Modélisation de scènes urbaines à partir de données aeriennes

Yannick Verdie

UNIVERSITY OF NICE - SOPHIA ANTIPOLIS

DOCTORAL SCHOOL STIC

SCIENCES ET TECHNOLOGIES DE L'INFORMATION ET DE LA COMMUNICATION

P H D T H E S I S

to obtain the title of

PhD of Science

of the University of Nice - Sophia Antipolis

Specialty : Computer Science

Defended by

Yannick VERDIE

Urban scene modeling from airborne data

Thesis Advisors : Florent Lafarge and Josiane Zerubia

prepared at INRIA Sophia Antipolis, Titane/Ayin Teams

defended on October 15, 2013

Jury :

Reviewers :	Niloy J. Mitra	-	UCL
	Luc Van Gool	-	ETH Zürich
Advisors :	Florent Lafarge	-	INRIA (TITANE)
	Josiane Zerubia	-	INRIA (AYIN)
Examinators :	Pierre Alliez	-	INRIA (TITANE)
	Mathias Ortner	-	EADS Astrium (Toulouse)
	Jean-Philippe Pons	-	ACUTE3D (Nice)

Acknowledgements

Foremost, I would like to thanks my co-advisor Florent Lafarge for his continuous support and hard-criticism towards my work which made me get the best out of myself. His general guidance and wise advices helped me in all time of research.

Besides my co-advisor, I would like to thanks my advisor Prof. Josiane Zerubia for her help, especially during the preparation of my 5-month trip to Japan and during the final stages of my thesis. Importantly, I would like to convey my thanks and gratitude to the rest of my thesis committee : Prof. Luc Van Gool, Prof. Niloy J. Mitra, Prof. Pierre Alliez, Jean-Philippe Pons and Mathias Ortner, for their time spent reading my thesis, constructive questions, and insightful comments. In addition, I gratefully acknowledge various contributors who provided useful datasets for experiments : Acute3D, IGN, BRDM, Tour du Valat ,Victor Lempitsky, Roger W. Ehrich and Qian-Yi Zhou.

Finally, I thank the professors and lab-mates from the various teams I visited (five teams in three years) : David Bommes, Ross Hemsley, Vladimir Krylov, Alejandro Mottini, Kaimo Hu and Manish Mandad, thanks for taking the time to proof read this thesis. Prof. Akihiro Sugimoto, thanks to have hosted me in your lab in Tokyo. I discovered there a new culture which is fascinating : Japan, with its quite unique mixture of traditionalism and post-modernism, will never stop to fascinate me.

Last, but not least, I am thankful to my close friends and family, for all their moral support and cheering they provided. Thank you.

Yannick Verdie

INRIA Sophia Antipolis

October 2013

Résumé

L'analyse et la reconstruction automatique de scène urbaine 3D est un problème fondamental dans le domaine de la vision par ordinateur et du traitement numérique de la géométrie. Cette thèse présente des méthodologies pour résoudre le problème complexe de la reconstruction d'éléments urbains en 3D à partir de données aériennes Lidar ou bien de maillages générés par imagerie Multi-View Stereo (MVS). Nos approches génèrent une représentation précise et compacte sous la forme d'un maillage 3D comportant une sémantique de l'espace urbain.

Deux étapes sont nécessaires ; une identification des différents éléments de la scène urbaine, et une modélisation des éléments sous la forme d'un maillage 3D.

Le Chapitre 2 présente deux méthodes de classifications des éléments urbains en classes d'intérêts permettant d'obtenir une compréhension approfondie de la scène urbaine, et d'élaborer différentes stratégies de reconstruction suivant le type d'éléments urbains. Cette idée, consistant à insérer à la fois une information sémantique et géométrique dans les scènes urbaines, est présentée en détails et validée à travers des expériences.

Le Chapitre 3 présente une approche pour détecter la 'Végétation' incluses dans des données Lidar reposant sur les processus ponctuels marqués, combinée avec une nouvelle méthode d'optimisation. Le Chapitre 4 décrit à la fois une approche de maillage 3D pour les 'Bâtiments' à partir de données Lidar et de données MVS. Des expériences sur des structures urbaines larges et complexes montrent les bonnes performances de nos systèmes.

Mots clefs : Traitement de l'image, Traitement de la géométrie, Reconstruction et Analyse de scènes urbaines, Modélisation 3D, Processus Ponctuels Marqués, LiDAR, Multi-View Stereo, Minimisation d'énergie, Champs Aléatoires de Markov

Abstract

Analysis and 3D reconstruction of urban scenes from physical measurements is a fundamental problem in computer vision and geometry processing. Within the last decades, an important demand arises for automatic methods generating urban scenes representations. This thesis investigates the design of pipelines for solving the complex problem of reconstructing 3D urban elements from either aerial Lidar data or Multi-View Stereo (MVS) meshes. Our approaches generate accurate and compact mesh representations enriched with urban-related semantic labeling.

In urban scene reconstruction, two important steps are necessary: an identification of the different elements of the scenes, and a representation of these elements with 3D meshes.

Chapter 2 presents two classification methods which yield to a segmentation of the scene into semantic classes of interests. The benefit is twofold. First, this brings awareness of the scene for better understanding. Second, different reconstruction strategies are adopted for each type of urban elements. Our idea of inserting both semantical and structural information within urban scenes is discussed and validated through experiments.

In Chapter 3, a top-down approach to detect 'Vegetation' elements from Lidar data is proposed using Marked Point Processes and a novel optimization method. In Chapter 4, bottom-up approaches are presented reconstructing 'Building' elements from Lidar data and from MVS meshes. Experiments on complex urban structures illustrate the robustness and scalability of our systems.

Keywords: Image processing, Geometry processing, Urban scene reconstruction, Scene understanding, 3D modeling, Marked Point Processes, LiDAR data, Multi-View Stereo data, Energy minimization, Markov Random Field

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Context

The goal of this thesis is to explore novel techniques for automatic 3D urban scene reconstruction from various data sources such as airborne Lidar data and Multi-View Stereo (MVS) data. The reconstruction of urban scenes is a topic of great interest in multiple domains (including public, private, and military). In the last century, more and more rural areas have been abandoned, and the occupants transited to urban areas [UN 2012]. This urbanization creates an increasing need for an automatic and accurate urban reconstruction to support human-related applications. In the domain of urban planning, risk management, virtual touring, military, advanced city-related applications will drastically benefit end-users for an everyday usage as well as companies for urban planing or any urban-related applications.

The recent rise of web-mapping services (such as Google earth [Google 2013] or Microsoft virtual earth [Microsoft 2013]) demonstrates the current interest and the great potential of automatic methods to generate 3D city maps. The demand only increases as the size of the cities gets larger and larger. Video game companies are also showing interest in automatic 3D urban scene reconstruction. Similarly, telecommunication companies see the benefit from such 3D reconstruction for efficiently deploying telecommunication antennas in urban areas. Finally, for rescue situations, 3D urban representation can assist autonomous robots navigation and drone flight route planning.

Despite this promising range of applications, the current 3D modeling coverage is still restricted to just a few areas worldwide. The primary reason for this is the difficulty of acquiring the data, which even today requires a lot of manual effort.

Within the last decade, however, two different techniques to automatically collect urban scene data have been proposed. On the one hand, laser scanners such as Lidar are used to provide an accurate and dense 3D point cloud of an urban scene. On the other hand, multiple images of the cities have successfully been used together in order to synthesize 3D meshes. While the usage of these two techniques to acquire 3D representations of urban scenes have been considerably improved and can now generate reliable and accurate representations, these methods generate a huge quantity of information, the direct usage of which is often intractable. Providing methods to convert this huge amount of data into a practically useful representation is a problem of great importance.

Ultimately, despite a lot of effort having been put into generating complete 3D urban reconstructions, no satisfying solution has yet been brought to light. Meeting these demands still represents a challenging problem.

1.2 Data representation

While there are multiple data types in urban scene acquisition, we shall focus on just two types, (1) MVS meshes - obtained from image data - and (2) Lidar data, both of which are used within the methods proposed in this thesis. These two types of data, which we will define later, are the most commonly chosen and used in urban scene reconstruction (see Fig 1.1).

1.2.1 Image data

1.2.1.1 General description

High quality image data was initially rare, expensive and available only to a few from the remote sensing community through satellites acquisition. However, recent advances in camera sensors and web-based image libraries, such as Flickr and Picassa has made high quality image data widely spread and publicly available. Image data is now easy to acquire, store, and is not costly. This results in billions of pictures being taken every year. Since these photos mostly depict (fully or partially) urban scenes, many projects of urban reconstruction have successfully been conducted

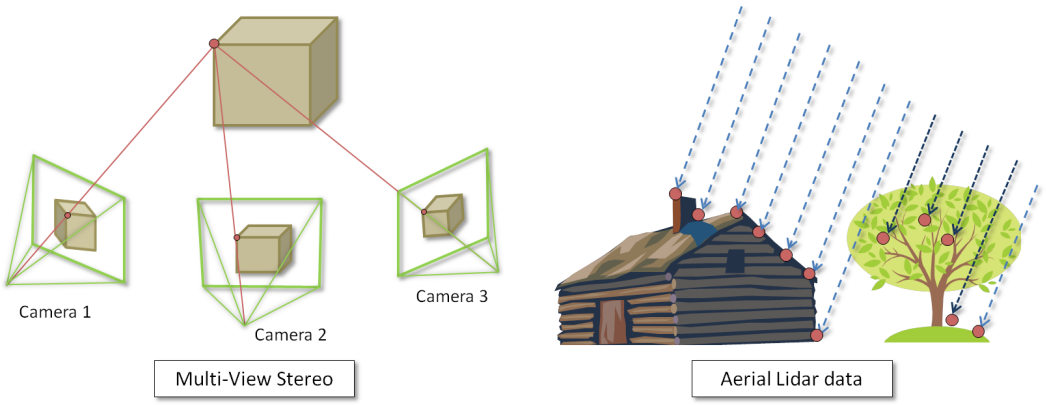


Figure 1.1: Illustration of the principle of Multi-View Stereo technique and aerial Lidar data. Note that for aerial Lidar data the lines in dark blue indicate pulses with multiple returns.

relying on them [Cipolla 1999, Dick 2004, Goesele 2007, Sinha 2008, Agarwal 2009, Furukawa 2010, Jancosek 2011]. The availability of this source of information makes image-based approaches extremely attractive see Fig 1.2). However, it raises the problem of scalability; as the number of images grows, the optimization becomes unstable and costly.

1.2.1.2 Multi-View Stereo data

Multi-View Stereo (MVS) data is usually of the form of a 3D point cloud or mesh generated from a collection of images. These meshes are obtained by the following steps (illustrated in Fig. 1.3, lower row). First, the position of the camera is recovered from each image. This step is usually done by a Structure from Motion (SfM) technique where salient features from each image are clustered within a feature space. The extrinsic parameters of the cameras are deduced from the set of correspondences and a sparse point cloud of selected salient features can be obtained (see Fig. 1.3, upper row). For more details on SfM technique, various methods for estimating camera parameters are described in details within a paper of Oliensis et al [Oliensis 2000]. Finally, MVS techniques generate dense point clouds and representations of the scene as 3D meshes (last two steps on Fig. 1.3, lower

Figure 1.2: Illustration of image data with corresponding scene reconstruction. Top-left: Examples of aerial image data. Bottom-left: Examples of stereo ground-based image data. Top-right: Reconstruction of the coliseum by Structure from Motion (SfM) technique with images from Flickr. Bottom-right: Reconstruction of the coliseum by patch of Multi-View Stereo (MVS) technique with images from Flickr. Both SfM and MVS images courtesy of Furukawa et al. [Furukawa 2010].

row). Note that the mesh is generated with traditional method such as Poisson reconstruction [Kazhdan 2006].

Following the taxonomy of Seitz et al. [Seitz 2006], MVS techniques are classified into four classes: 3D volumetric (e.g. the work of Hornung et al. [Hornung 2006]), depth map merging (e.g. Goesele et al. [Goesele 2006]), surface evolution (e.g. Zaharescu et al. [Zaharescu 2007]) and region-growing (e.g. Furukawa et al. [Furukawa 2010], Hiep et al. [Hiep 2009]). The latter approach is chosen to generate MVS meshes needed for our methods. This approach first obtains the camera parameters to generate a quasi-dense point cloud by traversing the epipolar lines on each image looking for matching features. A 3D Delaunay triangulation is then incrementally constructed from the sparse point cloud generated with matching features. A minimum s-t cut encoding discrete visibility and surface quality extracts an initial surface representation (this surface is modeled as the interface

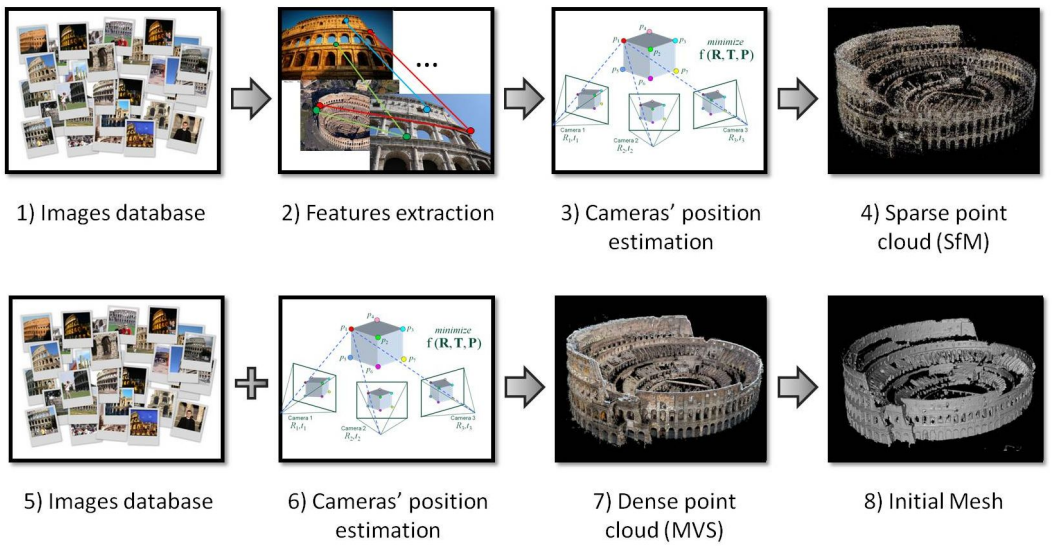


Figure 1.3: Steps for a standard SfM (upper row) and MVS (lower row): 1) An large number of images is gathered. 2) Features are extracted (with general methods such as SIFT, Harris corner detection) and grouped by similarity. 3) the relative pose of each camera is computed. 4) SfM: The features are projected in the 3D space to build a sparse point cloud. 5) and 6) The position of the cameras with respective images are collected (usually by SfM). 7) MVS: dense point cloud is generated from all the images and the relative pose of each camera. 8) A mesh is generated from the dense point cloud.

between volumes labeled *inside* an object and volumes labeled *outside* an object). Finally, this initial surface is used as a starting solution for a variational refinement. The refinement returns the final surface which typically has greater complexity to adapt to sharp creases and edges while being smooth and sparse on flat areas. The variational refinement also decreases the level of noise and corrects discontinuities on the original surface. More details and highlights on this method are given in the paper of Hiep et al. [Hiep 2009].

MVS meshes generated from Hiep et al. [Hiep 2009] are shown to be more accurate and complete than previous state-of-the-art methods in MVS techniques at that time. Particularly, the right mixture between photo-consistency and regularization

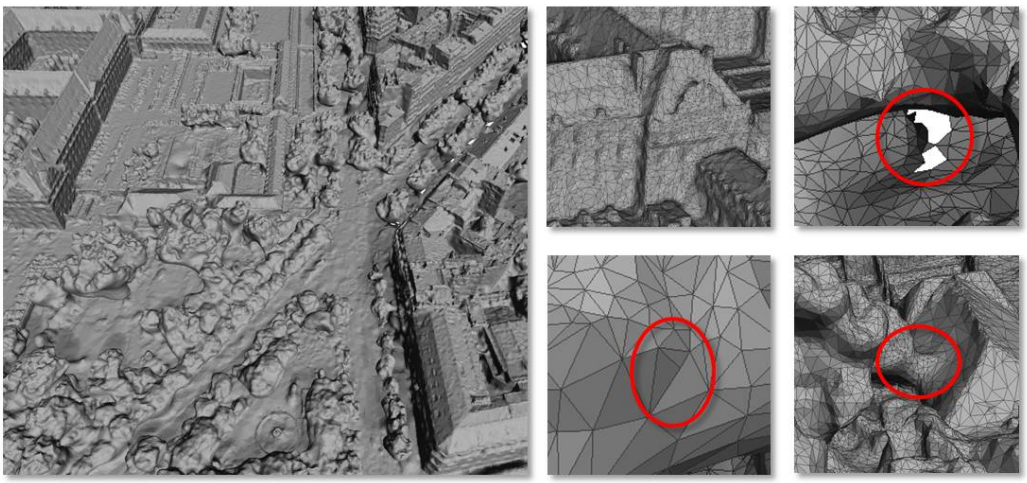


Figure 1.4: MVS meshes. Input data are dense meshes (approx. 10M facets per km2) generated by a Multi-View Stereo system from aerial images. They are semantic-free and contain geometric and topological defects such as holes, self-intersecting facets, wrong handles that merge urban components, and highly variable spatial distributions of vertices and facets (see close-ups).

leads to a smooth but accurate reconstruction of the scene. In addition, the mesh is automatically adjusted to image resolution and preserves sharp and small details of the objects. Comparing with Lidar data (described in the following Section 1.2.2), MVS data has a lower altimetric accuracy but a more homogeneous planimetric distribution. In addition, the neighborhood relationship between the points is induced and radiometric information contained in the images can support the semantization of the scene (see Chapter 2).

MVS meshes are, however, not optimal for certain applications such as urban planning, wireless propagation simulation or 3D navigation for which 3D-models must either be compact or contain semantical information allowing the identification of urban objects within the scene. Moreover, geometrical and topological defects exist in the meshes; the main problems come from the missing data (holes) and self-intersecting facets, the spatial heterogeneous distributions of vertices and facets, the loss of accuracy in presence of reflecting surfaces such as glass, and the merging of different urban components such as trees with facades. These defects - mostly

generated by the variational approach of Hiep et al. [Hiep 2009] - are illustrated in Fig. 1.4. Regarding the semantization of the mesh, a solution to generate the semantics of the urban scene from MVS meshes is proposed in Chapter 2. Additional information and a brief overview of methods using Structure from Motion and Multi-View Stereo techniques is given in Section 1.3.

1.2.2 Lidar data

1.2.2.1 General description

Lidar data delivers precise and quasi-dense point clouds suitable for urban reconstruction. Lidar - standing for "LIght Detection And Ranging" - is a technology using an active sensor to measure distances. The technology is based on the time-of-light principle where a laser device emits a light pattern and the reflected light is analyzed. Lidar technology relies on two main characteristics: First, the reflection of light pattern - also called backscatter effect - has an intensity depending on the size of the object as well as its intrinsic reflectance property. Thus, the reflectance property of the elements is known from the returned signal. Second, the time for the light pattern to travel back and forth is directly proportional to the distance from the scanner to the object, and therefore, a precise measure of the distance is obtained. This technology was initially used for meteorology, but it has quickly been identified to have great potential for large scale measurements. Lidar has been employed in numerous projects to asset the landscape topology such as ocean floors, forest canopies, urban sceneries and mountains. The properties listed below explain some reasons for the success of Lidar:

- **Versatility:** Different 'signals' can be sent depending on the situation. To penetrate the water, a narrow wavelength band (typically 532 nanometer beam) is used while a wider band is used for standard measurement (1064 nanometer beam).
- **Completeness:** The measurement is extremely dense with around 100 000 to 200 000 measurements per second, representing around eight (respectively thirty) points per square meter for airborne Lidar data (respectively terrestrial

Lidar data).

– **Accuracy:** The measurement is extremely precise with a sampling error of a few centimeters (respectively millimeters) for airborne Lidar data (respectively terrestrial Lidar data) .

Compared to image-based MVS data, Lidar measurements are very accurate, making this technology suitable for very detailed reconstruction of buildings and facades. In addition, airborne Lidar scanners can easily scan very large scenes. Another important advantage of this technology over conventional cameras is its relative immunity from lighting change and variation. Altogether, Lidar data is particularly appropriate for urban reconstruction. However, various factors affect the benefit of using such a technology. To begin with, this technology is still uncommon: the scanner required to collect the data is extremely expensive and very few laboratories can afford one. Furthermore, Lidar data is arranged as incomplete, semi-dense point clouds and the photometric information is often missing. The point clouds have missing measurements for two main reasons: Firstly, this technology is strongly dependent on the material property of the objects or surfaces scanned, leading to poor quality or missing measurements for non-opaque materials (e.g. transparent objects such as windows do not respond like opaque objects). Secondly, holes and discontinuities are found within the measurements due to the line-of-sight property of a static scanner. Moving the scanner or using multiple scanners are simple but effective ways to limit this problem (although this does not completely suppress it). These strategies are commonly chosen in the literature. Other methods to deal with missing data include synthesizing 3D points (up-sampling Lidar data techniques, i.e. the work of Wang et al. [Wang 2011]) or using closed surfaces for the 3D meshes (i.e. the work of Kim et al. [Kim 2011]). All these defects, i.e. poor measurements, missing data and occlusions, can significantly impair the quality of the Lidar data (see Fig. 1.5).

Note that although the minimum information provided by Lidar data is the distance to the objects, additional information is sometime provided, such as number of returned patterns (or number of echoes), amplitude of the returned

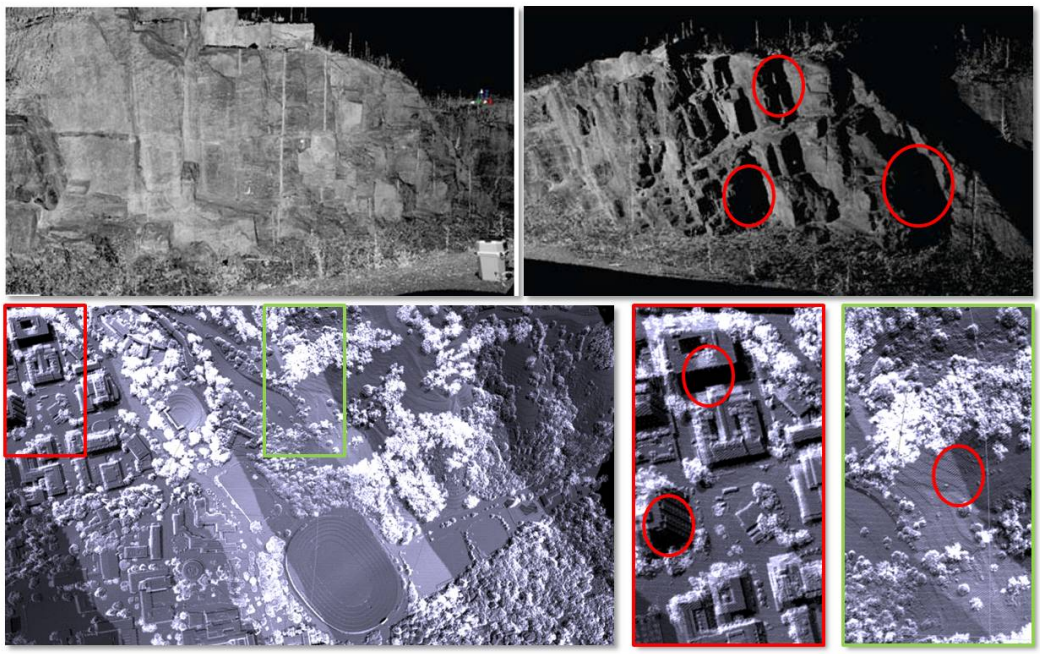


Figure 1.5: Top-left: Terrestrial Laser Scanner data from the viewpoint of the scanner where the measurement seems dense. Top-right: Same data visualized from the side of the line-of-sight of the scanner. Note the holes and the strong discontinuities (red circles) in the density of the measurement. The images are courtesy of [Lato 2010]. Bottom-left: Airborne Lidar data collected in five flight passes. Bottom-Right: Note that the vertical structures are not always scanned (red box) and that the overlapping swath width creates strong variation in the point cloud density (green box). Data taken from [Opentopography 2013].

signals, and first and last echo. This information gives hints on the material properties of the objects, and can be used for distinguishing them from each other (see the classification method described in Chapter 2). For more information on the analog form of the Lidar data and a way to digitalize it, one can consult the work of Mallet et al. [Mallet 2010]. They use a Marked Point Process (MPP) to model the backscattered signals as a mixture of Gaussians.

