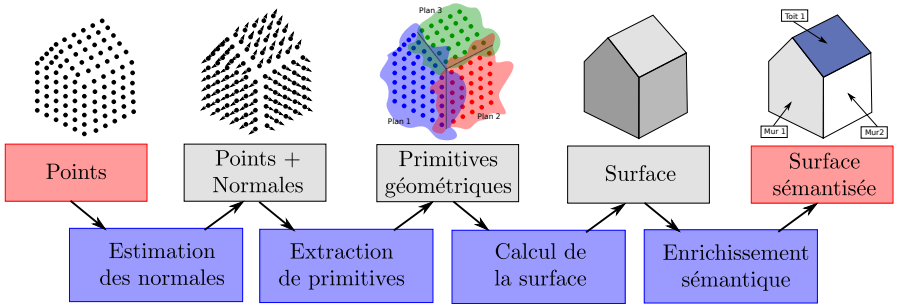


FIGURE 10.1 – Pipeline de reconstruction de maquettes numériques.

Estimation de normales

La partie I a présenté un nouvel estimateur de normales pour les nuages de points non structurés particulièrement adapté pour les zones anguleuses et les échantillonnages anisotropes. Il se base sur une transformée de Hough. Des hypothèses de normales sont générées à partir du voisinage du point observé et votent dans un accumulateur sphérique. Le nombre de tirages est régi par deux critères statistiques, l'un pour déterminer le nombre maximal d'hypothèses nécessaires pour une bonne estimation, et l'autre permettant de stopper les tirages lorsque le maximum est déterminé avec suffisamment de certitude. L'algorithme utilisé a été évalué sur divers nuages de points, avec différents niveaux de bruits et de points aberrants. Les normales sont correctement retrouvées dans les zones anguleuses, le tout en restant compétitif avec l'état de l'art quant au temps de calcul.

Fusion de primitives

Pour réparer les possibles sur-segmentations des algorithmes d'extraction de primitives, le chapitre II propose un critère statistique afin de déterminer si deux surfaces sont

identiques étant donnés les points qui leur sont associés. Ce critère repose sur une comparaison statistique de deux distributions de distances. Il exige uniquement de connaître une fonction distance entre un point et la surface considérée, ce qui permet de comparer des surfaces de types différents.

Reconstruction de surface planaire par morceaux

La partie III est consacrée à la reconstruction de surfaces à partir de nuages de points laser. Les environnements humains étant généralement constitués d'objets réguliers souvent plans et comportant des surfaces orthogonales, nous cherchons à reconstruire une surface idéalisée planaire par morceaux. Pour cela, nous générons des hypothèses de plans visibles (extraction de primitives dans le nuage de points) et invisibles (hypothèses fantômes) et nous les insérons dans un arrangement de plans 3D. Nous déterminons ensuite l'état plein ou vide de chacune des cellules de l'arrangement. Pour cela, une énergie prenant en compte l'anisotropie et intégrant des potentiels d'ordres supérieurs est définie, et une relaxation linéaire est effectuée afin d'être en mesure de calculer une solution dans un temps raisonnable. L'utilisation de potentiels d'ordres supérieurs permet de régulariser la surface en minimisant le nombre de coins et la longueur des arêtes dans la scène. Les solutions obtenues sont comparées à la principale méthode existante utilisant uniquement une régularisation sur l'aire de la surface reconstruite. Les résultats sont plus réalistes et les artefacts réduits.

Enrichissement sémantique

La dernière partie de ce manuscrit, chapitre 9, est consacrée à l'enrichissement sémantique de géométrie CAO et de nuages de points. L'approche proposée est bottom-up, partant des polygones générés à partir des données pour reconnaître par composition de manière hiérarchique des objets complexes tels que des escaliers ou des ouvertures dans les bâtiments. L'utilisation d'opérateurs maximaux et l'application automatique d'un ordre sur la résolution des contraintes permettent d'enrayer l'explosion combinatoire qui est généralement présente dans les approches bottom-up. La méthode est validée sur des exemples CAO et sur des nuages de points laser et photogrammétriques. Les résultats obtenus sont encourageants mais l'implémentation actuelle montre ses limites lorsque la géométrie est partiellement erronée ou manquante.