The following sections describe more details of the Lidar technology, and in particular two sub-categories which are terrestrial Lidar, and airborne Lidar.

1.2.2.2 Terrestrial Lidar

Terrestrial Laser Scanners (TLS) are one way to acquire Lidar data. The wide range of TLS yields to a variety of technical solutions of supporting platforms, such as cars, tripods or backpacks. Mounted on a car, the TLS is used for scanning facades of buildings as mentioned in the work of Fruh et al. [Fruh 2004]. In the same way, Haala et al. [Haala 2008] use the StreetMapper mobile laser scanning system that captures a 360° measurement of the street. In their setting, the system has four Lidar scanners with an accuracy of 20mm each. Note that the GPS/IMU information, as well as a 3D synthesized model of the city is used to register and properly align Lidar scan to each other.

Recently, TLS has been mounted on a backpack for indoor scene reconstruction [Liu 2010]. Image sensors and TLS are calibrated to each other to obtain an accurate reconstruction of indoor environments. Image features are used to properly compute a mosaic of images, and then extracted plane helps the mapping of the textures to the laser scan. In general, the measurement from a static-based TLS is an incomplete semi-dense point cloud (see Fig. 1.5, top row).

1.2.2.3 Airborne Lidar

Airborne Lidar data is acquired from Airborne Laser Scanners (ALS) mounted on aerial vehicles such as planes, drones, helicopters, etc. The scanner relies on GSM/IMU information to properly register the data. Within the last decade, a significant quantity of airborne Lidar data have been collected from planes by the American National Center for Airborne Laser Mapping [NCALM 2013] and made available for NSF projects. More recently, numerous projects hosted mostly in the US made their Lidar data also available for research purposes [USGS 2013, Opentopography 2013, NOAA 2013].

The principal advantage of ALS is its ability to perform quick acquisitions since the aerial vehicle covers a large region in a short period of time. However, the

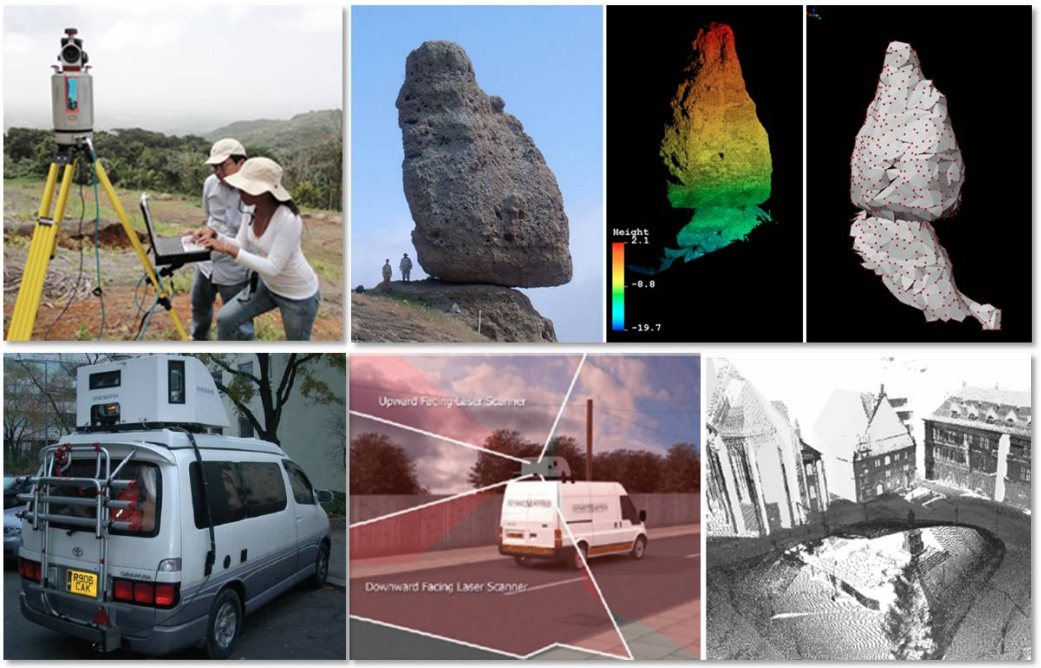


Figure 1.6: Top-left: Image of Terrestrial Laser Scanner (Riegl VZ-400) mounted on a tripod and operated by Professor Guoquan Wang and student Arlenys Ramiriz from the University of Puerto Rico at Mayaguez. Top-right: the resulting Lidar data and modeling. The images are courtesy of the UNAVCO project [UNAVCO 2008]. Bottom-left: Image of StreetMapper mobile laser scanning system. Bottom-right: the resulting Lidar data. Images courtesy of [Haala 2008].

principal drawback of terrestrial Lidar - i.e. the line-of-sight issue - is accentuated with aerial vehicles since the scanner is further aways from the objects and is oriented downwards. As a consequence, most of the vertical structures (e.g. facades, cliffs, etc.) are missing from the measurement. On Fig. 1.5 (red box), one can see that most of the vertical facades of the buildings are missing, but some parts were nonetheless partially captured due to the oblique line-of-sight of the scanner partially covering them. This makes the classification of the points even more difficult since one cannot enforce any strong priors on the data, such as assuming that no point belongs to a vertical structure.

Figure 1.7: Illustration of Airborne Laser Scanner (ALS) and airborne Lidar data. Left column: acquisition vehicle for airborne Lidar measurement. Right column: Airborne Lidar data with color corresponding to the different amplitudes of the backscattered signal. Images courtesy of [Geoinformatics 2007].

Even though these two types of data can be acquired from terrestrial or airborne sensors, we note that in practice Lidar data is captured through aerial scanners while MVS data is generally synthesized from ground-based images (taking advantage of the quantity of pictures taken by tourists and freely distributed online). Recently, a new trend can be observed in which point cloud data is used for indoor reconstruction [Liu 2010, Oesau 2013]. This tendency is mostly due to the recent availability of low-cost range scanners such as the Microsoft Kinect which provide dense but noisy measurement within a distance of one to five meters [Izadi 2011, Zhou 2013]. These approaches are only in the early development phase and out of the scope of this thesis.

In the following sections, we give a review on general approaches using image data, Lidar data, or both to reconstruct urban scenes.

1.3 Literature review on urban reconstruction

There are multiple approaches for reconstructing 3D models of urban scenery. In a similar manner as Musialski et al. [Musialski 2013], we distinguish between approaches requiring user input called *scene modeling* methods and fully automatic approaches called *scene reconstruction* methods. Scene modeling requires user interactions to control the process. The degree of interaction depends on the quality required by the user; more interaction leading to more precise reconstruction at the cost of scalability. In scene reconstruction, a full automatic pipeline of actions is performed to reconstruct the scene without any need for user interaction.

1.3.1 Interactive single-image based methods

Single-image scene modeling methods use an image as a reference for the user. This provides the base to manually annotate information such as vanishing points, planar clusters, etc. to compensate the depth information intrinsically missing in 2D images. Single-image scene modeling was of interest a few years ago. For example, Debevec et al. [Debevec 1996] required user annotations of the main edges of the buildings to infer the 3D properties of the scene. Cipolla et al. [Cipolla 1999] propose "PhotoBuilder": a method using uncalibrated images in combination with user interactions to reconstruct textured buildings. Recently, Jiang et al. [Jiang 2009] exploit symmetries and information provided by the user to properly reconstruct complex buildings. A limitation of the single-image scene modeling approach is its strong dependency on information provided by the user, making the system incompatible with an automatic execution. This directly restrains the use of such methods to single building reconstruction. Moreover, with only one image, too many degrees of freedom are left resulting in ambiguities.

1.3.2 Interactive and automatic multiple-image based methods

Multiple-image scene modeling and reconstruction methods like [Dick 2004, Goesele 2007, Furukawa 2009] are more robust solutions. We distinguish two main classes of techniques:

1.3.2.1 Structure from motion

Structure from motion (SfM) initially consisted of estimating the 3D camera location in each image taken from a temporal input sequence. By detecting a few relevant feature points in a set of pictures as well as similarities between themselves, one can obtain highly accurate camera parameters and a sparse 3D point cloud of the scene. Different approaches using SfM have been proposed in the literature [Hengel 2006, Sinha 2008, Cornelis 2008, Agarwal 2009]. Hengel et al. [Hengel 2006] fit user-defined models to a point cloud generated by SfM. Starting from an initial fit provided by the user, cuboids or planes are iteratively refined to make them fit properly when projected into the pictures. Sinha et al. [Sinha 2008] use SfM technique for the 3D modeling and user interactions during the texture mapping. Vanishing directions are accurately computed to assist the 2D sketching. From these draws, 3D planes textured by using energy optimization and Poisson blending are computed. Recently, remarkable automatic SfM methods were published [Goesele 2007, Agarwal 2009]. Thousand of pictures extracted from web-based services like Flickr were used to reconstruct complete urban districts. In these methods, the main challenge lies in the overwhelming quantity of images fed into the system. A parallel scheme is proposed to efficiently detect the similarities between pictures followed by a very efficient optimization method for extremely large non-linear least squares problems.

1.3.2.2 Multi-View Stereo

Multi-View Stereo (MVS) methods generate a dense 3D structure (points or patches) from an unsorted set of images. 3D structures are computed using the

known camera locations with their respective images. The location of the cameras can be obtained using a SfM technique. Initially, MVS methods relied on the concept of planar-sweeping to infer the 3D structures [Collins 1996]. However, the methods have drastically evolved within the last few years (for more details, we refer the reader to the excellent survey on MVS by Seitz et al. [Seitz 2006]). For example, Kang et al. [Kang 2001] improve the MSV method using a swifting window that properly handles occlusions. Hiep et al. [Hiep 2009] propose a robust automatic MVS method relying on graph optimization. This method generates high quality urban meshes which are unfortunately without any semantics. In Chapter 2, we will propose a method to add the missing semantics and to simplify meshes generated with this approach. Finally, Jancosek et al. [Jancosek 2011] improve MVS for weakly-supported surfaces. A new weight computation is proposed for properly labeling inside-outside 3D Delaunay tetrahedralization which leads to the mesh reconstruction. These methods provide large scene reconstructions but the resulting meshes have geometric defects, neither semantics nor any understanding of the scene. Moreover, to preserve small details of the scene, dense meshes are generated even in flat areas where it would not be necessary. Having large reconstructions as well as accurate details is a difficult task.

1.3.3 Interactive laser based methods

Laser-based scene modeling and reconstruction methods use data from laser scanners to synthesize urban scenes. A 3D point cloud of the scene is easily obtained from depth maps (as for Microsoft Kinect) or directly from the scanners (as for Lidar data). As explained before, laser-based data (and specifically Lidar data that is rather dense and quite semi-regular) is perfectly suited for urban modeling. However, noise and outliers often corrupt the measurement, requiring user interaction to reconstruct the urban scenes. Nan et al. [Nan 2010] propose an interactive fitting of boxes on incomplete Lidar data for reconstruction using data and contextual considerations. Discrete optimization is used to assemble boxes, by balancing snapping forces, a data-fitting term and a contextual term. Thanks to

manual editing, which brings high-level knowledge about the semantic of the scenes, accurate 3D models are generated in spite of missing or inaccurate data. On the other hand, Du et al [Du 2011] use low-cost depth cameras such as Microsoft Kinect and user interactions for interactive indoor modeling. In their work, online user interactions improve the real-time 3D modeling, for example, help with the loop closure problem [Strasdat 2011], inside-outside labeling and 3D indoor mapping.

1.3.4 Automatic laser based methods

While interactive methods produce satisfactory results, they operate appropriately only for individual buildings. When it comes to large scene reconstruction, the manual operations within the interactive methods become tedious. Thus, automatic large scene reconstruction [Zhou 2008, Poullis 2009, Zhou 2012, Lafarge 2012] seems to be the next logical step to handle large and detailed urban scenes. In the case when building footprints in the scene are unavailable, one can extract them as a collection of rectangles fitting an elevation map [Lafarge 2010a], as the result of a cell partitioning [Kada 2009], using alpha shape of a subset of points [Park 2006], grid labeling [Zhou 2008] or Voronoi cell labeling [Rottensteiner 2005]. In addition to the building footprints, additional features are sometime needed during the process. Verma et al. [Verma 2006] extract 3D segments and bounded 3D planes to generate adjacency graphs. These graphs are used to detect common building shapes (L-shape, U-shape, T-shape) to model the corresponding 3D structures. The remaining parts - the parts that are not modeled yet - are simply represented as rectilinear buildings. Lafarge et al. [Lafarge 2012] generate a 2D map of the projected bounded planes (sometime called 3D patches in the literature) and compute 2D relationships between projections. A Markov Random Field (MRF) modeling the connections is solved with graph cut [Boykov 2001] and then back-projected to 3D, leading to compact and accurate building reconstruction. Instead of directly processing the Lidar measurements, Wahl et al. [Wahl 2008] use a very dense and detailed Digital Surface Model (DSM) and simplify the mesh in such a way that relevant features are preserved. The simplification is based on semantic constraints on detected primitives (with "Efficient Ransac"

[Schnabel 2007]). The DSM can be simply obtained from the Lidar point cloud. More advanced DSM generation methods are described in details in the work of Al-Durgham et al. [Al-Durgham 2010].

While many researchers focus on the challenging problem of reconstructing buildings only, Lafarge et al. [Lafarge 2012] complete the urban modeling by also considering more objects of the urban environment. The proposed method for complex urban scene reconstruction also considers building, vegetation and ground reconstruction. First, a labeling problem of the Lidar point cloud in four classes (*building*, *ground*, *vegetation*, and *non-classified*) is done by solving a MRF formulation. The buildings are generated from a 2D grid extruded upward following 3D planar patches and simplified with a constrained decimation method. The ground is generated by meshing the Lidar points labeled as *ground* while the trees - the most dominant elements in the *vegetation* - are reconstructed using a 3D template (3D vertical ellipsoidal shapes) with parameters estimated using a watershed algorithm. Conceptually, this work is similar to the method presented in Chapter 4.

In regard to facade modeling, Pu and Vosselman [Pu 2009] propose a segmentation of the terrestrial Lidar data for fitting parametric models on facades. A bottom-up process is used to extract geometric and semantic features. Finally, hypothesized models of occluded parts of the buildings are used to generate the facades. Another methodology is to use grammar rules that properly describe the recurrent structures of facades. For example, Becker et al. [Becker 2009] exploit grammar rules on terrestrial Lidar data for robust facade reconstruction. A procedural modeling of building structures is performed based on a bottom-up and top-down strategy. A first bottom-up reconstruction serves as a base-line to infer grammar rules. These rules are then included into a data-driven top-down reconstruction. This approach copes with missing data and generates synthetic facades. However, all these methods - relying only on Lidar data - have the same limitation: using Lidar data, additional object information such as colors, textures,

etc. are difficult to recover compared to methods using image data. This limitation is intrinsically linked to the nature of the Lidar data.

1.3.5 Interactive and automatic multi-source methods

To improve the reconstruction and overcome the limitations mentioned before, one may fuse image information with the Lidar data. Fruh et al. [Fruh 2004] use both images and Lidar data to provide a fairly accurate textured 3D model of facades. Aerial imaging or DSM are combined with Lidar data generated from 2D vehicle-mounted scanners. The data is registered onto each other using information on the vehicle motion and on the scanner orientation. A non-convex optimization method computes the solution for the registration. In a similar fashion, Li et al. [Li 2011] propose an interactive scheme to merge 2D images and 3D point clouds. The result is an enhanced textured 3D model. Extracting vertical layer of buildings, the method successfully detects repetitive patterns on facades to assist the reconstruction of buildings with missing data. Zebelin et al. [Zebedin 2008] combine aerial image data with Lidar data to automatically generate multiple Level of Details (LOD) of the scene. An over-segmented 2D map is generated based on the principal directions of the buildings and an MRF optimization approach is used to merge the segmented cells together. An interesting aspect of their work is the possibility to set different merging coefficients during the optimization, thus controlling the LOD of the reconstruction. The aerial images are finally applied to texture the 3D meshes of the urban scenes.
The integration of multiple sources of data is not always possible and registration between the data remains a nontrivial task (see the work of Mastin et al. [Mastin 2009] or Wang et al. [Wang 2011] for example).

It is thus important to note that independently of the types of data used, the problem of reconstructing very dense and detailed large scenes enriched with semantics is still an open problem. A framework dealing with a huge amount of data in a timely manner, while generating compact, detailed, watertight and semantic-aware urban scenes is still a challenging task.

1.4 Evaluation of urban reconstruction methods

A particular challenge in urban scene reconstruction is to evaluate the reconstruction quality in term of compactness and accuracy. Despite the need of a general benchmark for urban scene reconstruction, no satisfying solution has yet been proposed. A very recent benchmark was established by the ISPRS community [Rottensteiner 2012] but it is not widely used yet: it is difficult to find a common basis among the multitude of solutions proposed in the literature. Various aspects explain why no common basis has been found yet:

- **Diversity of measurements:** As shown, multiple data is adequate for accurately representing urban sceneries. The diversity in types of data (airborne Lidar data, terrestrial Lidar data, airborne images and MVS meshes) and quality (density of Lidar point cloud, resolution of the images) makes it difficult to compare methods which use different data.
- **No ground truth:** In urban reconstruction, the ground truth is rarely available. Indeed, urban scenery evolves very quickly over the years and it is very difficult to have access to updated and accurate 3D synthesized models of the buildings or to their architectural maps with precise scale.
- **Availability of the data:** Because of the cost of data acquisitions, data is often protected by copyright and thus not publicly available. Thus, many research projects in the field operate on their own data which make comparison between them very delicate since the data is different and often not available.
- **No common evaluation technique:** Finally, there are many ways of evaluating the reconstructed model. Many metrics are used in the literature such as the Euclidean distance, the Hausdorff distance, etc. Moreover, the distance error is not the only important criterion to consider: the resulting topology and the compactness are also important. However, many papers use their own relevant criterion which makes it difficult to compare the methods with each other since the measurements from experiments are not evaluating the same error.

These are some reasons that explain why no common benchmark has been adopted by the urban reconstruction community yet: a benchmark that does not favor a method compared to others is difficult to create.

During the experiments in this thesis, we did not find any appropriate benchmark. Thus, we compared our approaches with existing methods in the field which, to the best of our knowledge, have been shown to generate good reconstruction of urban scenes. Special care is taken to compare our approaches with methods of similar type and quality of data. For example, our new optimization method presented in Chapter 3 is compared to existing methods and particularly the method of Descombes et al. [Descombes 2009]. Our building reconstruction from airborne Lidar data presented in Chapter 4 is compared with the method of Zhou et al [Zhou 2010] which uses the same type of data of similar quality. Concerning our building reconstruction from MVS meshes also presented in Chapter 4, we first remove all geometrical defects from the MVS meshes since we need 2D manifold meshes for fair comparison with the method of Cohen-Steiner et al. [Cohen-Steiner 2004].

1.5 Goals and proposed methods

1.5.1 Problem statement

Despite substantial effort (see Section 1.1 and Section 1.3) to tackle the difficult problem of automatically and accurately reconstructing large urban scenes from diverse input data (MVS meshes, Laser-based data, etc.), it remains an open problem. First, a simplification to overcome this challenge is to reconstruct the buildings as a set of aligned boxes with surfaces aligned with three dominant directions (known in the literature as the "Manhattan world" assumption [Denis 2008, Furukawa 2009, Vanegas 2010a]). This simplification makes the problem more tractable but the reconstruction does not always fit the reality. Such modeling assumptions often over-simplify the scenes. Therefore, we will design methods that are not based on this assumption. Second, we showed that user interactions help to properly reconstruct detailed buildings and can enrich the reconstruction with semantics, but it is limited to a single building

at a time. This limitation is not acceptable for modeling a full city-scale urban scene composed of thousand of buildings. Thus we will design fully automatic approaches. Third, we explained in Section 1.1 that two different types of data have interesting properties for solving the difficult problem of automatic urban scene reconstruction: MVS data provides coherent representation of the scene with important details but with only moderate accuracy and no semantics while Lidar data provides extremely precise information of the scene without strong coherence or photometric information on the objects scanned. Both data types are getting more and more common and cheaper to acquire, bringing to light a new problem: the huge quantity of measurement and its heterogeneous quality makes the need for novel scalable and robust urban reconstruction methods even more evident. Finally, the existing methods for urban reconstruction hardly provide semantical information on the scene which limits the possible applications of such reconstructions.

Therefore, it is obvious that approaches which are semantic-aware, flexible towards different types of data, fully automatic, except from the "Manhattan world" assumption and capable of handling large quantity of complete and incomplete information are required. Obtaining the semantics of the urban scene will help for the reconstruction and widen the range of potential applications. Solving these problems is the goal of this thesis.

1.5.2 Scope of our research

To address these problems, we restrain our research to fully automatic methods for large scene reconstruction. We contribute to the state-of-the-art by proposing methods for urban scene understanding and reconstruction. Specifically, the scope of our research covers the following properties:

- **Automatic processing:** The scene must be generated without user interactions. This criterion makes the solutions scalable and robust.
- **Efficiency:** The large-scale urban reconstruction should take reasonable time and be able to run on a personal computer. This criterion makes the solution usable with a common computer.

- **Semantic labeling:** The scene reconstruction will be enriched by semantic labeling such as *building*, *ground*, *vegetation*, etc. This information will be used for defining different strategies of urban reconstruction. This semantics widen the range of possible applications for our methods in public (urban planing), private (cell phone company) and military (Unmanned Aerial Vehicle flight planing) domains.
- **Accuracy:** The reconstruction has to be evaluated to guarantee precise urban reconstruction. It is important to provide reconstruction as accurate as possible to widen the range of applications (e.g. simulation, urban planing).
- **Sparse representation:** The reconstruction has to be extremely light-weight (i.e. the mesh has a minimum number of facets), while still watertight and accurate. Special care is taken to represent a maximum of details while preserving the topology of the buildings and generating a simplified representation of the scene.

Note that although both types of data are used (MVS data and Lidar data), none of them are used simultaneously as described in the 2D-3D fusion methods in Section 1.3. Moreover, converting image data to MVS meshes is achieved thanks to the method of Hiep et al. [Hiep 2009] (no further details on this process are given in this thesis).

1.6 Overview

We presented in Chapter 1 the different methods used in the literature for urban scene modeling and reconstruction. Two types of data, i.e. Lidar and MVS data (obtained from image data) have interesting properties and all accurate information needed for urban scene reconstruction. Additionally, a number of problems are identified, which we described shortly. Considerations are brought to light to specify the scope of this research and to clarify the work done during this thesis.

Chapter 2 describes the elements representing the urban scene. The various urban objects and categories are discussed resulting in the definition of classes of elements. The characteristics of these classes are successively described. This set-

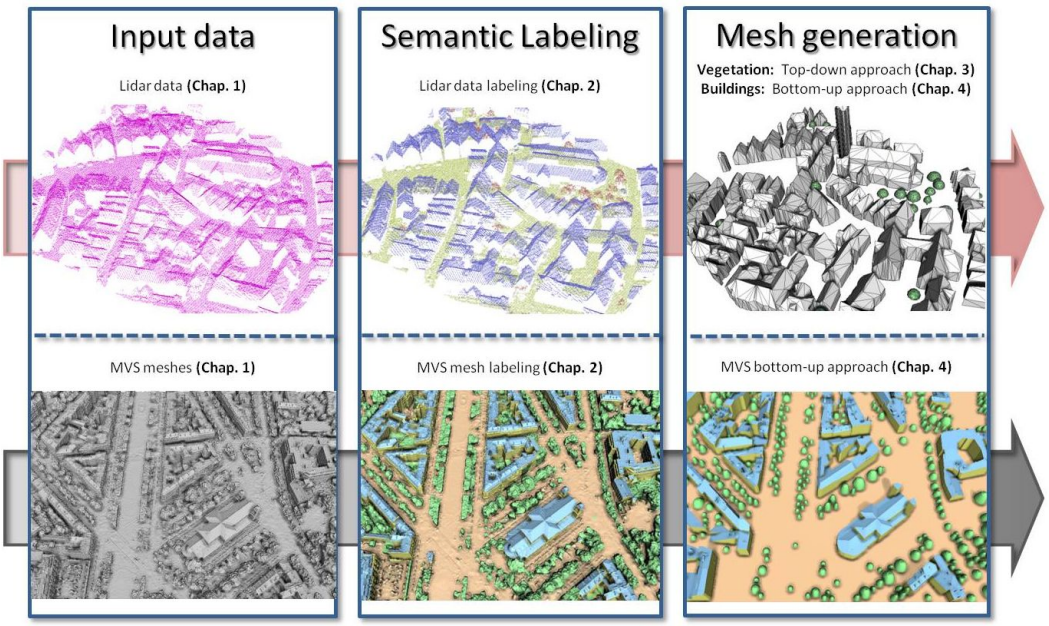


Figure 1.8: Pipeline illustration of our approaches and reference chapters within this thesis.

tles the base for the definition of relevant discriminative features used during the semantic labeling. The two labeling methods with Lidar data and MVS meshes are finally described. Examples of semantic labeling are given in Fig. 1.8, center-column. Parts of labeling methods presented in this chapter were published in our paper [Verdie 2011].

Chapter 3 presents an approach for detecting template-primitives in large scenes. The method determines the number of objects in the scene as well as their respective parameters. This new method enables reliable results in an acceptable computational time. We compare with the state-of-the-art methods and highlight the benefits of this approach. Among other experiments, we give an example for detecting template-based trees within Lidar data (Fig. 1.8, top-right corner). Parts of this method were published in our papers [Verdie 2012, Verdie 2013].

Chapter 4 presents approaches for urban mesh simplification and modeling. Two approaches are presented (for Lidar data and MVS meshes) with emphasis on the building modeling part which is the most challenging problem. Both approaches use

labeled data from the methods presented Chapter 2. Parts of these methods were published in our paper [Verdie 2011].

Chapter 5 concludes this thesis and presents some interesting future research directions.

Chapter 2

Urban scene description and understanding

2.1 Introduction

While most researchers focus on building reconstruction, which is probably the most complex element in an urban scene, we would like to emphasize the importance of other scene elements too often neglected. At least three distinct classes of elements can be detected in urban scenes: *building*, *vegetation*, and *ground*. These classes are present in most urban scenes, under different characteristics depending on the acquisition method (Lidar or MVS). Processing the MVS meshes, one can subdivides the *building* class into two more meaningful subclasses, i.e. *roof* and *facade*.

In this section, we first give a general description of each class. This is done by considering two different aspects for each class: (1) a general description and justification of the importance of this class in urban scenery, and (2) assumptions which are adopted during the 3D modeling of the class. In this chapter, the classification methods used to detect each class are described. A description of relevant geometric features and similarity functions is given (Section 2.2.1 and Section 2.3.1), followed by the definition of energy functions. Various experiments are then performed to validate the methods (Section 2.4.1 and Section 2.4.2). Note that the assumptions described here for the 3D modeling are developed later in Chapter 3 and Chapter 4.

We now consider each class of elements and give its properties.

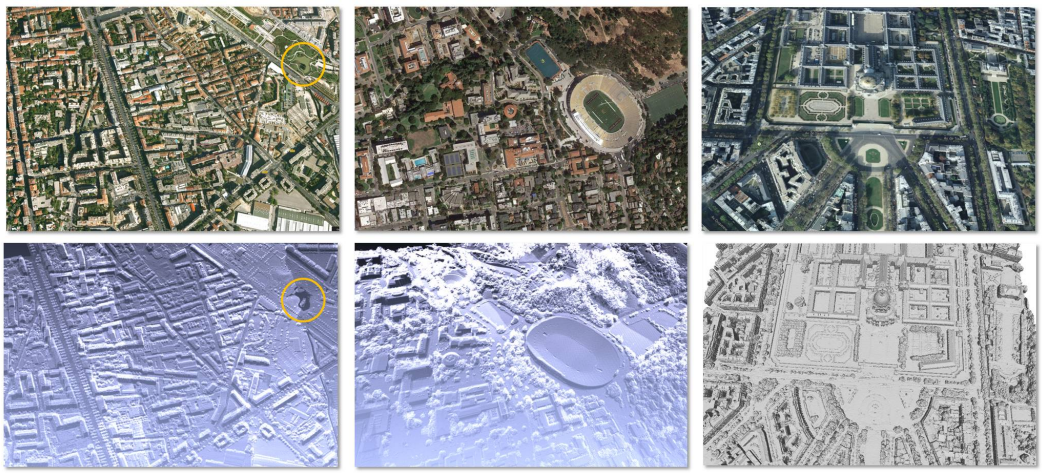


Figure 2.1: Aerial pictures and corresponding data of Marseille, France (left), Berkeley down-town, USA (middle), and Paris, France (right). The four classes of interest (*building*, *vegetation*, *ground*, and *small structure*) are easily recognizable. Notice the pond in the Lidar data of Marseille (left) that is visible as a "hole" in the data, the hilly ground of Berkeley down-town (center), and the uncommon building structures of Paris (right). Images courtesy of *Google* [Google 2013].

Building. (1) Analysis: This class of elements is the core of the urban scene representation. Whereas other classes have been neglected in many methods, *building* elements have always been included in the urban representation as a crucial element of the scene. This element constitutes the most challenging part of the reconstruction, since cities have buildings of various and unique shapes.

(2) Reconstruction: As explained before, a hypothesis initially adopted in the literature was to model the buildings as a set of bounding boxes of similar orientation. This approaches, generally known as "Manhattan world", was subject to many papers for American city representation. This approach unfortunately over-simplifies the complex and tedious problem of urban scene reconstruction, particularly for old European centers. For example, on Fig. 2.1, right, one can see unique and complex multi-wing building structures with a dome in the center of Paris: this complex building can not possibly be well approximated with the "Manhattan world" representation.

Vegetation. (1) Analysis: Numerous studies have illustrated the social, environmental, and economical importance of urban green spaces. For example, Wendel et al. [Wendel 2001] showed a correlation between the residents well-being and the proximity of green spaces within 3km radius. Similarly, Maas et al. [Maas 2006] stress a rise of demand from the population for green common spaces that provide sufficient isolation from the noisy, stressful urban environment. The cities, therefore, need sporadically distributed green spaces as a way to provide quiet areas for the population. Consequently, an urban scene will likely have green spaces and vegetation since the well-being of the citizens depends on it. This illustrates the important role played by the vegetation inside cities. Thus, detecting and reconstructing this class of elements (bushes, trees, etc.) is of clear importance for a proper representation of urban scene. Note that a simple glance at aerial pictures of cities (see Fig. 2.1 for Marseille, France (left), Berkeley down-town, USA (middle), and Paris, France (right)) is enough to notice the overwhelming presence of green spaces and vegetation within urban sceneries, regardless of its particular kind of urbanization.
(2) Reconstruction: Because the vegetation within urban scenes is controlled and constrained by urban elements surrounding it (roads, buildings, etc.), the vegetation in cities is mostly constituted of trees. Thus, during the reconstruction step, we assume that an appropriate modeling of the vegetation is obtained by only modeling the trees as simple geometric shapes, i.e. ellipsoidal shape, semi-ellipsoidal shape, and conic shape.

Ground. (1) Analysis: Contrary to the standard case of several American cities, many highly urbanized areas such as Lausanne, San Francisco, Marseille and many others, lay on hilly ground. However, the sloping nature of the underlying terrain is often neglected during urban modeling. In case of cities built on uneven terrain like, e.g. the areas cited above, the 3D modeling of the ground will certainly enhance the whole urban representation. For example, on the illustration Fig. 2.1, middle, the ground cannot be properly described by a rigid flat surface.
(2) Reconstruction: An accurate representation of an urban scene requires taking this class of elements into account in order to properly represent the local topology

as an irregular structure, i.e. a 2D manifold surface. One noticeable example of such assumptions is the work of Lafarge et al. [Lafarge 2012] where a grid mesh is first generated with each cell elevated to a locally coherent average altimetry. Then, a quadric mesh decimation simplifies the surface to create an accurate and fine ground representation.

Small structure. (1) Analysis: A last class of urban elements considered here is the class of small structures and clutters. This class gathers small details of the urban scene: objects on the ground such as cars, pedestrians, traffic lights, fences, etc., and objects representing small irregularities on the buildings such as chimneys, attics, dormer-windows, balconies and so on. The introduction of this class allows more flexibility during reconstruction by controlling the degree of details one wishes to obtain.
(2) Reconstruction: We note however, that the urban reconstruction is not strongly impaired by neglecting these objects. One may decide whether or not to include them within the reconstruction depending on the required degree of details. These details are usually ignored in the literature, with an exception of Satari et al. [Satari 2012] who focused on detecting and reconstructing dormers from aerial Lidar data.

As demonstrated, it is very important to consider the four classes of elements (*building*, *vegetation*, *ground*, and *small structure*) for an accurate reconstruction of the urban scenes. However the buildings remain the most challenging problem when considering complex and irregular building structures (see Paris on Fig. 2.1, right), the vegetation is omnipresent in urban scenes (see aerial pictures on Fig. 2.1) and the ground surface can be quite complex for hilly urban areas (see Berkeley downtown on Fig. 2.1, middle). Finally, small structures can be optionally included during the reconstruction to increase the level of details of the urban modeling.

Note that although many cities have rivers and water elements (like lakes and ponds), we choose not to consider the water elements (if any) during the labeling and reconstruction steps. To our knowledge, no one has yet worked on the detection and

representation of this element within the cities. The reason is that first, the current applications for automatic 3D urban scenes do not require an accurate detection of water area. Thus, this aspect is neglected and this element is often merged with the ground. Second, both types of data (MVS data and Lidar data) do not respond properly to the presence of water and this element is therefore represented poorly or not at all (see circle on Fig. 2.1). However, one can notice that "holes" inside the data may indicate an area with water. This could be a further direction for improving the completeness of the urban understanding.

2.2 Approach with Lidar data

2.2.1 Geometric features

We consider four features to be of interest concerning the classification of Lidar points as *building*, *vegetation*, *ground*, and *Small structure*. Note that due to our modeling approach, *Small structure* elements are from here on considered as Non-Classified elements and will be refereed as (*NC*).

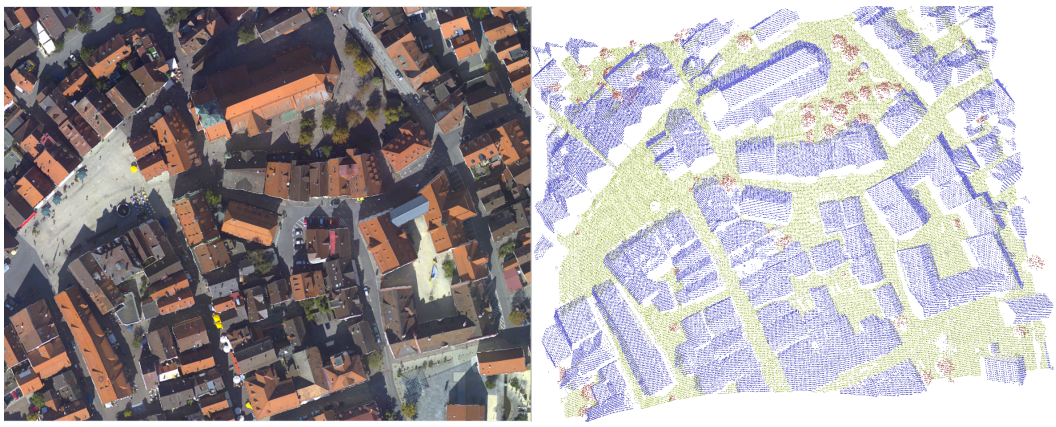


Figure 2.2: Point cloud classification- An aerial image (*left*) and the classified point set (*right*). Note that the four classes *building* (in blue), *vegetation* (in red), *ground* (in green), and *NC* (in white) are correctly separated. Images courtesy of *Google* [Google 2013].

Elevation. *Elevation* a_e is the height difference between the Digital Terrain Model *DTM* and the point height component. The DTM is obtained by histogram analysis on point altimetry within cells of size 50mx50m. This arbitrary cell size was chosen since it guaranties to have few points belonging to the ground in most of the urban scenery cases. Empty cells are filled up by extrapolating the cell values of its neighbor. This feature behaves in the following way: The further away a point is from the ground, the larger the a_e feature gets.

Local Non-planarity. *Local Non-planarity* a_p is the quadratic distance between the point and the least squared 3D plane computed with its spherical neighborhood. A spherical neighborhood of 2 meters radius is chosen. The orthogonal distance between the point and the best fitting plane of its neighborhood gives clue of the local planarity around the point. In other words, the more planar the neighborhood is, the lower the a_p feature gets.

Scatter. *Scatter* a_s is the local height dispersion of the point. When the number of echoes of the signal is known, a_s represents this number (i.e. a_s is between one and infinity). When the number of echo is unknown, $a_s = 1 + |H|$, with $|H|$ the mean curvature inside a spherical neighborhood. This means that the more irregular the neighborhood is, the higher the a_s feature gets.

Local Non-linearity. *Local Non-linearity* a_l is the quadratic distance between the point and the least squared 3D line computed with its spherical neighborhood. This means that the more linear the neighborhood is, the lower the a_l feature gets.

These attributes are weighted by four coefficients, α_e, α_p, α_s and α_l and then normalized in the interval [0,1]. Through a trial-and-error approach, we obtained the following formulation:

$$\begin{aligned} \beta_e &= \min\{1, \frac{|a_e|}{\alpha_e}\} \ , \ \beta_p = \min\{1, \frac{|a_p|}{\alpha_p}\} \\ \beta_s &= a_s^{\alpha_s} \ , \ \beta_l = \min\{1, \frac{|a_l|}{\alpha_l}\} \end{aligned} \tag{2.1}$$

Note that α_s is a positive weight coefficient smaller or equal to one. Note that from here on, we define the function $\bar{.} : [0,1] \longrightarrow [0,1]$ such as :

$$\bar{x} = 1 - x, x \in [0,1]$$

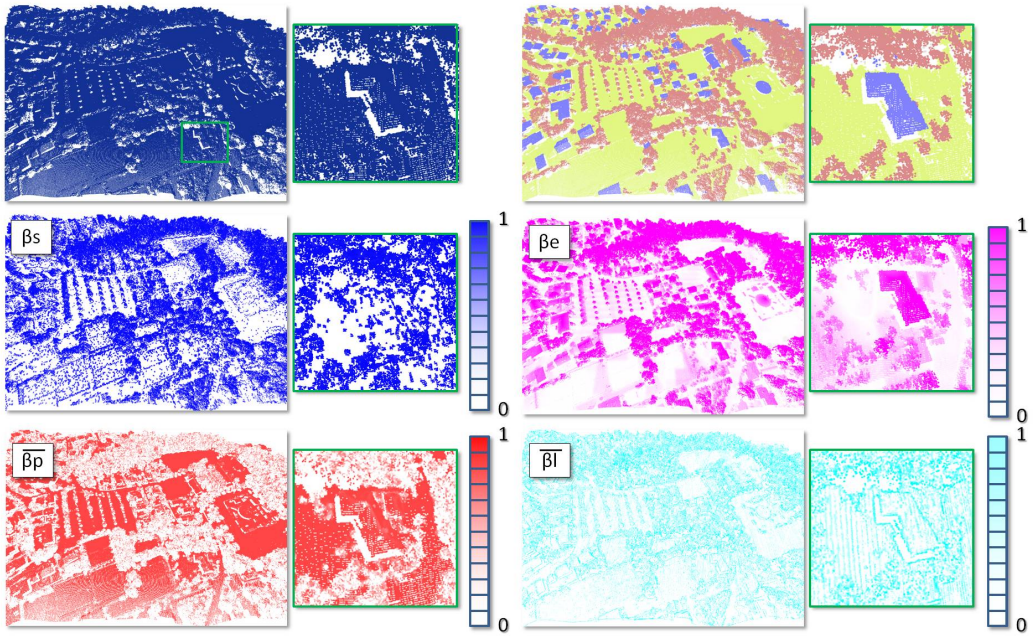


Figure 2.3: Point cloud classification- First row: Original Lidar data (*left*) and the classified point set (*right*). Note that the four classes *building* (in blue), *vegetation* (in red), *ground* (in green), and *NC* (in white) are correctly separated. The two following rows illustrate the features β_e, β_p, β_s, and β_p described above. Note that for a better visualization, we displayed $\bar{\beta_p}$ and $\bar{\beta_l}$.

On Fig.2.3, one can see that the features behave properly, i.e. β_s has a high value for scattered points (the vegetation points are visible), β_e has a high value at elevated points (building and vegetation points are visible), $\bar{\beta_p}$ has a high value for points having locally planar neighborhood (ground and building are visible), and $\bar{\beta_l}$ has a high value for points having locally linear neighborhood (the arris of the buildings are visible).

In the following sections, we define a similarity function $D_i : \mathbf{R}^3 \longrightarrow [0,1]$

for each class *building*, *vegetation*, *ground*, and *NC* which represents the inverse confidence that the point x_i belongs to the specific class (i.e. for each class c_l, we obtained a score between zero and one, zero means there is a strong confidence the point x_i belongs to the class c_l, one means otherwise). Although it seems counter-intuitive to define functions representing the inverse confidence, the rational is we will use such functions within a minimisation problem (described in Section 2.4.1).

2.2.2 *Vegetation* similarity function

Vegetation is often present in urban scenes and represents a key element whose detection constitutes a challenging problem. Indeed, vegetation objects such as trees, are often in very close proximity with roofs of buildings, and a clean segmentation of these two classes is ambiguous. In regard to small vegetation elements such as bushes, it is unclear whether to consider them as ground or vegetation. Moreover, chimneys and small clusters on the roofs are locally similar to bushes lying on the ground. This contributes to a possible mis-classification of vegetation with building elements.

We consider three features to be relevant concerning the classification of the points as *vegetation*. Vegetation points have high elevation $\beta_e \simeq 1$, high non-planarity $\beta_p \simeq 1$ and high scatter $\beta_s \simeq 1$.

The similarity function $D_i : \mathbf{R}^3 \longrightarrow [0, 1]$ is defined as:

$$D_i(x_i) = 1 - \beta_e.\beta_p.\beta_s$$

2.2.3 *Building* similarity function

We consider three features to be relevant concerning the classification of the points as *building*. Building points have high elevation $\beta_e \simeq 1$, low non-planarity $\beta_p \simeq 0$ and low scatter $\beta_s \simeq 0$.

The similarity function $D_i : \mathbf{R}^3 \longrightarrow [0, 1]$ is defined as:

$$D_i(x_i) = 1 - \beta_e.\bar{\beta}_p.\bar{\beta}_s$$

2.2.4 *Ground* similarity function

We consider three features to be relevant concerning the classification of the points as *ground*. Ground points have low elevation $\beta_e \simeq 0$, low non-planarity $\beta_p \simeq 0$ and low scatter $\beta_s \simeq 0$.

The similarity function $D_i : \mathbf{R}^3 \longrightarrow [0,1]$ is defined as:

$$D_i(x_i) = 1 - \bar{\beta_e}.\bar{\beta_p}.\bar{\beta_s}$$

2.2.5 *Small structure* similarity function

We consider three features to be relevant concerning the classification of the points as *NC*. NC points have high non-planarity $\beta_p \simeq 1$, high non-linearity $\beta_l \simeq 1$ and high scatter $\beta_s \simeq 1$. Note that the elevation feature has not been taken into account since *NC* elements can be either on buildings (such as chimneys) or on the ground (such as small bushes).

The similarity function $D_i : \mathbf{R}^3 \longrightarrow [0,1]$ is defined as:

$$D_i(x_i) = 1 - \beta_l.\beta_p.\beta_s$$

2.3 Approach with Multi-View Stereo data

2.3.1 Geometric features

In this work, MVS data is represented as 3D meshes. We consider four features to be relevant concerning the classification of the facets. The semantic is similar to the one presented in Section 2.2.1. However, our approach is different since the data has other characteristics.

Many methods have been proposed in the literature for segmenting synthetic meshes, e.g. [Kalogerakis 2010], but to our best knowledge, none is dedicated to urban scene classification from MVS meshes. We propose a solution for distinguishing specific classes of urban objects. Four classes are considered: *ground*, *vegetation*, *facade* and *roof*. The classification relies on simple but efficient geometric assumptions: *(i) ground* is characterized by locally flat surfaces located below the other

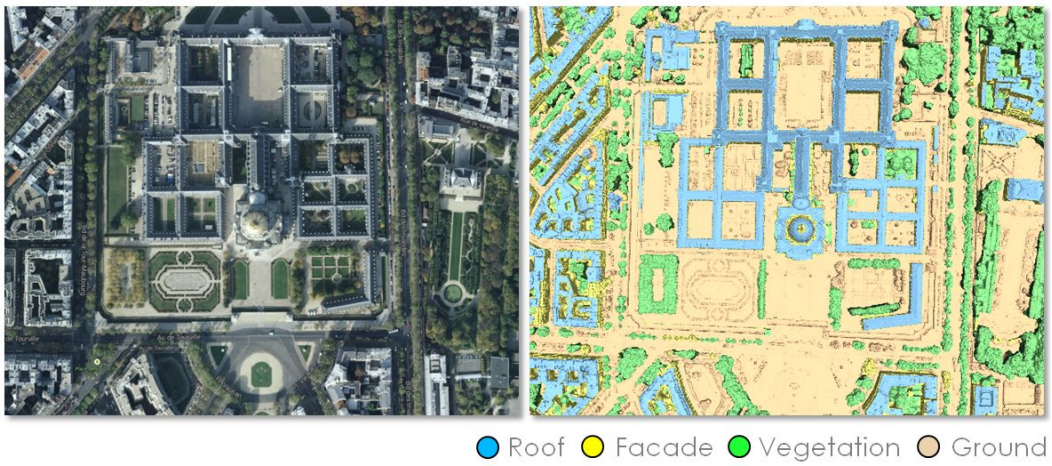


Figure 2.4: MVS classification- An aerial image (*left*) and the classified MVS data (*right*). Note that the four classes *building*, *vegetation*, *ground*, and *facade* are correctly separated. Image courtesy of Google [Google 2013].

classes, *(ii) vegetation* (and specifically trees) has irregular curved surfaces, *(iii) facade* elements are vertical structures connecting *roof* and *ground*, and *(iv) roof* elements are mainly composed of piecewise-planar surfaces.

Curvature-based clustering. As MVS meshes are extremely dense, classifying each triangular facet would lead to both high running time and regularization constraints with a restricted impact. Instead, groups of connected facets - that we call *f-clusters* - are considered. f-clusters are obtained by clustering the facets with similar mean curvatures [Botsch 2010]. A region growing is used to efficiently regroup facets; the propagation is relatively fast as the facet adjacency is known. This clustering procedure preserves the planar components as shown on Fig 2.5.

Discriminative attributes. Three different geometric attributes are computed from the f-clusters of the mesh for distinguishing the different classes of interest.