10.2 Discussion et perspectives

Ce travail a permis la mise en place d'une chaîne de traitement de nuages de points pour la reconstruction de maquettes numériques de bâtiments : géométrie et information sémantique.

L'approche sous forme d'une chaîne de traitement est une approche naturelle. Les informations sont traitées par niveaux de complexité croissante. On part des informations locales (les normales), pour construire des primitives puis la géométrie complète (la surface), et terminer par la reconnaissance d'objets complexes dans le modèle géométrique ainsi créé.

Chaque étape est bien identifiée et correspond à un problème restreint qui, sauf pour l'enrichissement sémantique, a beaucoup été étudié dans la littérature. Cette séparation en sous-problèmes permet le développement d'outils spécifiques tels que ceux qui ont été présentés dans ce travail de thèse. Chaque élément de la chaîne de traitement que nous

proposons est compétitif avec l'état de l'art, tant du point de vue de la précision que du temps de calcul.

Cependant l'un des principaux inconvénients d'une telle méthode est le caractère unidirectionnel du processus. La sortie d'un bloc est l'entrée du bloc suivant, cumulant ainsi les erreurs à chaque étape. De plus, il n'y a pas de boucle de rétroaction. Le choix de reconstruire la surface avant d'y attacher des informations sémantiques est restrictif. Il est en effet concevable d'utiliser des informations sémantiques pour guider l'estimation de la surface.

Il serait particulièrement intéressant de lier les étapes de calcul de surfaces et d'enrichissement sémantique. Bien que l'algorithme présenté dans la partie III soit capable de gérer des éléments manquants, tels que des plateaux de marches invisibles, il n'est pas capable d'inférer des éléments complexes, comme un ensemble de marches manquantes. Cette génération peut-être guidée par la découverte de symétrie et de répétition mais aussi par la sémantique attachée aux éléments. Savoir qu'il manque une marche pour atteindre le pallier permettrait de l'ajouter même si elle n'a pas été détectée. De même, découvrir que toutes les fenêtres d'une façade sont identiques et quelles sont alignées sur une grille permettrait de compléter les primitives manquantes et de corriger les primitives détectées (figure 10.2).

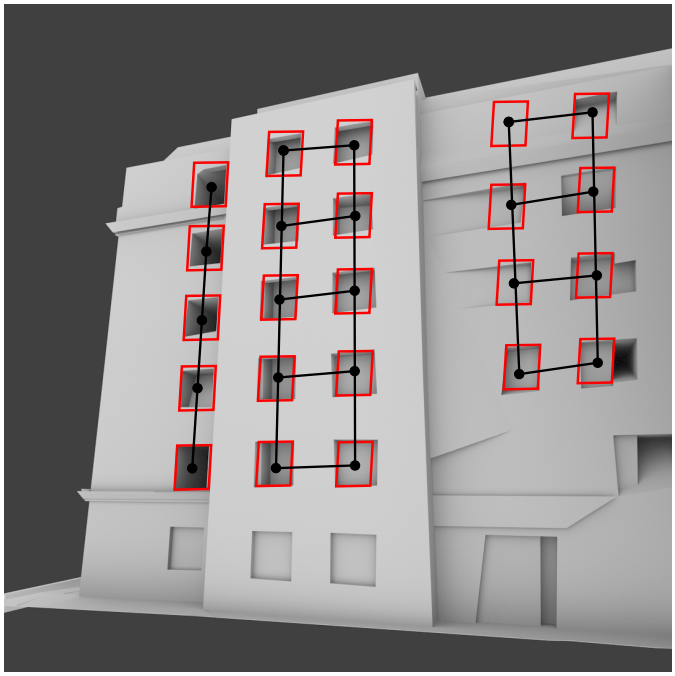
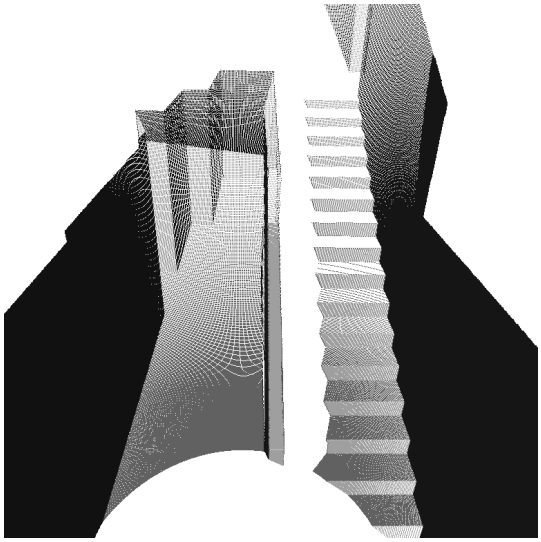
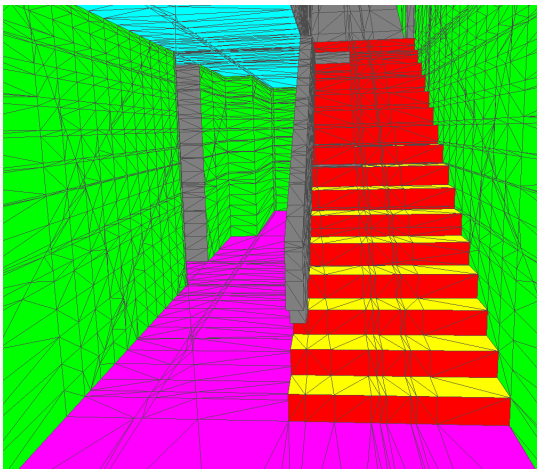