-The elevation attribute a_e of a facet estimates the relative height of its centroid with respect to the ground level. By considering a facet f_i whose centroid has a

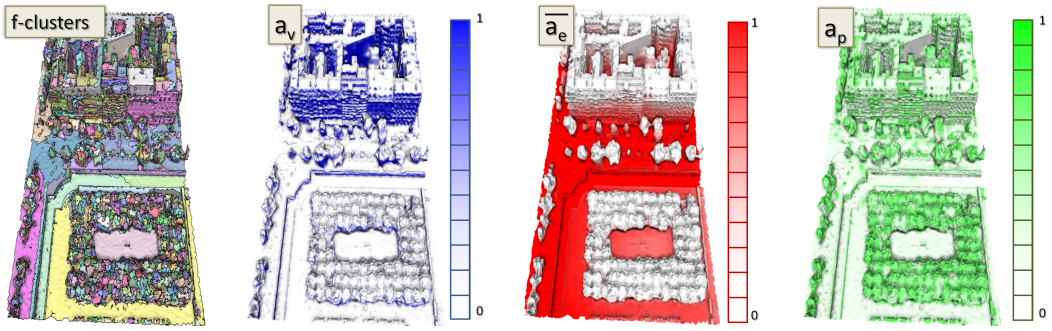


Figure 2.5: Clustering and discriminative attributes. The clustering procedure groups connected facets having similar mean curvatures (top left, each color corresponding to a f-cluster). The combination of verticality (a_v), elevation (a_e), and non-planarity (a_p) attributes allows us to distinguish classes of interest through geometric information only. For a better visualization, we display $\bar{a_e} = 1 - a_e$.

Z-value z_i, the elevation attribute is given by

$$a_e(f_i) = \sqrt{\frac{z_i - z_{min}}{z_{max} - z_{min}}} \tag{2.2}$$

where z_{min} (respectively z_{max}) represents the minimal (respectively maximal) Z-value of the facet centroids located in an extended spatial neighborhood. The width of this spatial neighborhood has to be large enough to meet ground components and small enough to be robust with respect to hilly grounds. In practice, this width is fixed to 30 meters. This attribute behaves in the following way: The further away a facet is from the ground, the higher the a_e attribute gets.

-The non-planarity attribute a_p measures the degree of non-planarity of the local neighborhood of a facet. It is expressed by

$$a_p(f_i) = \frac{1}{2}H(f_i) + \frac{1}{2}\frac{|\lambda_{min}|}{|\lambda_{max}|} \tag{2.3}$$

where $H(f_i)$ is the normalized Gaussian curvature of f_i, and λ_{max} (respectively λ_{min}) is the highest (resp. lowest) eigenvalue of the PCA over all the vertices of its f-cluster. Intuitively, this means that the more non-planar the neighborhood is, the higher the a_p attribute gets.

-The verticality attribute a_v measures the orientation of facet f_i with respect to the Z-axis.

$$a_v(f_i) = 1 - |\mathbf{n}_i \cdot \mathbf{n}_z| \tag{2.4}$$

where $\mathbf{n}_i$ is the facet normal, and $\mathbf{n}_z$ is the unit vector along the Z-axis. This attribute behaves in the following way: The more vertical the facet is, the higher the a_v attribute gets.

These three geometric attributes are computed for each facet, and are in value in the interval [0,1]. On Fig. 2.5, once can see that the features behave properly, i.e the f-clusters are clustered with similar mean curvature (e.g, planar area are within the same f-clusters), a_e has a high value for facets having high altimetry (building and vegetation facets are visible), a_v has high value for vertical structures (facades and trunks are visible), and a_p has high value for points having high curvature (vegetation and trees are visible).

In the following sections, we define a similarity function $D_i : \mathbf{F} \longrightarrow [0, 1]$ for each class *roof*, *vegetation*, *ground*, and *facade* which represents the inverse confidence that the facet f_i belongs to the specific class (i.e. for each class c_l, we obtained a score between zero and one, zero means the facet f_i belongs to the class c_l, one means otherwise). Although it seems counter-intuitive to define functions representing the inverse confidence, the rational is we will use such functions within a minimisation problem (described in Section 2.4.2).

2.3.2 *Vegetation* similarity function

We consider two features to be relevant concerning the classification of the f-clusters as *vegetation*. Vegetation f-clusters have high non-planarity $a_p \simeq 1$ and low verticality $a_v \simeq 0$.

The similarity function $D_i : \mathbf{F} \longrightarrow [0, 1]$ for facet f_i of a f-cluster is defined as:

$$D_i(f_i) = 1 - \bar{a_v}.a_p$$

2.3.3 *Roof* similarity function

We consider three features to be relevant concerning the classification of the f-clusters as *roof*. Roof f-clusters have low non-planarity $a_p \simeq 0$, low verticality $a_v \simeq 0$, and high elevation $a_e \simeq 1$.

The similarity function $D_i : \mathbf{F} \longrightarrow [0,1]$ for facet f_i of a f-cluster is defined as:

$$D_i(f_i) = 1 - \bar{a_p} \cdot \bar{a_v} \cdot a_e$$

2.3.4 *Facade* similarity function

We consider two features to be relevant concerning the classification of the f-clusters as *facade*. Facade f-clusters have low non-planarity $a_p \simeq 0$ and high verticality $a_v \simeq 1$.

The similarity function $D_i : \mathbf{F} \longrightarrow [0,1]$ for facet f_i of a f-cluster is defined as:

$$D_i(f_i) = 1 - \bar{a_p} \cdot a_v$$

A particularity of MVS meshes compared to aerial Lidar data is the presence of vertical structures that can possibly represent building facades. However, the similarity function $D_i(f_i)$ thereby described does not discriminate vertical flat structures of buildings (e.g. facades) and vertical flat structures of vegetation (e.g. trunks of large trees). Therefore, a simple post processing test is performed to relabel facade f-clusters as vegetation when required. This step is further described in Section 2.4.2.

2.3.5 *Ground* similarity function

We consider three features to be relevant concerning the classification of the f-clusters as *ground*. Ground f-clusters have low elevation $a_e \simeq 0$, low non-planarity $a_p \simeq 0$ and low verticality $a_v \simeq 0$.

The similarity function $D_i : \mathbf{F} \longrightarrow [0,1]$ for facet f_i of a f-cluster is defined as:

$$D_i(f_i) = 1 - \bar{a_p} \cdot \bar{a_v} \cdot \bar{a_e}$$

2.4 Experiments

Once a similarity function is defined for each class of elements, a non-supervised energy minimization problem is formulated to label the data. In this section, we will present an energy formulation and labeling results for Lidar and MVS data.

2.4.1 Semantization of Lidar data

A non-supervised energy minimization problem is defined to classify the point cloud. Graph-cuts with α-expansion [Boykov 2001] is used to reach a solution close to the global optimum.

The energy is defined as:

$$E(x) = \sum_i D_i(x_i) + \gamma \sum_{\substack{i=1..n \\ j \in N_i \\ i<j}} \delta(x_i, x_j) \tag{2.5}$$

where $\delta(x_i, x_j)$ is the pairwise interaction between the label x_i of the point i and the label x_j of the point j, defined as the standard Potts model with γ parameter. N_i is the neighboring points of the point i and $D_i(x_i)$ is the partial data term defined as a combination of the four weighted geometrical attributes:

$$D_i(x_i) = \begin{cases} \beta_e.\beta_p.\beta_s & if \quad \mathrm{x_i} = ground \\ \bar{\beta}_e.\bar{\beta}_p.\bar{\beta}_s & if \quad \mathrm{x_i} = vegetation \\ \bar{\beta}_p.\beta_s.\beta_l & if \quad \mathrm{x_i} = NC \\ \bar{\beta}_e.\beta_p.\beta_s & if \quad \mathrm{x_i} = building \end{cases} \tag{2.6}$$

The initial configuration is chosen as $\arg_x \min \sum_i D_i(x_i)$. One can see that our energy function fits the requirements for using Graph-cuts algorithm. The weight parameters α_e, α_p, α_s and α_l were set to 6, 0.5, 0.05 and 0.25 respectively. These parameters, which were obtained through a trial-and-error approach, could be improved by a learning method, but we notice a stable behavior of our system for a wide range of data and do not think it is necessary to make the system heavier. The resulting classification is illustrated on Fig. 2.2 and Fig 2.6.

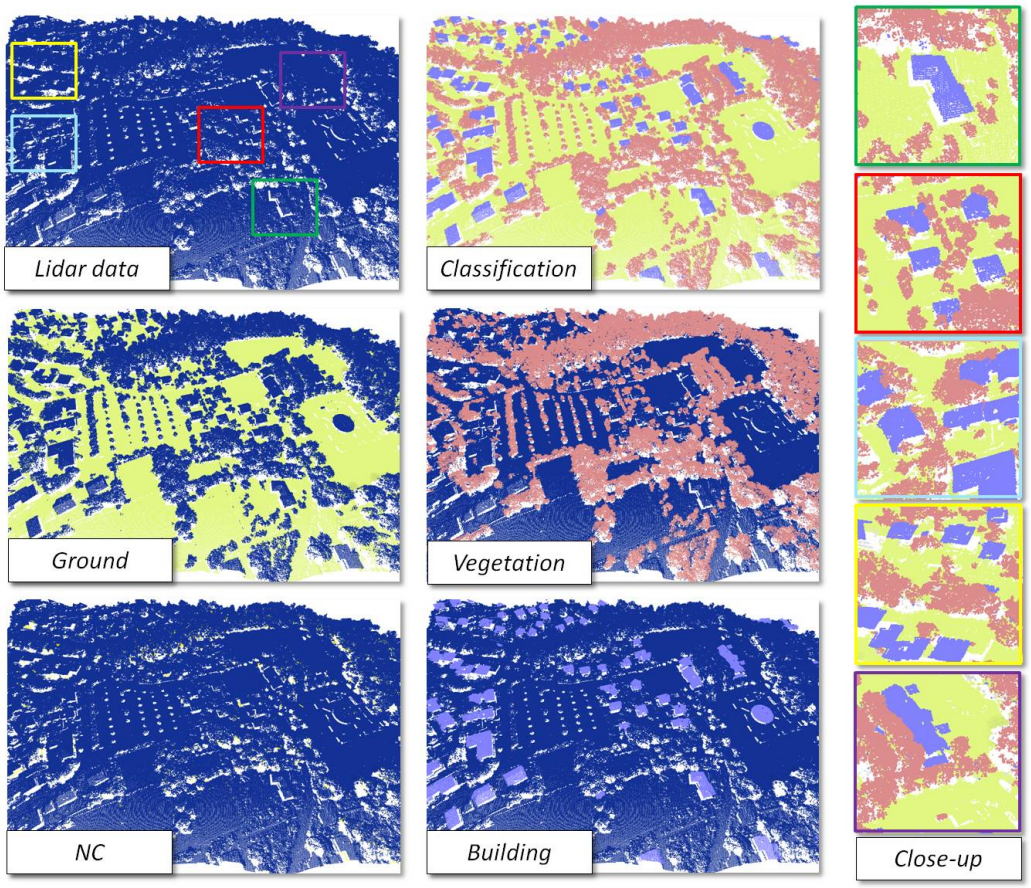


Figure 2.6: Result of Lidar classification: First column, top: Initial point cloud. Middle: *Ground* class. Bottom: *NC* class. Second column, top: the different classes (*Vegetation*, *Building*, *Ground*, *NC*) for the classified Lidar data. Middle: *Vegetation* class. Bottom: *Building* class.

2.4.2 Semantization of MVS data

Labeling. A Markov Random Field (MRF) with pairwise interactions is used to label each f-cluster by one of the four classes of interest {*ground*, *vegetation*, *facade*, *roof*}. The quality of a label configuration l is measured by the energy U of the standard form:

$$U(l) = \sum_{i \in S} D_i(l_i) + \gamma \sum_{\{i,j\} \in E} V_{ij}(l_i, l_j) \tag{2.7}$$

where D_i and V_{ij} constitute the unary data term and propagation constraints respectively, balanced by the parameter $\gamma > 0$. S denotes the set of f-clusters. E represents the pairs of adjacent f-clusters, two f-clusters being adjacent if they have at least one common edge in the input triangular mesh.

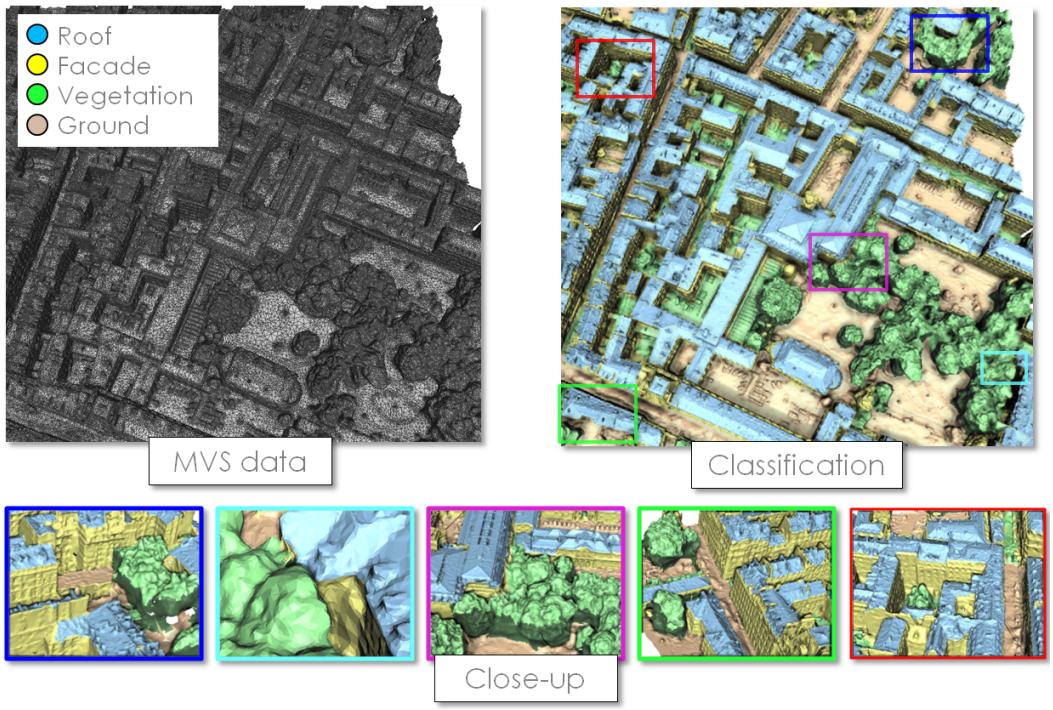


Figure 2.7: Semantic labeling. Four types of urban objects are identified from MVS meshes: roof, facade, ground and vegetation. Both the regularizing term of the energy and the correction rules bring spatial consistency to the labeling. Note in particular how roofs and facades are accurately separated, as well as trees glued to facades (see close-ups).

In the subsequent labeling, the geometric attribute of a f-cluster is expressed as the area-weighted sum of the attributes of its facets. The data term D_i is formulated as a combination of the discriminative geometric attributes described above weighted

by the area A_{f_i} of each facet of the f-cluster i. It is given by

$$D_i(l_i) = \sum_{f_i} A_{f_i} \times \begin{cases} (1 - \bar{a_p} \cdot \bar{a_v} \cdot a_e) & \text{if } l_i = ground \\ (1 - a_p \cdot \bar{a_v}) & \text{if } l_i = vegetation \\ (1 - \bar{a_p} \cdot a_v) & \text{if } l_i = facade \\ (1 - \bar{a_p} \cdot \bar{a_v} \cdot \bar{a_e}) & \text{if } l_i = roof \end{cases} \tag{2.8}$$

where $\bar{a_.} = 1 - a_.$. The pairwise interaction V_{ij} between two adjacent f-clusters i and j favors local smoothness. It is expressed by

$$V_{ij}(l_i, l_j) = C_{ij} \cdot \cos \alpha_{ij} \cdot 1_{\{l_i \neq l_j\}} \tag{2.9}$$

where $1_{\{.\}}$ is the characteristic function, C_{ij} is the connection length between f-clusters i and j (computed as the sum of the edges common to the two f-clusters), and $\alpha_{i,j}$ represents the angle between the estimated normals of the f-clusters i and j. Note that, as the unary data term and pairwise potential are weighted by the f-cluster areas and contour lengths, this energy formulation can be seen as a facet-based energy with grouping constraints.

An approximate solution of this energy minimization problem is found using the α-β swap algorithm [Boykov 2001]. In our experiments γ has been set to 0.5. One can imagine learning this parameter from a training dataset, but we noticed in practice that it is stable on a wide range of input meshes.

Local correction. Because of the geometric and topological defects contained in the meshes, three types of errors frequently occur from the labeling when dealing with complex scenes: *(i)* roof superstructures such as chimneys or dormer-widows can be wrongly labeled as *vegetation*, these elements being too small and too irregular to be locally planar, *(ii)* potential vertical components of large trees can be labeled as *facade*, and *(iii)* whole *vegetation* elements can be labeled as *facade*. In order to correct these labeling errors, three rules based on simple urban assumptions are formulated.

- *Rule 1.* f-clusters labeled as *facade* and adjacent to f-clusters labeled as *vegetation* and *ground* must be turned to *vegetation*.

- *Rule 2.* f-clusters labeled as *vegetation* and only adjacent to f-clusters labeled as *roof* must be turned to *roof*. This assumption is realistic considering large trees cannot be found on top of roofs.
- *Rule 3.* f-clusters labeled as *facade* and only surrounded by f-clusters labeled as *ground* (no roof around) must be turned to *vegetation*.

As illustrated on Fig. 2.7 and Fig. 2.8, these three simple rules give a coherent labeling in presence of small irregular roof superstructures and trees with cylindrical shapes. The labeling procedure allows the scene to be decomposed into clusters of urban objects. This decomposition highly reduces the complexity of the subsequent operations as a block of buildings [1] or a group of trees can be reconstructed independently in the scene. Note finally that looking carefully at Fig. 2.7 and Fig. 2.8, one can notice two variants of green for the labeling, i.e. light green and dark green. While these two colors are for the same class of elements *vegetation*, this distinction is made to illustrate the effect of *Rule 1* which turns *facade* elements in *vegetation* elements. The elements labeled as *vegetation* during the optimization method are shown in light green whereas these labeled by *Rule 1* are shown in dark green.

2.4.3 Comparison

It is difficult to compare the labeling results of these two methods quantitatively since the methods process two different types of data with different characteristics. Moreover, there is no ground truth available for either data type. Thus, we are evaluating and comparing the results of both methods visually, and we discuss the advantages and limitations of the methods by comparison.

Visually, both labeling for Lidar data and MVS data are convincing and detailed. The MVS labeling is more informative since *facade* and *roof* elements are distinguished while only *building* elements are detected with Lidar data. Thus, the resulting labeling of MVS data provides better understanding of the scene.

1. An isolated building or a block of buildings can be easily extracted by searching for all the connected f-clusters labeled as *roof* and *facade*.

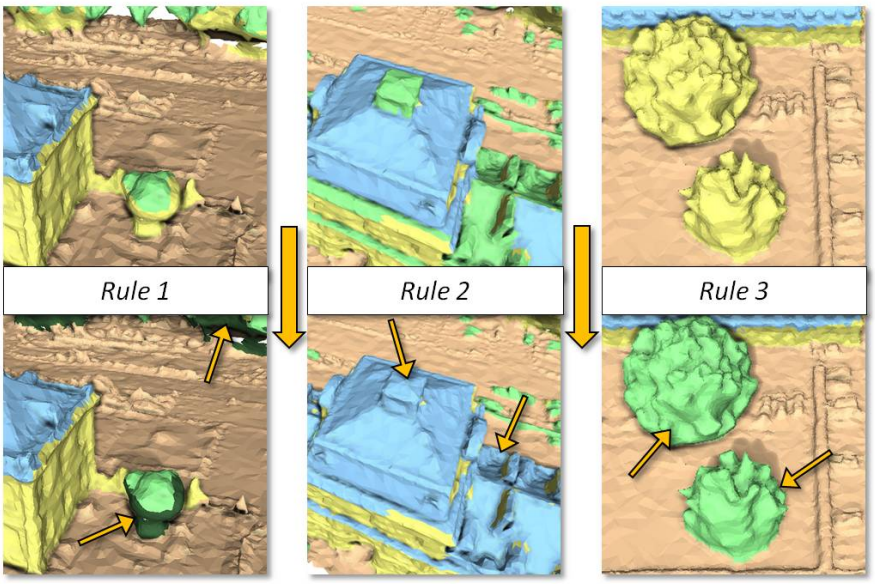


Figure 2.8: Local correction: Effect of the rules on the semantic labeling. Top row: Labeling without rules, which presents small inconsistencies. Bottom row: Correction of the labeling by applying one specific simple semantic rule. The corrections are indicated by orange arrows. In this example, only one rule is active at the time to illustrate its effect on the labeling. In practice, all of rules are successively applied.

Concerning *vegetation* class, the detection accuracy (true positive) with Lidar data is better, especially when the number of returned pulses (echo) is known. However, the number of mis-classifications (false positives) is higher with Lidar data as well since chimneys and small clutters are labeled as *vegetation* too. In the case of MVS data, this mis-classification of *building* as *vegetation* is corrected during the post-processing step (*Rule 1*).

The *ground* classification provides good results in both cases. Both cases rely on a local histogram analysis of the altimetry, which implies that we expect to have a *ground* element within the neighborhood considered. During all the experiments, this assumption was met by considering a neighborhood of size between 30 and $50m^2$. However, we think experiments on extremely hilly terrain are yet to be done to fully validate the approach. When comparing the resulting ground elements from

Lidar and MVS data, we noticed that the method for MVS data is slightly less accurate, but the difference is not significant enough to tell whether this is due to the method or the type of data.

Finally, both methods are not robust when unknown classes are present in the scene. For example, labeling methods with Lidar or MVS data will incorrectly label a bridge in the urban scene partially as *building* element, partially as *ground* element. A solution would consist in adding the *bridge* class to the labeling and to propose discriminative features for this class. Note however that this class of elements is usually uncommon in an urban scene.

2.5 Summary

We have described two methods for urban scene semantics and understanding from aerial Lidar data and MVS meshes.

2.5.1 Approach with Lidar data

With this approach, we obtained a labeling of point clouds as *building*, *vegetation*, *ground* and *small element*. This labeling improves the scene understanding and helps to reconstruct the scene (this is further developed in Chapter 4). During the experiments, we noticed the results are more accurate when the number of echoes is specified in the Lidar data. The *vegetation* labeling becomes more robust since most of the multiple-pulse returns within Lidar data are vegetation elements. Contrary to the labeling strategy for MVS data where two steps are needed (see sections 2.4.2), the labeling of the Lidar data does not require post-processing step. This may indicate the labeling method with Lidar data is more robust and general.

Limitations This approach remains globally less accurate than its MVS counterpart. Indeed, we noticed that chimneys and other small building elements are sometimes mis-labeled as *vegetation*. This is rectified by using a large value for the piecewise term of the energy function (γ parameter, formula (2.5)). As a side-effect, this approach oversmoothes the details, which results in oversimplified labeling of the

urban scene. In addition, this method cannot distinguish *facade* and *roof* elements from *building* mostly due to intrinsic limitations of the data acquisition method (see Section 1.2.2).

2.5.2 Approach with MVS data

With the developed approach, we obtain a labeling of MVS meshes as *roof*, *facade*, *vegetation* and *ground*. To the best of our knowledge, urban MVS mesh semantics has never been proposed before. Since MVS data is more complete and has vertical structures represented, the *building* class has been split into two subcategories, i.e. *roof* and *facade*. The resulting urban scene labeling is performed in two steps, a standard MRF multi-label approach followed by a cascade of simple semantic rules. This leads to a reliable and useful MVS mesh semantic that yields a simplification process described in Chapter 4. This semantics can also be used for texture mapping, scene understanding, and so on.

Limitations Our algorithm is first limited to four classes of urban objects. Even if it is sufficient for accurate modeling of the entire districts in Paris (see Fig. 2.7), more urban components could be taken into account, such as bridges (e.g Venise) or elevated roads (e.g Tokyo). Second, this approach is less general than the one for Lidar data since the three semantics rules are applied during a post-processing step. Finally, in the current state, few parameters have been chosen manually. A more general and robust approach could be achieved by automatically learning these parameters.

Since the presented labeling methods enable classification of the urban scene elements, we are now able to adopt appropriate reconstruction strategies for each of them. Thus, in the following two chapters, we will describe methods for reconstructing specific elements of an urban scene. First, in Chapter 3, a top-down approach is presented for detecting and modeling elements such as trees in aerial Lidar point clouds. Our contribution is a new optimization method that can be applied to various problems: from detecting objects in 3D point clouds to 2D elements in images.

Second, in Chapter 4, bottom-up approaches for reconstructing elements from urban scenes represented by Lidar point clouds and MVS meshes are described. With Lidar point clouds, a method for automatic generation of complex building meshes is described relying on the point cloud classification technique proposed above. With MVS meshes, independent approaches are described to automatically reconstruct urban scene elements, i.e., buildings, vegetation and ground.

Chapter 3

Method for geometric object detection in large scenes

In this chapter, we present a general top-down approach for object detection in large scenes. Among multiple applications such as bird and cell counting (Section 3.3.1.1), road and river detection (Section 3.3.1.2), images labeling (Section 3.3.3), our approach also deals with aerial Lidar data to detect parametric 3D templates of trees within urban, mixed, and mountainous area (Section 3.3.2). We will show that this approach which relies on Markov Point Processes (MPP) is general enough to be applicable for a large variety of problems. Throughout this chapter, we will give a general background and related works on MPPs, present former and novel optimization methods for them, and give various examples of problems that can be solved with these approaches.

Note that detection of trees inside Lidar data will be considered as a particular case for our approach. In this case, one can optionally use the labeling technique proposed in Chapter 2. This decreases the problem complexity since only a small subset of the point cloud need to be processed, making the approach very efficient.

3.1 Introduction

Markov point processes are probabilistic models introduced by [Baddeley 1993] to extend the traditional Markov Random Fields (MRF) by using an object-based formalism. Indeed, Markov point processes can address object recognition problems by directly manipulating parametric entities in dynamic graphs, whereas MRFs are restricted to labeling problems in static graphs.

These mathematical tools exploit random variables whose realizations are configurations of parametric objects, each object being assigned to a point positioned in the scene. The number of objects is itself a random variable, and thus must not be estimated or specified by a user. Another strength of Markov point processes is their ability to take into account complex spatial interactions between the objects and to impose global regularization constraints in a scene. A point process is usually specified by three key elements:

Some parametric objects. They can be defined in discrete and/or continuous domains. They usually correspond to geometric entities, *e.g.* segments, rectangles, circles or planes, but can more generally be any type of multi-dimensional function. The complexity of the objects directly impacts on the size of the configuration space.

An energy. It is used to measure the quality of a configuration of objects. The energy is typically defined as a combination of a term assessing the consistency of objects to the data, and a term taking into account spatial interactions between objects in a Markovian context.

A sampler. It allows the search for the object configuration minimizing the energy. As the configuration space is of variable dimension and the energy is usually non-convex, Monte Carlo based samplers capable of exploring the whole configuration space are required, in most cases a Markov Chain Monte Carlo (MCMC) algorithm [Hastings 1970, Green 1995, Liu 2001].

3.1.1 Related works

The growing interest in these probabilistic models is motivated by the need to manipulate parametric objects interacting in complex scenes. Many works relying on point processes have been recently proposed to address the variety of image and vision problems listed below.

Population counting. [Descombes 2009] propose a point process for counting populations from aerial images, each entity being captured by an ellipse. [Ge 2009] present a point process for a similar application, but dedicated to crowd detection

from ground-based photos, for which objects are defined as a set of body shape templates learned from training data. Multi-view images are used by [Utasi 2011] to detect people by a point process in 3D where the objects are specified by cylinders.

Structure extraction. [Sun 2007] and [Lacoste 2005] propose point processes for extracting line-networks from images by taking into account spatial interactions between lines to favor the object connection and also certain types of line junctions more likely to appear in real networks. [Stoica 2007] extend these line-network models in third dimension for recovering the cosmic filament network from point clouds representing the map of the Universe. The junction point processes developed by [Chai 2013] allow the extraction of line-networks using a graph-based representation. Junction points are not associated to geometric shapes, but are instead marked by some angles indicating the directions of the adjacent points so that a junction-point configuration is equivalent to a planar graph. [Ortner 2008] and [Chai 2012] detect buildings by displacing and connecting rectangles from aerial images. The latter use an auxiliary point process of line-segments to reinforce the rectangle extraction, whereas the former embed the point process into a MRF model to provide a structure-driven segmentation of images.

Texture analysis. [Nguyen 2010] develop a model for texture recognition in which the spatial distribution of visual keypoints discriminates the textures. [Zhu 2005] describe natural textures by a layout of textons, which can be seen as a realization of a point process specified by a texton library. [Lafarge 2010a] present a general model for extracting different types of geometric features from images, including line, rectangles and disks. A mixture of object interactions are considered such that the process can reconstruct a large variety of textures.

Object recognition. [van Lieshout 2008] develops a point process for tracking rectangular colored objects from video. A mono-dimensional point process is proposed by [Mallet 2010] for modeling 1D-signals by mixtures of parametric

functions while imposing physical constraints between the signal modes.

3.1.2 Motivations

The results obtained by these point processes are convincing and competitive with respect to other families of methods, but the performances are particularly limited in terms of computation time and convergence stability, especially on large scenes. These drawbacks explain why industry has been reluctant until now to integrate these mathematical models in their products. Indeed, the works mentioned in Section 3.1.1 emphasize complex model formulations by proposing parametrically sophisticated objects [Ge 2009, Lafarge 2010a], advanced techniques to fit objects to the data [Utasi 2011], and non-trivial spatial interactions between objects [Mallet 2010, Ortner 2008, Sun 2007]. However, these works usually rely on standard sampling procedures, mainly on the Reversible Jump Markov Chain Monte Carlo (RJMCMC) algorithm [Green 1995]. The computation time generated by such a sampler is reasonable only from data of small size. For example, the building extraction algorithm proposed by [Ortner 2008] requires around six hours from an image portion of size 1000×1000 pixels only (0.25 km^2 area). Such a solution is obviously not reasonable when dealing with entire aerial and satellite images.

In the literature, few works have addressed the optimization issues from such complex models. The proposed solutions are mainly based on some improvements of the traditional RJMCMC sampler.

Jump-Diffusion. Proposed by [Grenander 1994], this algorithm has been designed to speed-up the MCMC sampling by combining diffusion dynamics with a RJMCMC sampler. Both mechanisms play different roles: the former performs reversible jumps between the different subspaces, whereas the latter conducts stochastic diffusion within each continuous subspace, the global process being controlled by a common relaxation parameter. However this algorithm is restricted to specific energy forms [Srivastava 2002, Han 2004, Lafarge 2010a].

Data-Driven MCMC. Data considerations can also be used to drive the MCMC sampling with more efficiency [Tu 2002]. The idea consists in modeling the proposition kernels of the sampler in function of discriminative tests from data so that the ratio of relevant perturbations is strongly increased. This strategy can be dangerous if the proposition kernels are not correctly estimated from data.

Parallelization mechanisms. Some works have also proposed parallelization procedures by using multiple chains simultaneously [Harkness 2000] or decomposition schemes in configuration spaces of fixed dimension [Byrd 2010, Gonzalez 2011]. However they are limited by border effects, and are not designed to perform on large scenes. In addition, the existing decomposition schemes cannot be used for configuration spaces of variable dimension, and as a consequence, they are not adapted to sample point processes. Parallel tempering [Earl 2005] runs multiple chains in parallel at different temperatures while frequently exchanging configurations during the sampling. This technique brings robustness to the cooling schedule, but remains slow in practice as each chain explores the all configuration space.

Multiple births and deaths. A mechanism based on multiple creation and destruction of objects has also been developed to address population counting problems [Descombes 2009, Utasi 2011]. Nevertheless this algorithm is semi-deterministic and can only address problems in which object interactions are simple. In addition, object creations require the discretization of the point coordinates which induces a significant loss of accuracy.

These alternative versions of the conventional MCMC sampler globally allow the improvement of optimization performances in specific contexts. That said, the gains in terms of computation time remain weak and are usually realized at the expense of convergence stability, especially in large scenes. Finding a fast efficient sampler for general Markov point processes clearly represents a challenging problem.

3.1.3 Point Process background

3.1.3.1 Definitions and notations

A point process describes random configurations of points in a continuous bounded set K. Mathematically speaking, a point process Z is a measurable mapping from a probability space $(\Omega, \mathcal{A}, \mathbb{P})$ to the set of configurations of points in K such that

$$\forall \omega \in \Omega, p_i \in K, Z(\omega) = \{p_1, ..., p_{n(\omega)}\} \tag{3.1}$$

where $n(\omega)$ is the number of points associated with the event ω. We denote by $\mathcal{P}$, the space of configurations of points in K. Fig. 3.1 shows a realization of a point process for $K \subset \mathbb{R}^2$.

The most natural point process is the homogeneous Poisson process for which the number of points follows a discrete Poisson distribution whereas the position of the points is uniformly and independently distributed in K. Point processes can also provide more complex realizations of points by being specified by a density $h(.)$ defined in $\mathcal{P}$ and a reference measure $\mu(.)$ under the condition that the normalization constant of $h(.)$ is finite:

$$\int_{\boldsymbol{p} \in \mathcal{P}} h(\boldsymbol{p}) d\mu(\boldsymbol{p}) < \infty \tag{3.2}$$

The measure $\mu(.)$ having the density $h(.)$ is usually defined via the intensity measure $\nu(.)$ of an homogeneous Poisson process such that

$$\forall B \in \mathcal{B}(\mathcal{P}), \ \mu(B) = \int_B h(\boldsymbol{p}) \nu(d\boldsymbol{p}) \tag{3.3}$$

Specifying a density $h(.)$ allows the insertion of data consistency, and also the creation of spatial interactions between the points. Note also that $h(.)$ can be expressed by a Gibbs energy $U(.)$ such that

$$h(.) \propto \exp -U(.) \tag{3.4}$$

3.1.3.2 Markovian property

Similarly to random fields, the Markovian property can be used in a point process to create a spatial dependency of the points in a neighborhood.
A point process Z of density h is *Markovian under the neighborhood relationship* $\sim$ if and only if $\forall \mathbf{p} \in \mathcal{P}$ such that $h(\mathbf{p}) > \mathbf{0}$,

(i) $\forall \tilde{\boldsymbol{p}} \subseteq \boldsymbol{p}$, $h(\tilde{\boldsymbol{p}}) > 0$,

(ii) $\forall u \in K$, $h(\boldsymbol{p} \cup \{u\})/h(\boldsymbol{p})$ only depends on u and its neighbors $\{p \in \boldsymbol{p} \,:\, u \sim p\}$.

The expression $h(\boldsymbol{p} \cup \{u\})/h(\boldsymbol{p})$ can be interpreted as a conditional intensity. The Markovian property for random fields can thus be naturally extended in case of point processes by defining a symmetric relationship between two points of K. As shown later, the Markovian property is essential to facilitate the sampling of point processes.

3.1.3.3 From points to parametric objects

Each point p_i can be marked by additional parameters m_i such that the point becomes associated with an object $x_i = (p_i, m_i)$. This property is particularly attractive to address vision problems requiring the handle of complex parametric objects. We denote by $\mathcal{C}$, the corresponding space of object configurations where each configuration is given by $\boldsymbol{x} = \{x_1, ..., x_{n(\boldsymbol{x})}\}$. For example, a point process on $K \times M$ with $K \subset \mathbb{R}^2$ and the additional parameter space $M =]-\frac{\pi}{2}, \frac{\pi}{2}] \times [l_{min}, l_{max}]$ can be seen as random configurations of 2D line-segments since an orientation and a length are added to each point (see Fig. 3.1). Such point processes are also called marked point processes in the literature.

The most popular family of point processes corresponds to the Markov point processes of objects specified by Gibbs energies on $\mathcal{C}$ of the form

$$\forall \boldsymbol{x} \in \mathcal{C}, \;\; U(\boldsymbol{x}) = \sum_{x_i \in \boldsymbol{x}} D(x_i) \; + \sum_{x_i \sim x_j} V(x_i, x_j) \tag{3.5}$$

where $\sim$ denotes the symmetric neighborhood relationship of the Markov point process, $D(x_i)$ is a unitary data term measuring the quality of object x_i with respect

to data, and $V(x_i, x_j)$, a pairwise interaction term between two neighboring objects x_i and x_j. The $\sim$relationship is usually defined via a limit distance ε between points such that

$$x_i \sim x_j = \{(x_i, x_j) \in \boldsymbol{x}^2 : i > j, ||p_i - p_j||_2 < \varepsilon\} \tag{3.6}$$

In the sequel, we consider Markov point processes of this form. Note that this energy form has similarities with the standard multi-label energies for MRFs [Szeliski 2008]. Our problem can indeed be seen as a generalization of these MRF models. This particular case is detailed in Section 3.3.3.

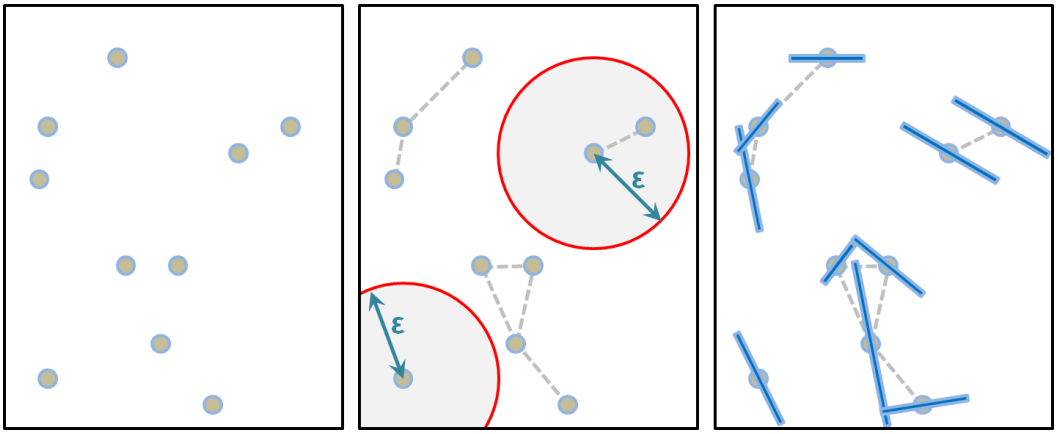


Figure 3.1: From left to right: realizations of a point process in 2D, of a Markov point process, and of a Markov point process of line-segments.The grey dashed lines represent the pairs of points interacting with respect to the neighboring relationship which is specified here by a limit distance ε between two points (Eq. 3.6).

3.1.3.4 Simulation

Point processes are usually simulated using a RJMCMC sampler [Green 1995] to search for the configuration which minimizes the energy U. This sampler consists of simulating a discrete Markov Chain $(X_t)_{t\in\mathbb{N}}$ on the configuration space $\mathcal{C}$, converging towards a target density specified by U. At each iteration, the current configuration $\boldsymbol{x}$ of the chain is locally perturbed to a configuration $\boldsymbol{y}$ according to a density function $Q(\boldsymbol{x} \to .)$, called a proposition kernel. The perturbations are local, which

means that $\boldsymbol{x}$ and $\boldsymbol{y}$ are very close, and differ by no more than one object in practice. The configuration $\boldsymbol{y}$ is then accepted as the new state of the chain with a certain probability depending on the energy variation between $\boldsymbol{x}$ and $\boldsymbol{y}$, and a relaxation parameter T_t. The kernel Q can be formulated as a mixture of sub-kernels Q_m chosen with a probability q_m such that

$$Q(\boldsymbol{x} \to .) = \sum_m q_m Q_m(\boldsymbol{x} \to .) \tag{3.7}$$

Each sub-kernel is usually dedicated to specific types of moves, as the creation/removal of an object (Birth and Death kernel) or the modification of parameters of an object (*e.g.* translation, dilatation or rotation kernels). The kernel mixture must allow any configuration in $\mathcal{C}$ to be reached from any other configuration in a finite number of perturbations (irreducibility condition of the Markov chain), and each sub-kernel has to be reversible, *i.e.* able to propose the inverse perturbation. Details on kernel computation for image and vision problems can be found in [Descombes 2011].

Algorithm 1 RJMCMC sampler [Green 1995]

1- Initialize $X_0 = \boldsymbol{x}_0$ and T_0 at $t = 0$;

2- At iteration t, with $X_t = \boldsymbol{x}$,

– Choose a sub-kernel Q_m according to probability q_m

– Perturb $\boldsymbol{x}$ to $\boldsymbol{y}$ according to $Q_m(\boldsymbol{x} \to .)$

– Compute the Green ratio

$$R = \frac{Q_m(\boldsymbol{y} \to \boldsymbol{x})}{Q_m(\boldsymbol{x} \to \boldsymbol{y})} \exp\left(\frac{U(\boldsymbol{x}) - U(\boldsymbol{y})}{T_t}\right) \tag{3.8}$$

– Choose $X_{t+1} = \boldsymbol{y}$ with probability $\min(1, R)$, and $X_{t+1} = \boldsymbol{x}$ otherwise

The RJMCMC sampler is controlled by the relaxation parameter T_t, called the temperature, depending on time t and approaching zero as t tends to infinity. Although a logarithmic decrease of T_t is necessary to ensure the convergence to the global minimum from any initial configuration, one uses a faster geometric decrease which gives an approximate solution close to the optimum [Baddeley 1993, Salamon 2002].

3.2 Approach

3.2.1 Sampling in parallel

The conventional RJMCMC sampler performs successive perturbations on objects. Such a procedure is obviously long and fastidious, especially for large scale problems. A natural idea but still unexplored for Markov point processes consists in sampling objects in parallel by exploiting their conditional independence outside the spatial neighborhood. Such a strategy implies partitioning the space K so that simultaneous perturbations are performed at locations far enough apart to not interfere and break the convergence properties.

3.2.1.1 From sequential to parallel sampling

Let $(X_t)_{t\in\mathbb{N}}$, be a Markov chain simulating a Markov point process with a MCMC dynamics, and $\{c_s\}$ be a partition of the space K, where each component c_s is called a cell. Two cells c_1 and c_2 are said to be *independent on* X if the transition probability for any random perturbation falling in c_1 at any time t does not depend on the objects and perturbations falling in c_2, and vice versa.

One can demonstrate that the transition probability of two successive perturbations falling in independent cells under the temperature T_t is equal to the product of the transition probabilities of each perturbation under the same temperature. In other words, realizing two successive perturbations on independent cells at the same temperature is equivalent to performing them in parallel. To do so, let us consider two cells c_1 and c_2 independent on $(X_t)_{t\in\mathbb{N}}$. We denote by $\boldsymbol{x}$, a realization of the point process such that $\boldsymbol{x} = (x_1, x_2, u)$ where x_1 (respectively x_2) represents the set of points falling in the cell c_1 (respectively c_2), and u is the remaining set of points falling in $K - \{c_1, c_2\}$. Let $\boldsymbol{y}$ be a new configuration of points obtained from $\boldsymbol{x}$ by two perturbations on the cells c_1 and c_2 so that $\boldsymbol{y} = (y_1, y_2, u)$, as illustrated on Fig. 3.2.

The transition probability $\Pr[X_{t+2} = \boldsymbol{y} | X_t = \boldsymbol{x}]$ of moving from the state $\boldsymbol{x}$ at time

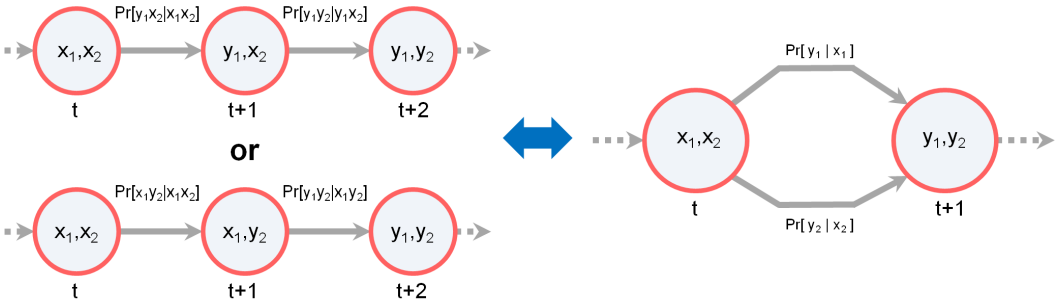


Figure 3.2: Illustration of the equivalence between two successive perturbations on independent cells c_1 and c_2, and two simultaneous perturbations on each cell.

t to the state $\boldsymbol{y}$ at time $t+2$ can be expressed as

$$\begin{aligned}
&\Pr[X_{t+2} = \boldsymbol{y} | X_t = \boldsymbol{x}] \\
&\quad = \Pr[X_{t+2} = (y_1, y_2, u) | X_{t+1} = (y_1, x_2, u)] \\
&\qquad \times \Pr[X_{t+1} = (y_1, x_2, u) | X_t = \boldsymbol{x}] \\
&\qquad + \Pr[X_{t+2} = (y_1, y_2, u) | X_{t+1} = (x_1, y_2, u)] \\
&\qquad \times \Pr[X_{t+1} = (x_1, y_2, u) | X_t = \boldsymbol{x}]
\end{aligned} \tag{3.9}$$

or, the temperature parameter is constant between t and $t+2$, which means that

$$\begin{aligned}
&\Pr[X_{t+2} = \boldsymbol{y} | X_{t+1} = (y_1, x_2, u)] \\
&\quad = Q((y_1, x_2, u) \to \boldsymbol{y}) \\
&\qquad \times \min\left[1, \tfrac{Q(\boldsymbol{y} \to (y_1, x_2, u))}{Q((y_1, x_2, u) \to \boldsymbol{y})} \exp\left(\tfrac{U((y_1, x_2, u)) - U(\boldsymbol{y})}{T_{t+1}}\right)\right] \\
&\quad = Q((y_1, x_2, u) \to \boldsymbol{y}) \\
&\qquad \times \min\left[1, \tfrac{Q(\boldsymbol{y} \to (y_1, x_2, u))}{Q((y_1, x_2, u) \to \boldsymbol{y})} \exp\left(\tfrac{U((y_1, x_2, u)) - U(\boldsymbol{y})}{T_t}\right)\right] \\
&\quad = \Pr[X_{t+1} = \boldsymbol{y} | X_t = (y_1, x_2, u)]
\end{aligned} \tag{3.10}$$

The cells c_1 and c_2 being independent, the transition probability for the perturbation y_2 falling in c_2 does not depend, by definition, on x_1 and y_1. Thus we have in particular

$$\begin{aligned}
&\Pr[X_{t+1} = (y_1, y_2, u) | X_t = (y_1, x_2, u)] \\
&\quad = \Pr[X_{t+1} = (x_1, y_2, u) | X_t = (x_1, x_2, u)]
\end{aligned} \tag{3.11}$$

This leads to the equation

$$\begin{aligned}
&\Pr[X_{t+2} = \boldsymbol{y} | X_{t+1} = (y_1, x_2, u)] \\
&\quad = \Pr[X_{t+1} = (x_1, y_2, u) | X_t = \boldsymbol{x}]
\end{aligned} \tag{3.12}$$

Similarly, one can demonstrate that

$$\begin{aligned} &\Pr[X_{t+2} = \boldsymbol{y} | X_{t+1} = (x_1, y_2, u)] \\ &\quad = \Pr[X_{t+1} = (y_1, x_2, u) | X_t = \boldsymbol{x}] \end{aligned} \tag{3.13}$$

Finally, by inserting Eq. 3.12 and 3.13 in Eq. 3.9, the expected result is obtained

$$\begin{aligned} &\Pr[X_{t+2} = \boldsymbol{y} | X_t = \boldsymbol{x}] \\ &\quad = 2! \Pr[X_{t+1} = (y_1, x_2, u) | X_t = \boldsymbol{x}] \\ &\qquad \times \Pr[X_{t+1} = (x_1, y_2, u) | X_t = \boldsymbol{x}] \end{aligned} \tag{3.14}$$

where 2! is the combinatorial coefficient corresponding to the number of permutations of perturbations in the sequential chain. This proof can be easily extended by recurrence in case of n cells mutually independent on $(X_t)_{t \in \mathbb{N}}$, with $n > 2$.