FIGURE 10.2 – Régularité pouvant aider à la reconstruction de la façade (forme et configuration).

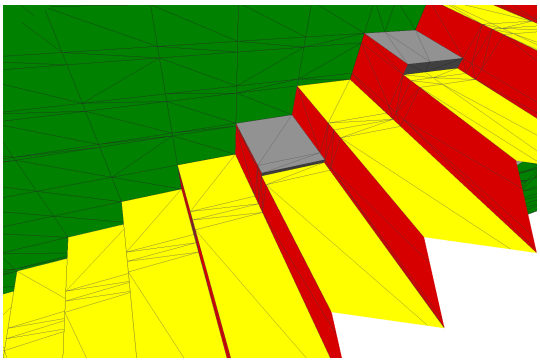
Il est en effet possible d'imaginer une boucle sur les deux étapes jusqu'à convergence. Les premières ébauches en ce sens semblent encourageantes. La figure 10.3 présente un nuage de points laser synthétique d'un escalier (figure 10.3a), une partie des marches n'est pas échantillonnée car cachée par la rampe. La reconstruction produit certains artefacts (figure 10.3c) qui sont supprimés par une boucle de rétroaction simple consistant à pénaliser les surfaces pour lesquelles aucun label sémantique n'a pu être affecté.



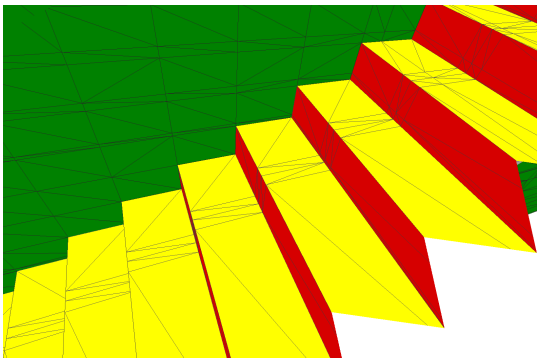
(a) Nuage de points.



(b) Vue d'ensemble de la reconstruction avant rétroaction.



(c) Détail avant rétroaction.



(d) Détail après rétroaction.

FIGURE 10.3 – Amélioration des marches d'un escalier dans l'invisible.

Enfin, un aspect qui n'a pas été abordé dans ce travail est celui de l'étiquetage des volumes. Tout au long de cette thèse, l'objectif était la surface : construction puis étiquetage de celle-ci. Or, les objets sont fondamentalement volumiques et l'étape de calcul de surfaces travaille déjà sur des volumes (étiquetage plein ou vide des cellules).

Les véritables maquettes numériques sont construites par volumes et non par surfaces. Il est possible d'imaginer de fusionner les deux étapes critiques (surfaces et labels). Cette unique étape porterait sur l'étiquetage des volumes, non seulement comme pleins ou vides, mais avec des labels sémantiques.