3.2.1.2 How to guarantee cell independence?

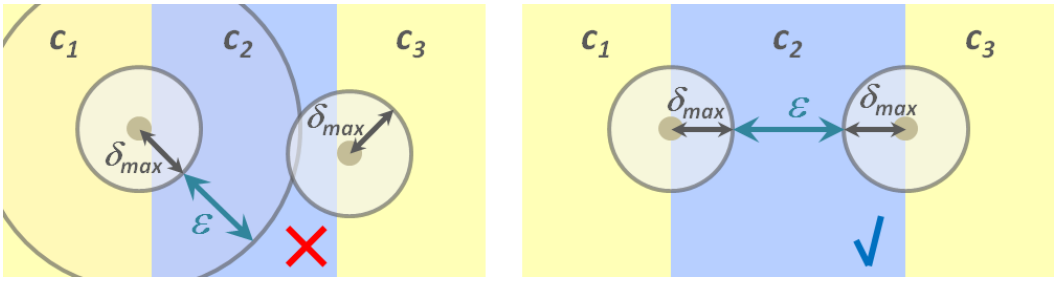


Figure 3.3: Independence of cells. On the left case, the width of the cell c_2 is not large enough to ensure the independence of the cells c_1 and c_3: the two grey points in c_1 and c_3 cannot be perturbed at the same time. On the right case, the cells c_1 and c_3 are independent as Eq. 3.15 is satisfied.

By definition, two cells are independent if the transition probability for any random perturbation falling in the first cell does not depend on the objects and perturbations falling in the second cell, and vice versa. It implies that the two cells must be located at a minimum distance from each other. As illustrated in Fig. 3.3, this distance must take into account the width ε of the neighboring relationship induced by the Markovian property so that every possible object falling in the first cell cannot be a neighbor of the objects falling in the second cell. As an object can

be displaced to another cell during a perturbation, the minimum distance must also consider the length of the biggest move allowed as object perturbation, denoted by $\delta_{\max}$. Considering these two constraints, the independence between two cells c_1 and c_2 can then be guaranteed when

$$\min_{p_1 \in c_1,\, p_2 \in c_2} ||p_1 - p_2||_2 \geq \varepsilon + 2\delta_{\max} \tag{3.15}$$

3.2.1.3 How to construct a cell partition?

Knowing a condition for insuring the cell independence (Eq. 3.15), the objective is now to find a partitioning of the space K optimizing the performance of the sampling in parallel.

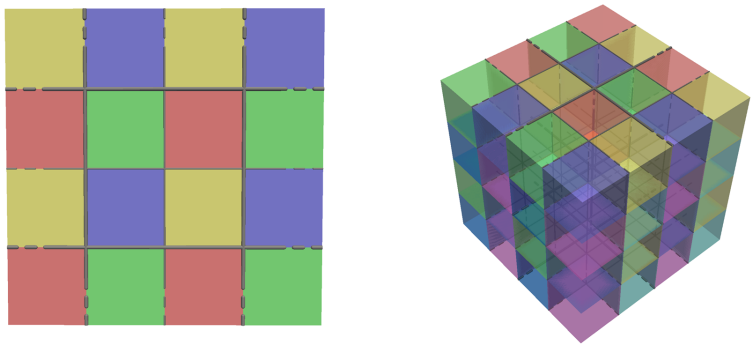


Figure 3.4: Regular partitioning scheme of K. In dimension two (left), the cells are squares of identical size regrouped into 4 mic-sets (yellow, blue, red and green). Each cell is adjacent to cells belonging to different mic-sets. In dimension three (right), the cells are cubes regrouped into 8 mic-sets.

The natural idea consists of partitioning K into a regular mosaic of cells with size greater than or equal to the minimum distance between independent cells, *i.e.* to $\varepsilon + 2\delta_{\max}$. The cells can then be regrouped into $2^{\dim K}$ sets such that each cell is adjacent to cells belonging to different sets. Fig. 3.4 illustrates the partitioning scheme for $\dim K = 2$ and $\dim K = 3$. This partitioning scheme guarantees the mutual independence between all the cells of a same set. In the sequel, such a set is called a *mic-set* (set of Mutually Independent Cells). Each cell of a mic-set can thus be perturbed simultaneously using a MCMC dynamics.

However, the number of cells which can be perturbed simultaneously is limited by the computer architecture, in our case by the number of threats available in GPU. When sampling in large scenes, this number is usually much lower than the number of cells in a mic-set if a cell width of $\varepsilon + 2\delta_{\max}$ is imposed. One solution could consist in fixing the cell width so that the number of cell in a mic-set cannot exceed the maximum number of simultaneous perturbations. This option does not take advantage of the Markovian property, and leads to average performances in practice. In order to fully exploit both the potential of the computer architecture and the Markovian property, a non-regular partitioning of K is required. This problem is addressed in the next section where spatial information from input data is exploited to ideally partition K.

3.2.2 Data-driven mechanism

The creation of a non-regular partitioning of K leads us to consider a proposition kernel which distribute the points non-uniformly in the space K. In other words, perturbations do not have the same occurrence according to their locations in K. To design such a proposition kernel, one needs to take into account the characteristics of observed scenes. Contrary to the data-driven solution proposed by [Tu 2002], our mechanism must be compatible with the parallelization constraints mentioned in Section 3.2.1.3.

3.2.2.1 Principle

Our idea consists in creating a proposition kernel as an accumulation of uniform sub-kernels spatially restricted on the domain of the cells. Indeed, such a mixture of sub-kernels has the interesting property of still guaranteeing the parallelization of the sampling. In the literature, mixtures of sub-kernels are frequently used in MCMC dynamics to simulate point processes, but with a restricted role where each sub-kernel is dedicated to a perturbation type (*e.g.* birth and death, translation, rotation, etc).

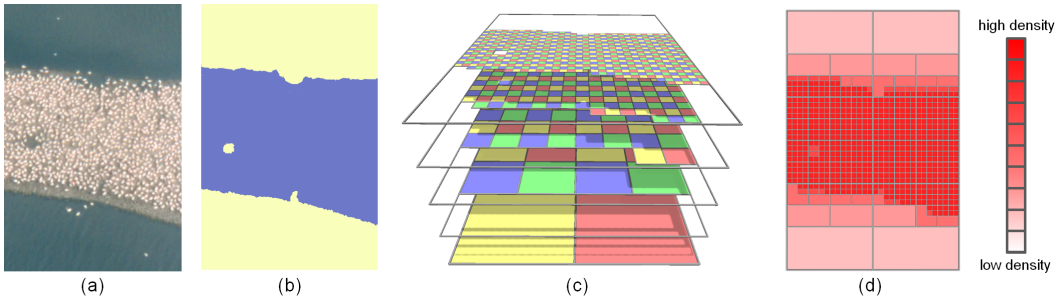


Figure 3.5: Space-partitioning tree in dimension two. (b) A class of interest (blue area) is estimated from (a) an input image. (c) A quadtree is created so that the levels are recursively partitioned according to the class of interest. Each level is composed of four mic-sets (yellow, blue, red and green sets of cells) to guarantee the sampling parallelization. (d) The accumulation of the probabilities $q_{c,t}$ over the different levels of the quadtree generates a density map allowing the points to be non-uniformly distributed in the scene. Note how the density map focuses on the class of interest while progressively decreasing its intensity when moving away. Image courtesy of *Tour du Valat* [du Valat 2013].

The sub-kernel accumulation mechanism is driven by a space-partitioning tree $\mathcal{K}$ which is defined as a set of L sub-partitions of K, denoted by $\{c_s\}^{(1)}, .., \{c_s\}^{(L)}$, and organized so that, for $i = 2..L$, $\{c_s\}^{(i)}$ is a subdivided partition of $\{c_s\}^{(i-1)}$. Each level of the space-partitioning tree corresponds to a set of cells having an identical size. A 1-to-$2^{\dim K}$ hierarchical subdivision scheme is considered to build the space-partitioning tree, typically a quadtree in dimension two and an octree in dimension three.

Given a space-partitioning tree, a density map specifying how the points must be spatially distributed in the space K can then be constructed. The creation of this density map relies on the accumulation of the uniform sub-kernels spatially restricted to the subspace supporting every cell of the space-partitioning tree $\mathcal{K}$, as defined in Eq. 3.7 and illustrated in Fig. 3.5.

3.2.2.2 Data-driven space-partitioning tree

In order to create a relevant space-partitioning tree, the data is used to guide the cell subdivision. We assume that a class of interest in K, in which the objects have a high probability to belong to, can be roughly distinguished from the data. The extraction of such a class is not addressed in this chapter, and is supposed to be done by a segmentation algorithm of the literature adapted to the considered application. A cell at a given level of the tree is divided into $2^{\dim K}$ cells at the next level if it overlaps with the given class of interest. The hierarchical decomposition is stopped before that the size of the cell becomes inferior to $\varepsilon + 2\delta_{\max}$, *i.e.* before that the cell independence condition (Eq. 3.15) is not longer valid.

The space-partitioning tree allows the creation of a proposition kernel in an elegant way as points are naturally and efficiently distributed in K. On the area of interest, the intensity of the density map is maximal. When moving far from the class of interest, the intensity progressively decreases as shown in Fig. 3.5, while being ensured to be non-null. In addition, the sampling is not severely affected when the class of interest is inaccurately extracted.

3.2.2.3 Proposition kernel formulation

Given a space-partitioning tree $\mathcal{K}$ composed of L levels, and $2^{\dim K}$ mic-sets for each level, a general proposition kernel Q can then be formulated as a mixture of uniform sub-kernels $Q_{c,t}$, each sub-kernel being defined on the cell c of $\mathcal{K}$ by the perturbation type $t \in \mathcal{T}$, such that

$$\forall \boldsymbol{x} \in \mathcal{C},\ Q(\boldsymbol{x} \to .) = \sum_{c \in \mathcal{K}} \sum_{t \in \mathcal{T}} q_{c,t}\, Q_{c,t}(\boldsymbol{x} \to .) \tag{3.16}$$

where $q_{c,t} > 0$ is the probability of choosing the sub-kernel $Q_{c,t}(\boldsymbol{x} \to .)$. The probability $q_{c,t}$ allows us to specify the intensity of the density map, given the space-partitioning tree. In practice, this probability is chosen as

$$q_{c,t} = \frac{\Pr(t)}{\#\text{cells in } \mathcal{K}} \tag{3.17}$$

where $\Pr(t)$ denotes the probability of choosing the perturbation type $t \in \mathcal{T}$. The expression of $q_{c,t}$ (Eq. 3.17) allows the finest levels in the space-partitioning tree to be favored so that the perturbations mainly focus on the domain supporting the class of interest and its surrounding. As discussed later in Section 3.3.1.1, other possible expressions may become more interesting when the class of interest cannot be reliably extracted.

Four types of perturbations are usually considered in practice so that $\mathcal{T} = \{$birth and death, translation, rotation, scale$\}$. When objects have several possible models, the perturbations consisting in switching the model of an object can also be used. Note that the proposition kernel Q is reversible as a sum of reversible sub-kernels. Note also that such a proposition kernel allows us to visit the whole configuration space $\mathcal{C}$, as guaranteed by the sub-kernels of the coarsest level of $\mathcal{K}$.

3.2.3 New sampling procedure

The kernel defined in Eq. 3.16 is embedded into a MCMC dynamics so that the proposed sampler allows a high number of simultaneous perturbations generated by a data-driven proposition kernel.

3.2.3.1 Algorithm

The proposed sampler, detailed in Algorithm 2, can be seen as a parallelized extension of the traditional RJMCMC with data-driven proposition kernel. As illustrated on Fig. 3.6, each perturbation is completely independent of the other perturbations simultaneously proposed in the cells of the considered mic-set.

Note that the temperature parameter is updated after each series of simultaneous perturbations such that the temperature decrease is equivalent to a cooling schedule by plateau in a standard sequential MCMC sampling. Note also that the space-partitioning tree protects the sample from mosaic effects. In practice, the sampling is stopped when no perturbation has been accepted during a certain number of iterations.

Algorithm 2 Our data-driven parallel sampler

1-Initialize $X_0 = \boldsymbol{x}_0$ and T_0 at $t = 0$;

2-Compute a space-partitioning tree $\mathcal{K}$;

3-At iteration t, with $X_t = \boldsymbol{x}$,

– Choose a mic-set $S_{mic} \in \mathcal{K}$ and a kernel type $t \in \mathcal{T}$ according to probability $\sum_{c \in S_{mic}} q_{c,t}$

– For each cell $c \in S_{mic}$,

  – Perturb $\boldsymbol{x}$ in the cell c to a configuration $\boldsymbol{y}$ according to $Q_{c,t}(\boldsymbol{x} \to .)$

  – Calculate the Green ratio

$$R = \frac{Q_{c,t}(\boldsymbol{y} \to \boldsymbol{x})}{Q_{c,t}(\boldsymbol{x} \to \boldsymbol{y})} \exp\left(\frac{U(\boldsymbol{x}) - U(\boldsymbol{y})}{T_t}\right) \tag{3.18}$$

  – Choose $X_{t+1} = \boldsymbol{y}$ with probability $\min(1, R)$, and $X_{t+1} = \boldsymbol{x}$ otherwise

– Update $T_{t+1} = \alpha T_t$

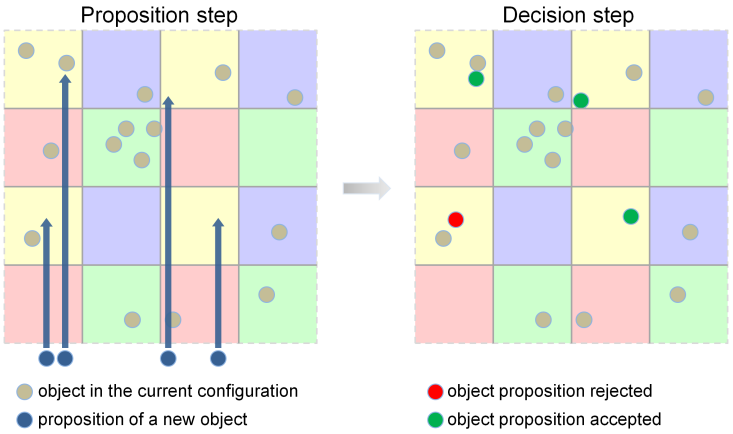


Figure 3.6: Mechanism of the sampler. At a given iteration, a mic-set is chosen in the space-partitioning tree (here, the set of yellow cells), as well as a kernel type (here, births illustrated by the insertion of blue dots). (left) During the proposition step, a perturbation will be proposed simultaneously in each cell of the selected mic-set, independently from each other. (right) Each perturbation will then be either accepted (green dot) or rejected (red dot) during the decision step. The decisions are independent, relying on the Green ratio computations and numbers randomly chosen in the interval $[0, 1]$, as formulated in Eq. 3.18.

3.2.3.2 Implementation

The algorithm has been implemented on GPU using CUDA. A thread is dedicated to each simultaneous perturbation so that operations are performed in parallel for each cell of a mic-set. The sampler is thus all the more efficient as the mic-set contains many cells, and generally speaking, as the scene supported by K is large. Moreover, the code has been programmed to avoid time-consuming operations. In particular, the threads do not communicate between each other, and memory coalescing permits fast memory access. The memory transfer between CPU and GPU has also been minimized by indexing the parametric objects. The experiments presented in this section have been performed on a 2.5 Ghz Xeon computer with a Nvidia graphics card (Quadro 4800, architectures 1.3).

3.3 Experiments

3.3.1 Experiments with images

3.3.1.1 Population counting

The algorithm has been evaluated on population counting problems from large-scale images using a point process in 2D, *i.e.* with $\dim K = 2$. The problem presented in Fig. 3.7 consists in detecting migrating birds to extract information on their number, their size and their spatial organization. Such problem has been addressed previously by [Descombes 2009]. The point process is marked by ellipses which are simple geometric objects defined by a point (center of mass of an ellipse) and three additional parameters. This object shape is well adapted to capture the bird contours. The energy is specified by a unitary data term based on the Bhattacharyya distance between the radiometry inside and outside the object, and a pairwise interaction penalizing the strong overlapping of objects. Details on the energy are given in Appendix A. The probability $\Pr(t)$ of choosing the perturbation type t is set to 0.2 if $t =$'birth and death', 0.4 if $t =$'translation', 0.2 if $t =$'rotation', and 0.2 if $t =$'scale' in our experiments.

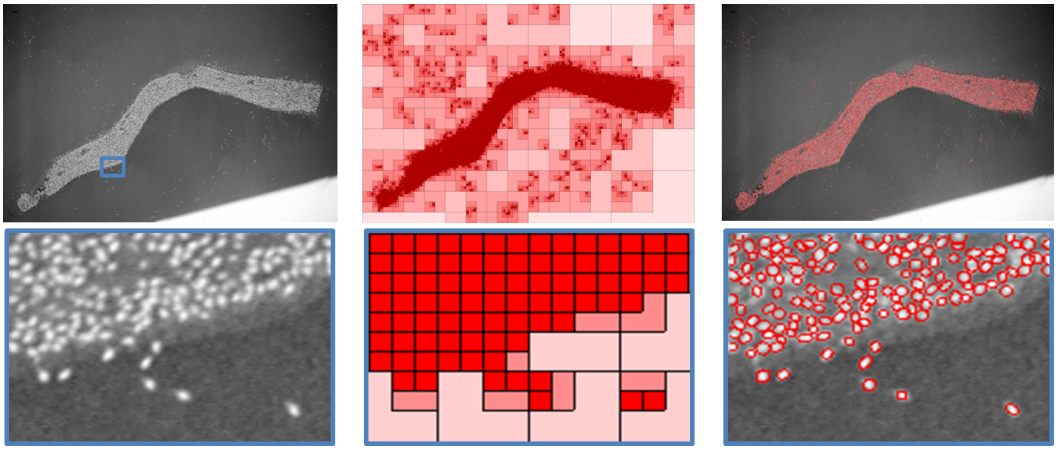


Figure 3.7: Bird counting by a point process of ellipses. (right) More than ten thousand birds are extracted by our algorithm in a few minutes from (left) a large scale aerial image. (middle) A quadtree partitioning the scene is used to create a density map so that the objects are more frequently proposed in the locations of interest. Note, on the cropped parts, how the birds are accurately captured by ellipses in spite of the low quality of the image and the partial overlapping of birds. Images courtesy of *Tour du Valat* [du Valat 2013]

Table 3.1: Stability of the various samplers. The coefficients of variation of the energy, time and number of objects reached at the convergence are computed over 50 simulations.

	Coefficient of variation		
	energy	time	#objects
RJMCMC [Green 1995]	7.3%	4.2%	1.7%
Parallel tempering [Earl 2005]	5.0%	20%	4.4%
DDMCMC [Tu 2002]	8.1%	0.7%	1.8%
multiple birth and death [Descombes 2009]	5.0%	2.1%	1.3%
our sampler with regular partitioning	7.4%	6.2%	1.6%
our sampler with space-partitioning tree	4.4%	1.8%	1.1%

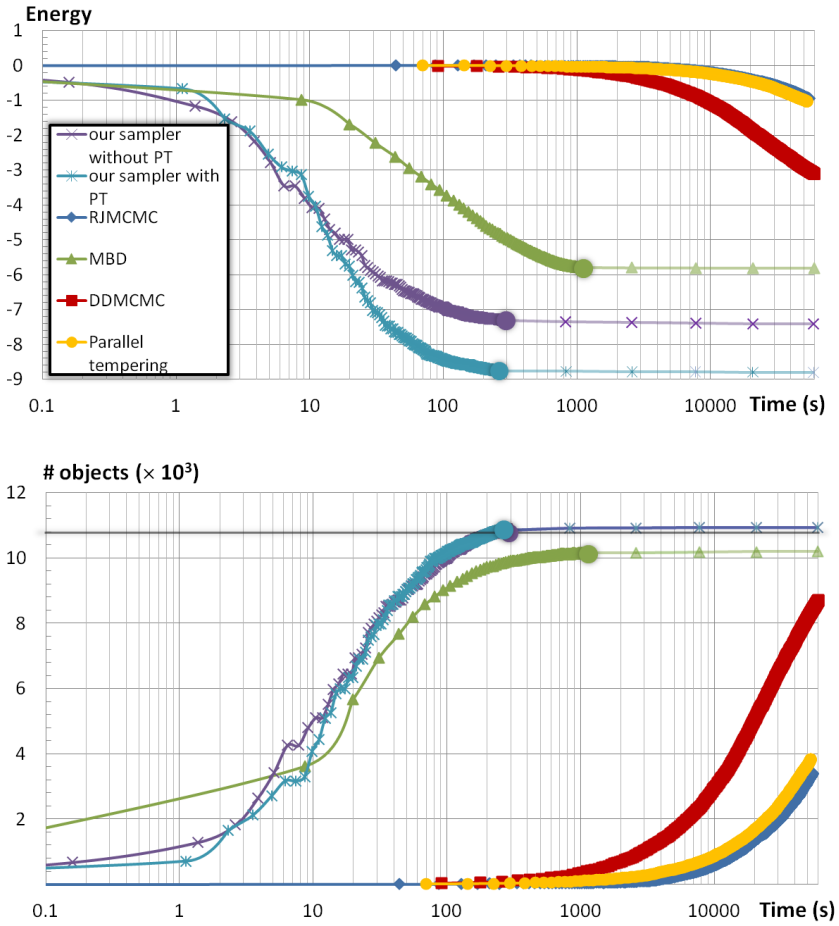


Figure 3.8: Performances of the various samplers. The top graph describes the energy decrease over time from the bird image presented in Fig. 3.7 (the colored dots correspond to algorithm convergence). Note that time is represented using a logarithmic scale, and that the slow convergence of RJMCMC [Green 1995], DDMCMC [Tu 2002], and parallel tempering [Earl 2005] algorithms is not displayed on the graph. The bottom graph shows the evolution of the number of objects during the sampling. Contrary to the other samplers, the number of objects found by our sampler with and without space-partitioning tree (PT) is very close to the ground truth (black line). Note that estimating the correct number of objects does not mean that the objects are correctly fitted to the data, but it is an important criterion for population counting problems as underlined by [Lempitsky 2010].

Computation time, quality of the reached energy, and stability are the three important criteria used to evaluate and compare the performance of samplers (Table 3.1). As shown on Fig. 3.8, our algorithm obtains the best results for each of the criteria compared to the existing samplers. In particular, we reach a better energy (-8.76 *vs* -5.78 for [Descombes 2009], and -2.01 for [Green 1995]) while significantly reducing computation times (269 sec *vs* 1078 sec for [Descombes 2009], and $> 10^5$ sec for [Green 1995], [Tu 2002], and [Earl 2005]). Fig. 3.8 also underlines an important limitation of the point process sampler of reference for population counting [Descombes 2009] compared to our algorithm. Indeed, the discretization of the object parameters required in [Descombes 2009] causes approximate detection and localization of objects which explains the average quality of the reached energy.

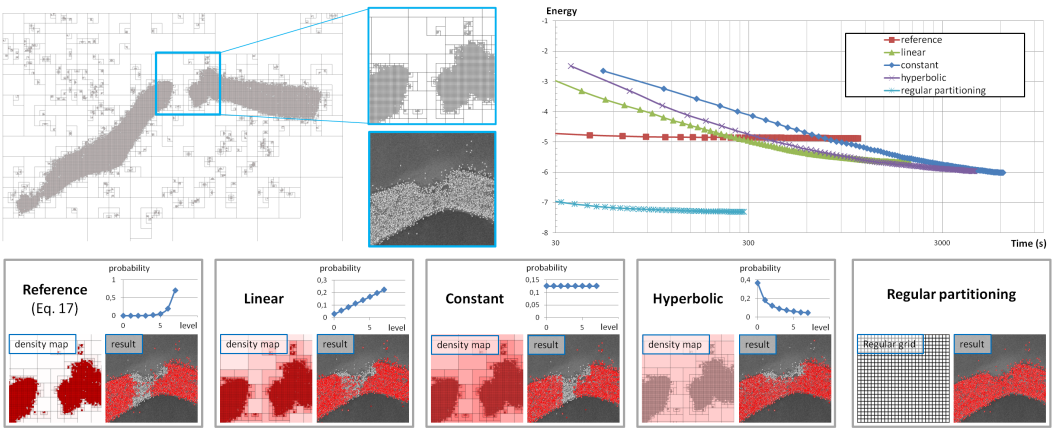


Figure 3.9: Behavior of the sampler with a non relevant space-partitioning tree. (top left) When the class of interest cannot be correctly extracted, the space-partitioning tree is of low quality, as shown on the closeup in which a large area of interest is missed. (bottom left frame) The probability $q_{c,t}$ defined in Eq. 3.17 does not perform well. (bottom center frames) More interesting choices for $q_{c,t}$ can then be used, as the linear, constant and hyperbolic formulations which progressively favors the selection of the coarsest levels in the space-partitioning tree. The graph represents the probability of selecting the space-partitioning tree levels, the value 0 being the coarsest level of the tree. (bottom right frame) Note that the use of a regular partitioning can become especially efficient in such a situation. (top right) The energy graphs summarizing the performances of these different formulations.

The stability is analyzed by the coefficient of variation, defined as the standard deviation over mean, and known to be a relevant statistical measure for comparing methods having different means. Our sampler provides a better stability than the existing algorithms, including the multiple birth and death sampler which is supposed to be particularly stable thanks to its semi-deterministic mechanism. Note that the DDMCMC algorithm is particularly stable in terms of time, but is more likely to be stuck in local minimums as energy variation is high. The impact of the data-driven mechanism is also measured by performing tests with a proposition kernel based on a regular partitioning of the space K (see Fig. 3.4).

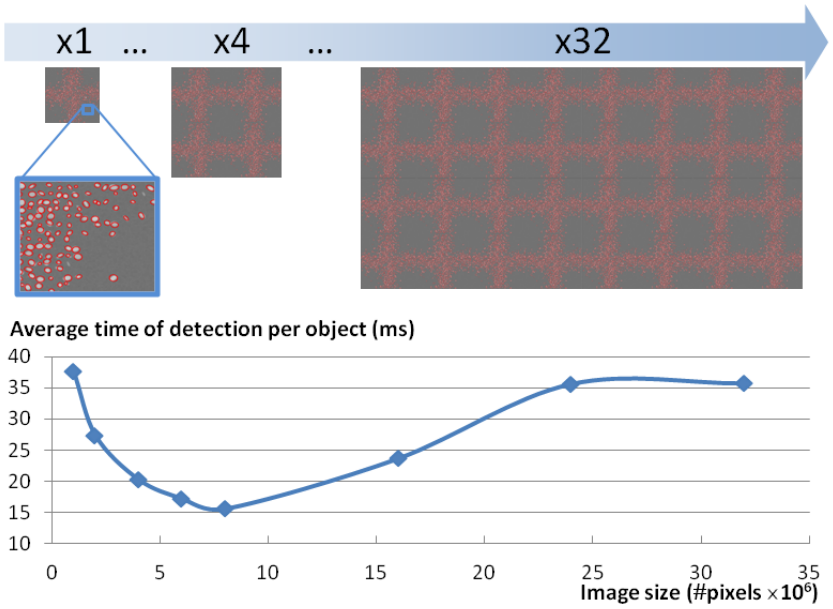


Figure 3.10: Impact of the data size on the computation times. (top) The performances of our sampler have been tested from a set of simulated images of ellipses corrupted by blur and noise, whose size progressively increases. (bottom) The results are displayed on the graph representing the evolution of the average time of detection per object in function of the image size. The minimum average time is met (here, 15 ms) when the optimal occupancy conditions are reached. In this experiment, the image size is directly proportional to the number of used threads as the cell size at the finest level of the trees is identical for all simulated images.

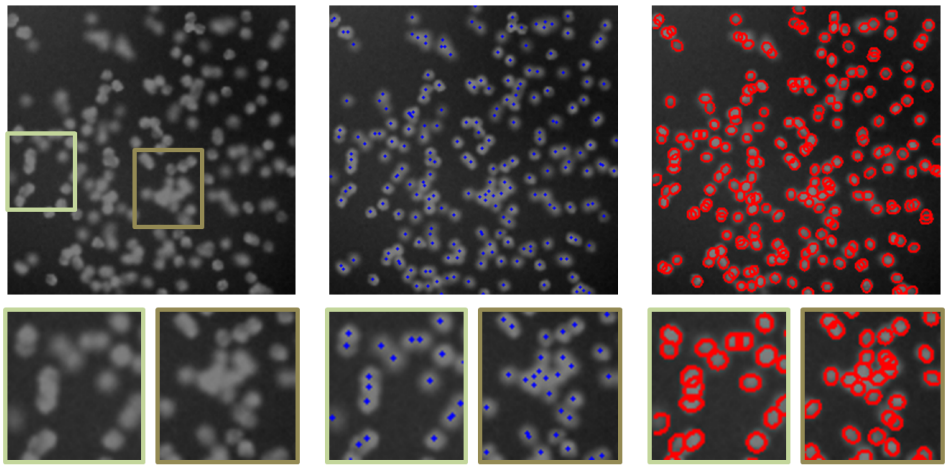


Figure 3.11: Cell counting from (left) the microscope image *cell17*. (middle) Ground Truth shows the location of each cell through a blue cross. (right) Our 2D point process of ellipses captures the cells with very few errors. As illustrated on the right crop, omissions can appear when many cells are regrouped in a tiny area. Note that, in such a case, it is very difficult to visually detect the cells, even for an expert. Images simulated by [Lehmussola 2007].

The performances decrease but remain better than the existing algorithms. In particular, the sampler loses stability, and the objects are detected and located less accurately than by using a relevant space-partitioning tree. However the use of a regular partition of K is an interesting solution when the class of interest is not correctly extracted from the data, more precisely when entire parts of the class of interest are omitted. This leads to generate space-partitioning trees of low quality.

Fig. 3.9 shows the behavior of the sampler in such a situation. In particular, one can see that the formulation of the probability $q_{c,t}$ proposed initially in Eq. 3.17 is not relevant anymore, and needs to be modified in order to favor the selection of the coarsest levels in the space-partitioning tree. A constant or hyperbolic formulations for $q_{c,t}$ then become interesting solutions. Because of the construction of the density map, note that the sampler is not affected when the contours of the class of interest are rough.

Table 3.2: Comparisons with the cell counting approach proposed by [Lempitsky 2010] from the microscope images simulated by [Lehmussola 2007]. The given values correspond to the number of cells detected in a set of images. Our algorithm provides a better estimation of the number of cells than both the $L1$- and Tikhonov-regularization versions of [Lempitsky 2010]. In particular, our algorithm is more accurate where cells are highly concentrated.

	our method	Lempitsky ($L1$-reg.)	Lempitsky (Tikhonov-reg.)	Ground Truth
cell17	**209**	202.9	194.1	213
cell18	184	**184.6**	175.9	185
cell19	**187**	192.2	180.1	188
cell20	**169**	174.1	170.4	169
cell21	147	**148.6**	144.4	149
cell22	**184**	182.6	176.5	184
cell23	**159**	158.3	157.6	161
RMSE	**1.93**	4.71	9.21	-

This gap with the others algorithms in terms of performances becomes more marked when the input scene is larger. Contrary to the existing samplers, the average time of detection per object of our sampler does not explode when increasing the data size, but it falls until reaching the optimal occupancy conditions of the GPU architecture (see Fig. 3.10). During this stage, there is indeed no extra cost in terms of computation time for using additional threads, *i.e.* for increasing the number of simultaneous perturbations. The average time of detection per object then slightly increases before becoming stable.

Tab. 3.2 and Fig. 3.11 and 3.12 show results on cell counting from microscope images. These images have been simulated by [Lehmussola 2007] and are provided with ground truth, i.e. the exact number and location of cells are known for each image. Our algorithm has been compared to the supervised approach proposed

by [Lempitsky 2010] in Tab. 3.2. Note that, contrary to our algorithm, their approach just delivers an estimated number of cells per images without locating and delineating them. As shown in Tab. 3.2, the root mean square error (RMSE) in terms of number of cells from our sampler is more than twice lower than from [Lempitsky 2010].

Note finally that the proposed model is general and can be used for many different population counting problems, as illustrated on Fig. 3.13. This model does not just count a population but it also provides helpful information on its spatial organization in order to track and analyze its behavior. Note also that ellipses can be substituted by any type of parametric 2D-objects in the model formulation.

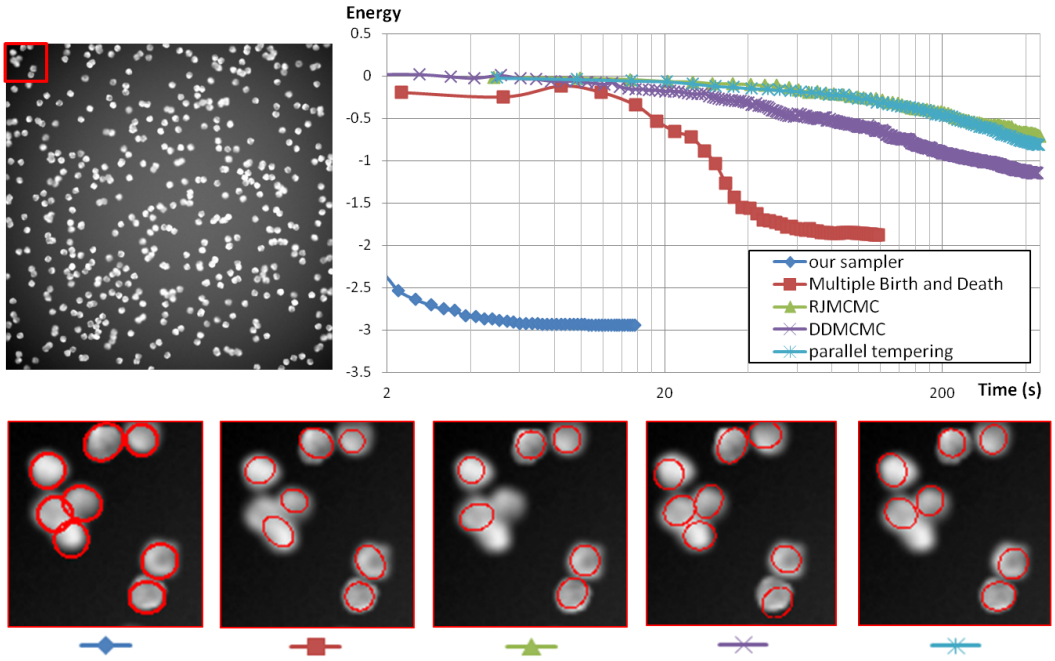


Figure 3.12: Performances of various samplers on cell counting. The top right graph presents the performances of the existing algorithms (time and energy) from a microscope image, the bottom close-ups show the quality of the reached configurations. Our sampler allows both low running time and good configuration quality. Images simulated by [Lehmussola 2007].

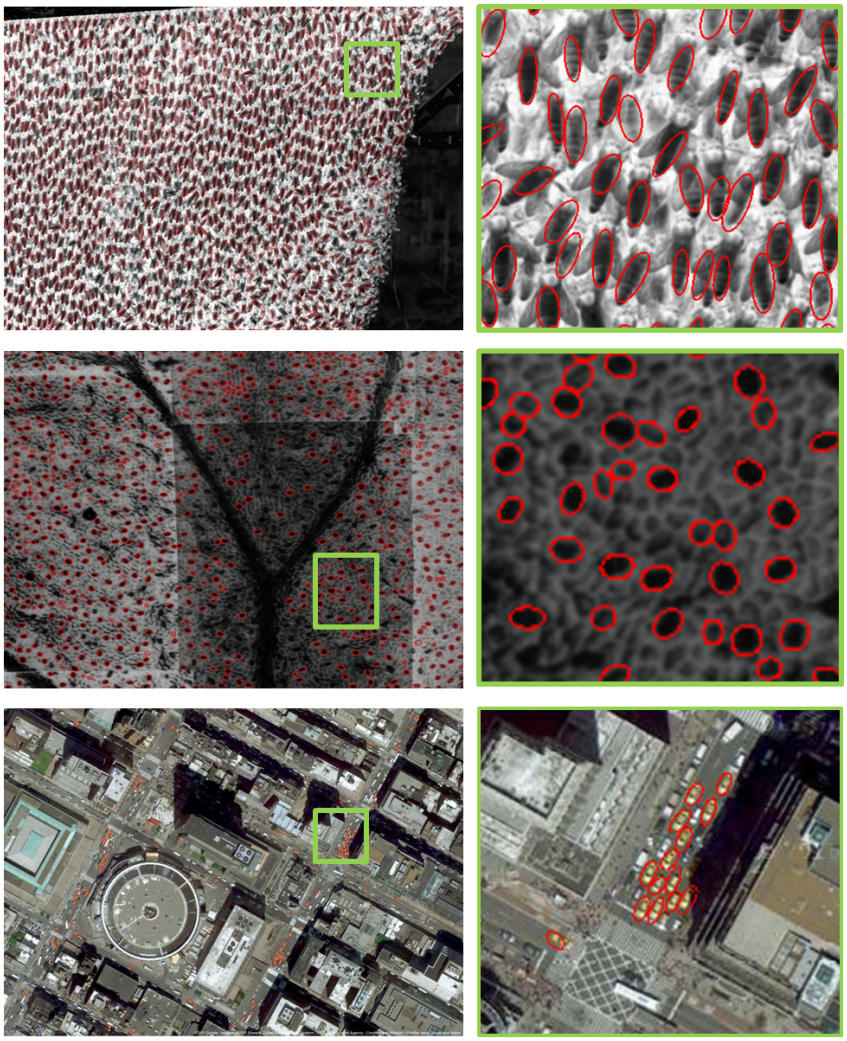


Figure 3.13: Various population counting problems. Our algorithm captures different objects of interest by ellipses in large scenes, as (top) bees from beehive pictures, (middle) opened stomata from microscope images of leaf, and (bottom) yellow cabs from aerial images. 1167 bees (respectively 757 stomata and 87 taxis) are detected in 12 minutes (respectively 168 seconds and 165 seconds). Note that the computation time is higher for bee detection because the partitioning scheme contains few cells, *i.e.* 75. As shown on the close-ups, the objects of interest are globally well detected in spite of the high concentration and overlap of objects. Images courtesy of *Roger W. Ehrich* and *Google* [Google 2013]

3.3.1.2 Structure extraction

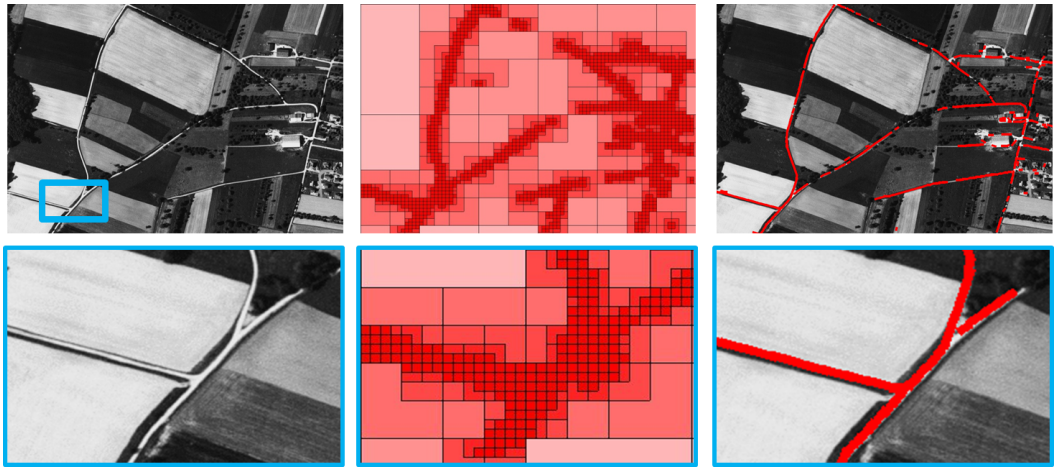
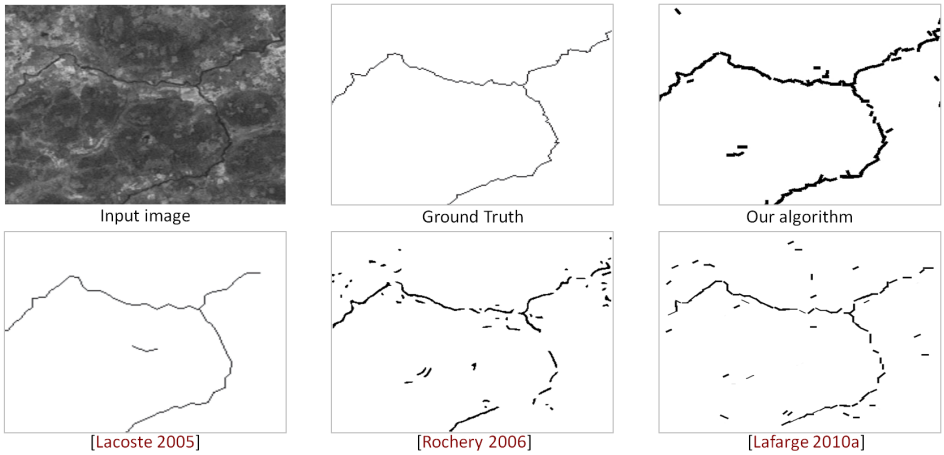


Figure 3.14: Line-network extraction by a point process of line-segments. (middle) Even with a rough density map, (right) the road network is recovered (red segments) by our algorithm in 16 seconds from (left) a satellite image. Similarly to most existing methods, parts of the network can be omitted when roads are hidden by trees at some locations, as shown on the close-up. Images courtesy of *IGN*.

The algorithm has also been tested for recovering specific structures from images, in this case line-networks. The parametric objects are specified by line-segments defined as a point (center of mass of a line-segment) and two additional parameters (length and orientation). Contrary to the population counting model previously used, the pairwise potential includes a connection interaction to link the line-segments. This constraint allows the object configurations to be structured as line-networks, each connection between line-segments representing a junction in the line-network. Details on this energy formulation can be found in Appendix A. The probability $\Pr(t)$ of choosing the perturbation type t is set to 0.4 if $t =$'birth and death', 0.1 if $t =$'translation', 0.4 if $t =$'rotation', and 0.1 if $t =$'scale' in our experiments.

Figure 3.14 shows a road network extraction result obtained from a satellite image. Our method significantly improves the computation times with respect to existing methods, as detailed on [Benchmark 2013]. 16 seconds are required in our



	Time (s)	FPR	FNR	model
our sampler	16.8	0.02	0.25	line
[Lafarge 2010a]	108	0.02	0.45	line
[Lacoste 2005]	2700	0.01	0.35	line
[Rochery 2006]	600	0.01	0.50	pixel

Figure 3.15: Extraction of a river network from a satellite image by different methods. Our result is visually competitive with respect to existing methods. The structure of the river is correctly recovered by connected line-segments. The accuracy of our algorithm in terms of False Positive / False Negative rates is competitive, better than [Lafarge 2010a] and [Rochery 2006], and similar to [Lacoste 2005]. Note that our algorithm significantly improves the computation times. Images courtesy of *BRDM*.

case, compared to 7 minutes by a Jump-Diffusion algorithm [Lafarge 2010a], 155 minutes for a RJMCMC-based method [Lacoste 2005], and 60 minutes for an Active Contour approach [Rochery 2006]. One can see in Fig. 3.15 that the visual quality of our extracted networks is relatively inferior to those of [Lacoste 2005], even if the quantitative results are similar. That said, their method relies on a heavy formalism, *i.e.* the *Quality Candy* model, requiring many parameters (*i.e.* around fifteen) whose tuning by trial and error is a difficult task as several parameters have unstable behaviors.

3.3.2 Experiments with Lidar data

Our algorithm has been tested with $\dim K = 3$ on an original model for extracting predefined parametric 3D-templates from unstructured point clouds containing a lot of outliers and noise. This model has been used to detect trees from Laser scans of urban environments composed of many other different objects such as buildings, ground, cars, fences, wires, *etc.* This model also allows the recognition of the shapes and types of trees. The objects associated with the point process correspond to a library of different 3D-templates of trees detailed in Appendix A (Fig. A.1 and A.2). The unitary data term of the energy measures the distance from points to the surface of the 3D-object, whereas the pairwise interaction takes into account constraints on object overlapping as well as on tree type competition. Compared to the former applications, the configuration space

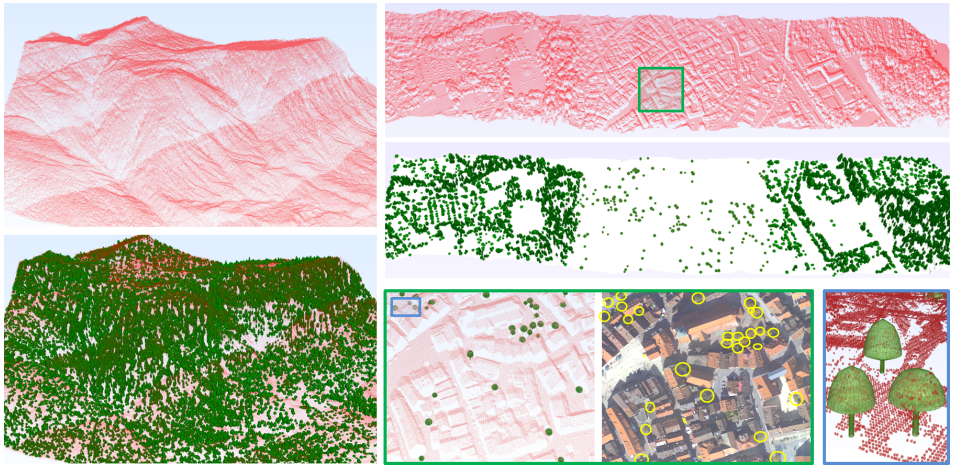


Figure 3.16: Tree recognition from point clouds by a 3D-point process specified by 3D-parametric models of trees. Our algorithm detects trees and recognizes their shapes in large-scale (left, input scan: 13.8M points) natural and (top right, input scan: 2.3M points) urban environments, in spite of other types of urban entities, e.g. buildings, car and fences, contained in input point clouds (red dot scans). An aerial image is joined to (bottom right) the cropped part to provide a more intuitive representation of the scene and the tree location. Note, on the cropped part, how the parametric models fit well to the input points corresponding to trees.

$\mathcal{C}$ is of higher dimension since the objects are parametrically more complex. This allows our algorithm to exploit more deeply its potential. The rotation kernel is not used here since the objects are invariant by rotation. However, a switching kernel is used in order to exchange the type of objects (*i.e.* conoidal, elliptical and semi-elliptical). The probability $\Pr(t)$ of choosing the perturbation type t is set to 0.4 if t ='birth and death', 0.2 if t ='translation', 0.1 if t ='switching', and 0.3 if t ='scale' in our experiments.

Fig. 3.16 shows results obtained from laser scans of large urban and natural environments. 30 (respectively 5.4) thousand trees are extracted in 96 (resp. 53) minutes on the 3.7km^2 mountain area (resp. 1km^2 urban area) from 13.8 (resp.

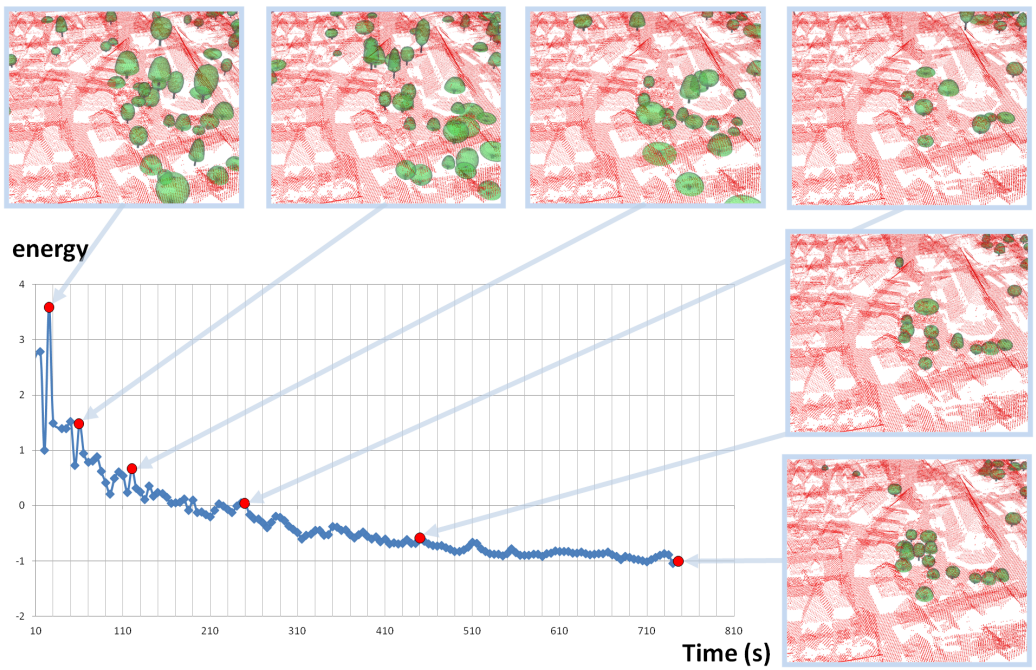


Figure 3.17: Evolution of the object configurations during the sampling. At high temperature, objects of low quality are frequently accepted, leading to non relevant object configurations (top left close-ups). When the temperature decreases, the process becomes progressively selective (top right close-ups). At low temperature, the current object configuration evolves through some local adjustments: the process is stabilizing close to the global minimum (bottom right close-ups).

2.3) million input points. The computation times can appear high, but finding non-trivial 3D-objects in such large scenes by point processes is a challenge which, to our knowledge, has not been achieved until now due to the extreme complexity of the state space. Note also that the performances could be improved by reducing the space $\mathcal{C}$ with a 3D-point process on manifolds, *i.e.* where the z-coordinate of points is determined by an estimated ground surface.

Evaluating the detection quality with accuracy for this application is a difficult task since no ground truth exists. As illustrated on the cropped part in Fig. 3.16, we have manually indexed the trees on different zones from aerial images acquired with the laser scans. The objects are globally well located and fitted to the input points with few omissions, even when trees are surrounded by other types of urban entities such as buildings. The non-overlapping constraint of the energy allows us to obtain satisfactory results for areas with high tree concentration. Errors may occur in distinguishing the tree type in spite of the tree competition term of the energy. Fig. 3.17 shows the evolution of object configurations during the sampling procedure.

3.3.3 Experiments with Markov Random Fields

As Markov point processes can be seen as a generalization of MRFs, we evaluated the potential of our sampler for optimizing energies for MRF-based labeling problems. Indeed, traditional MRFs represent simplified point processes having the two following characteristics:

(i) the dimension of the configuration space is fixed (graph structure is not dynamic anymore, but static),

(ii) the parametric objects become labels, *i.e.* a finite set of integers.

Under these conditions, the standard energy form of Markov point process (Eq. 3.5) can be reformulated as

$$U(l) = \sum_{i \in \mathcal{V}} D_i(l_i) + \sum_{(i,j) \in \mathcal{E}} V(l_i, l_j) \tag{3.19}$$

where $\mathcal{V}$ is the set of vertices (pixels in case of images), $\mathcal{E}$ is the set of edges (i.e. pairs of adjacent vertices), and $l \in [1, N]^{card(\mathcal{V})}$ is a configuration of labels over $\mathcal{V}$ with N, the number of labels of the problem. This formulation is actually the standard MRF-energy for labeling problems, as described by [Szeliski 2008]. Note that there is no constraint imposed on the form of the pairwise interaction term V when using our sampler. This point constitutes an important advantage compared to graph-cut based methods.

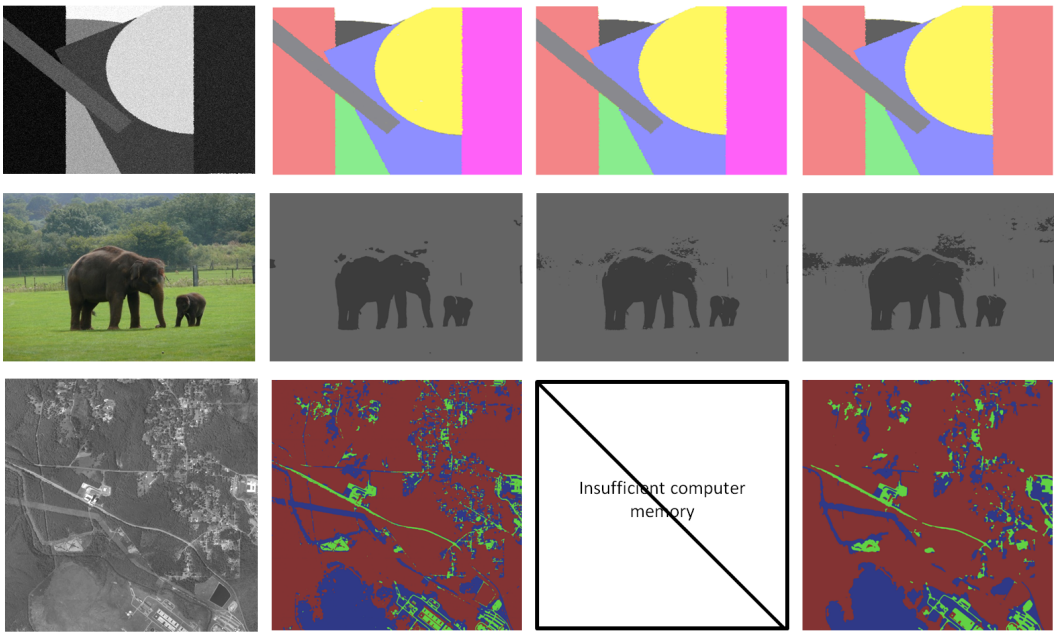


Figure 3.18: Image segmentation. (left) Input images (from top to bottom: Simulated, Elephants and Aerial). (second column) Results obtained by our sampler, by (third column) α-expansion, and by (last column) ICM. The goal here is not to evaluate the segmentation model which is obviously not optimal, but to compare the results from various optimization techniques with different input image sizes and neighboring distances ε. Images courtesy of *Google* [Google 2013]

In order to compare the potential of our sampler with the other optimization algorithms, we assume that no data knowledge can be extracted from the input images. A regular partitioning scheme (Fig. 3.4) is then used, and the label at each pixel are perturbed randomly. The width of a cell is given by the cell independence

condition (Eq. 3.15). As $\delta_{max} = 0$ (objects are fixed spatially), the width of cells must be superior or equal to the limit distance ε of the neighborhood relationship. For instance in case of an image labeling problem with a 4- or 8-connexity neighborhood, the condition simply implies that each cell can be associated to one single pixel.

	our sampler		α-expansion		α-β swap		BP		ICM	
	energy	time(s)	energy	time(s)	energy	time(s)	energy	time(s)	energy	time(s)
Simulated										
260k pixels	1.3192	**0.92**	1.3187	6.759	1.3186	7.71	**1.3185**	8.25	4.1186	4.7
8 labels	($\times 10^3$)		($\times 10^3$)		($\times 10^3$)		($\times 10^3$)		($\times 10^3$)	
$\varepsilon = 1$ pixel										
Elephants										
10M pixels	40.74	**40.01**	40.64	45.22	40.64	73.84	**40.63**	209.7	40.77	55.5
2 labels	($\times 10^3$)		($\times 10^3$)		($\times 10^3$)		($\times 10^3$)		($\times 10^3$)	
$\varepsilon = 1$ pixel										
Aerial										
72M pixels	**3.751**	**2047.6**	-	-	-	-	-	-	18.69	132.6
3 labels	($\times 10^6$)								($\times 10^6$)	($\times 10^3$)
$\varepsilon = 21$ pixels										

Table 3.3: Performances of various optimization algorithms in terms of reached energy and computation time. From a small image, the gain of performance of our sampler is relatively minor: the reached energy is similar to graph-based methods while running times are slightly improved. From a big image and a large neighborhood distance, e.g. Aerial, our algorithm becomes more interesting as the conventional graph-based approaches encounter memory problems whereas deterministic iterative methods, e.g. ICM, are extremely slow. Note that the energy value from the ICM algorithm can be poor as this optimization method gets stuck in local minimums.

The performances of our sampler have been tested from a basic model for image segmentation. The unitary data term $D_i(l_i)$ measures the quality of label l_i at

pixel i through Gaussian distributions. More precisely, the radiometry distribution of each class is modeled by a Gaussian law whose mean and standard deviation are model parameters to be estimated or fixed by a user. The potential V corresponds to the conventional Potts model [Li 2001] so that the labeling is smoothed in a local neighborhood of length ε.

Fig. 3.18 and Tab.3.3 show the performances obtained from various images and provides comparisons with the standard optimization techniques [1], *i.e.* max-product Belief Propagation (BP) [Weiss 2001], Graph-Cut based algorithms [Boykov 2001] and Iterated Conditional Modes (ICM) [Besag 1986]. Our sampler competes well with these algorithms. From a small input image, the reached energy is usually slightly higher than by using α-expansion, α-β swap or BP, but the computation time is lower. The gain of time becomes especially attractive when the image size and the neighborhood length increase. In particular, the results obtained from *Aerial*, show that our sampler proposes an interesting alternative to the standard graph-based optimization techniques which encounter memory problems from such very big images, here $8,500 \times 8,500$ pixels with neighborhood radius $\varepsilon = 21$ pixels.

Our sampler benefits from three advantages compared to graph-cut methods: (i) it can be performed from very big images without memory problems, (ii) the potential term V can have any form, and (iii) data-driven proposition kernels can be introduced in the sampler. For instance, one can use an edge detector on the input image to support a space-partitioning tree in which the lowest levels focus on discontinuities whereas the highest levels target the homogeneous zones of the image. Note also that the neighborhood distance ε can be considered as variable on the image domain. In particular, one can spatially adapt the subdivision stopping criterion of cells in function of local maximal neighborhood distances.

1. GCO C++ library (http://vision.csd.uwo.ca/code/).

3.4 Summary

We proposed a new algorithm to sample point processes whose strengths rely first, on the Markovian assumption that enables the sampling to be performed in parallel and second, on the adoption of a data-driven mechanism allowing efficient distributions of the points in the scene. Our algorithm improves the performances of the existing samplers in terms of computation times and stability, especially on large scenes where the gain is very considerable. It can be used without particular restrictions, contrary to most samplers, and even appears as an interesting alternative to the standard optimization techniques for MRF labeling problems.

Contributions: We presented an original solution to address the problem of finding a fast and efficient sampler for MPP which drastically reduce computation times while guaranteeing convergence stability and quality of the reached configurations. Our algorithm brings several important contributions to the field.

Sampling in parallel. Contrary to the conventional MCMC sampler which makes the solution evolve by successive perturbations, our algorithm can perform a large number of perturbations simultaneously using a unique chain. The Markovian property of point processes is exploited to make the global sampling problem spatially independent in a local neighborhood.

Data-driven mechanism. Point processes mainly use uniform proposition kernels which are computationally easy to simulate, but make the sampling particularly slow. We proposed an efficient mechanism allowing the modifications, creations or removals of objects by taking into account spatial information extracted from the observed data. Contrary to the data-driven solutions proposed by [Tu 2002] and [Ge 2009], our proposition kernel is not built directly from image likelihood, but is created via a space-partitioning tree in order to guarantee the sampling parallelization.

Efficient GPU implementation. We proposed an implementation on GPU which significantly reduces computation times with respect to existing algorithms, while increasing stability and improving the quality of the obtained solution.

The potential of GPU is efficiently exploited in both optimizing the number of operations in parallel, and limiting the memory transfer between GPU and CPU.

Original models for urban reconstruction. To evaluate the performance of the sampler, we proposed original point processes for vision problems. In particular, a model for detecting complex 3D objects in large-scale point clouds is designed. This model is applied to tree recognition from laser scans of large urban and natural environments. To our knowledge, it is the first point process sampler to date to perform in such highly complex state spaces.

Limitations: While we properly addressed the central limitation of MPPs that is the convergence time during optimization, few issues remain. First, our novel optimization scheme is efficient on condition that the templates are small compared to the scene, so that numerous independent sub-problems can be solved simultaneously. Second, common optimization methods for MPPs need a stochastic relaxation of the energy, which is refereed as simulated annealing where the temperature parameter plays a major role in the convergence quality. The starting temperature, as well as the decreasing speed factor - the cooling schedule - have been estimated by conventional methods. In this context, our method is not different than previous optimization methods: we have not proposed improvements or advances in this aspect.

The technical assumption of small templates is not the only limiting factor in this method. As every top-down approach, MPPs are intrinsically limited by the library of objects used for modeling (in the context of urban reconstruction from Lidar data, three types of template trees, i.e. ellipsoidal-shaped tree, conic-shaped tree, and semi-ellipsoidal-shaped tree). Since only simple parametric-based models can be used, MPPs are not adapted for modeling building structures. Although few papers tried developing in this direction [Ortner 2008, Lafarge 2010b], all agreed that the resulting urban models are simplistic and limited by the library of objects; MPPs are not flexible enough to represent complex building structures. Therefore, this top-down approach fits well the model-based tree detection but is inappropriate

for reconstructing buildings.

Perspectives: In future work, it would be interesting to extend the algorithm to point processes in 4D for addressing spatio-temporal problems in which 3D-objects evolve during time. Also, one could improve the algorithm to optimize MRF-based energy formulations and develop a more competitive solution by modeling a data-driven proposition kernel whose density map is of lower intensity in homogeneous regions.

Several parameters are needed to define the underlying mathematical model. When correctly chosen, these parameters reduce the search space, rendering the problem more tractable. However, the current implementation relies on tedious manual fine-tuning of parameters. A learning approach for these parameters by supervised techniques and the use of heuristics to automatically set the cooling schedule parameters [Varanelli 1999, Ben-Ameur 2004, Ben Hadj 2010] have yet to be explored. This would firstly render the corresponding step less time-consuming and secondly improve the performances by using the parameter settings optimal in a certain way. Both convergence time and quality are thus likely to benefit from an automatic learning step compared to the current manual trial-and-error task.

Due to the above mentioned limitations of the top-down approaches, bottom-up solutions are considered for complex building reconstruction. In the next chapter, we describe two bottom-up solutions for airborne Lidar and MVS data. As we will see, these solutions rely on few assumptions on the urban scene structures and are appropriate for reconstructing complex buildings.

Chapter 4

Methods for urban mesh reconstruction

In this chapter, we will present the last two algorithms for Lidar and MVS data which consist in generating 3D-models of urban scenes. While for both data types, the aim is to automatically reconstruct 3D urban elements, the designed methods for Lidar data (Section 4.2) and MVS meshes (Section 4.3) are different. The fundamental difference in data requires different strategies for the labeling (Section 2.2.1 and Section 2.3.1) and for the reconstruction of the urban scene (Section 4.2 and Section 4.3). Since the labeling step has been described before, we focus here on the reconstruction approaches.

A brief overview of the various techniques used for Lidar and MVS data along with the motivation are given bellow in Section 4.1, followed by description of the methodology for modeling urban elements (Section 4.2 and Section 4.3). Finally, in this chapter, we will perform various experiments to validate both methods (Section 4.4).

4.1 Introduction

Automatic city modeling has experienced an increasing interest in Computer Graphics and Computer Vision in the last decade. This research topic is particularly challenging as it combines scene understanding, object modeling in 3D and large scale processing as mentioned in [Vanegas 2010b, Musialski 2013]. Two disctinct problems are usually addressed in the literature: city reconstruction and city generation. The former aims to provide accurate geometric models of urban scenes from

physical measurements as detailed in [Musialski 2013], mostly in an automatic way. The latter uses procedural modeling to generate realistic 3D models of cities highly semantized from grammatical rules, with possible user interaction [Vanegas 2010b]. In these following, we focus on the former city reconstruction problem.

The literature on urban scene reconstruction from airborne acquisitions is relatively rich, with various directions having been explored [Musialski 2013]. An initial approach was using depth images from stereo vision. This constitutes 2.5D view-dependent inputs for modeling cities which are usually highly noisy. The works of [Zebedin 2008] and [Lafarge 2010b] propose compact 3D-models of buildings from depth maps: the former labels a 2D space partition driven by geometric primitives, whereas the later assembles 3D-blocks of urban structures using Monte Carlo sampling. The major limitation of depth maps comes from the difficulty in distinguishing buildings from high vegetation. Other types of data are commonly used because they do not suffer too much from this limitation, such as Lidar and MVS data. These have interesting properties for urban reconstruction (see Section 1.2).

4.1.1 Motivations for Lidar data.

Lidar data became very popular in the mid-2000 leading to a series of works mainly focused on parsing building components and accurate extraction of building contours, e.g. [Poullis 2009, Toshev 2010, Lafarge 2012, Zhou 2012, Lin 2013]. Points generated by Lidar acquisition systems are usually very accurate, but contrary to depth maps, they are geometrically unstructured and are free of radiometric information. Planar primitives represent the favorite geometric tools for recovering roofs and facades. Efforts have been made towards parsing the planes, e.g. [Toshev 2010], discovering global regularities [Zhou 2012], or decomposing the different structures of buildings [Lin 2013]. These methods mainly focus on building reconstruction from a geometric point of view.

Some methods address the urban reconstruction problem by inverse procedural modeling which consists in finding grammatical rules from physical measurements. This approach proposes convincing and highly semantized models of facades, but

remains lightly explored for airborne based city reconstruction as construction rules are harder to analyze and define.

4.1.2 Motivations for MVS data.

With the recent advances in Multi-View Stereo (MVS) [Seitz 2006], some efficient techniques have been developed for creating dense meshes from high resolution images. Contrary to depth maps and Lidar scans, these triangular meshes that we call *MVS meshes* constitute real-3D representations in the sense that information is retrieved on the vertical components of the scene. As highlighted in Section 1.2.1.2 and illustrated on Fig. 1.3, MVS meshes are particularly accurate and provide an impressive amount of details, surpassing in quality the other 3D models for visualization-based applications. They constitutes a new alternative to standard depth maps and Lidar scans for tackling automatic city reconstruction problems. Efforts have been made towards inserting primitives in mesh, for instance, [Lafarge 2010c] use region growing and iterative non-linear minimization for fitting usual primitives, [Cohen-Steiner 2004] use a variational approach to approximate the mesh with a set of shape proxies, or [Li 2011] use global mutual relations between primitives detected from RANSAC to generate an idealized synthetic model of the mesh.

4.2 Approach with Lidar data

4.2.1 Building structure extraction

Once the point cloud is classified using the formalism presented in Section 2.4.1, the next step consists in extracting structural information from the *building* elements of the scene.

Building footprints- Building footprints are extracted by projecting the points labeled as *building* and *NC* on a XY-grid, and then by locally propagating the information on the empty cells of the grid. As the building facades are located in between the points labeled as *ground* and the points labeled as *building* or *NC*,

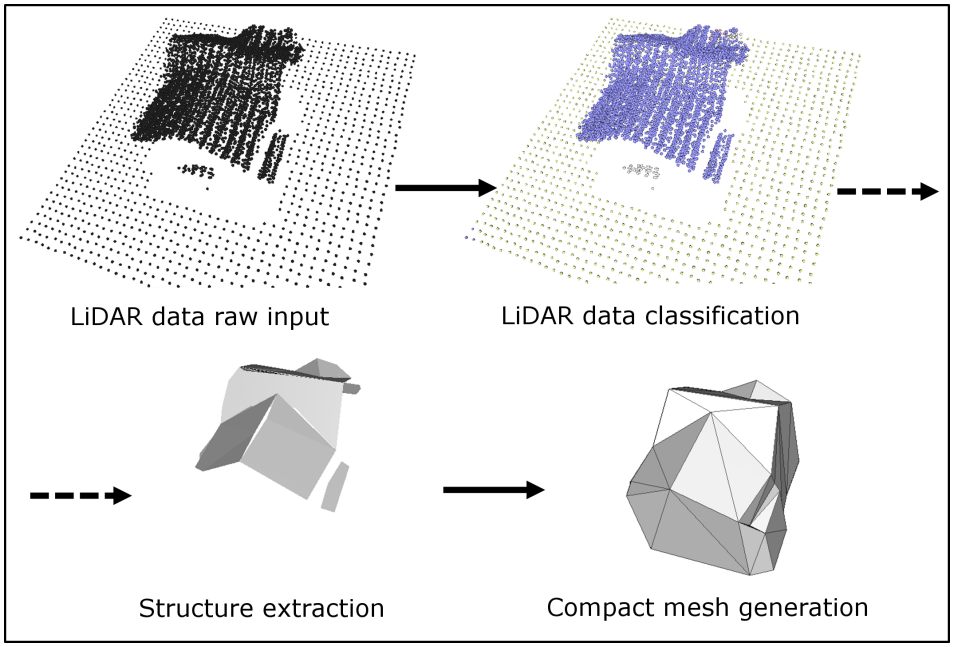


Figure 4.1: Pipeline of the proposed approach. Note that the Lidar data classification has been described previously in Section 2.2.1

the building boundary is dilated by one cell-size.

Roof sections- The *planar components of the roofs* are detected by region growing. One could decide to also extract non-planar elements using an iterative non-linear minimization such as Levenberg-Marquardt optimization, but this solution was not retained because of its expensive CPU cost. Indeed, a non-planar roof can be approximated by a side-by-side sequence of small planar clusters with a satisfactory result in most cases. The propagation of the region growing is based on the regularity of the point normals in a local neighborhood.

The *roof contours* are located through a sequence of connected 3D-segments. The points representing the building edges are detected using the 2D Alpha-shape algorithm [Sack 2000]. Indeed, this method is a generalization of the convex hull method and obtains better results than the original convex hull because the shape of each roof is not necessary convex. However, it requires an additional parameter α such as each edge of the resulting contour has a circle of radius α empty. The

triangulation of the 2D point cloud (only XY value of each point is considered) is computed to determine a set of possible α-shapes. As the point cloud density in our experiments is about 2 points per square meter, the first α-shape with $\alpha > 3$ is considered. The contour is then simplified using an adaptive Douglas-Peucker algorithm [Sack 2000]. The parameter ε, which is required to stop the procedure, constitutes a crucial point because it controls the simplification level of the contours. It is computed by taking into account the size of the contour:

$$\varepsilon = \varepsilon_{Min} + (\varepsilon_{Max} - \varepsilon_{Min}).(1 - e^{-\frac{nb}{RC}}) \tag{4.1}$$

where nb is the number of points of the contour, ε_{Min} and ε_{Max} are respectively the lower and upper bounds fixed to 0.03 and 0.3, and RC is the mean number of points fixed to 50. One can understand that a small contour requires more details (low ε) whereas a large contour does not (high ε).

Finally, some *structure adjustments* are made on the roof sections in order to connect them when they are close enough. A neighborhood relationship between roof sections is defined: two roof components are neighbors if their Euclidean distance is inferior to one meter. For each pair of neighboring roof sections, the roof contour points of interest are then projected on the line intersecting the two planes. While this step may be sufficient for simple building (e.g. building #1 on Fig. 4.8) , it does not guaranty a watertight reconstruction.

4.2.2 Compact mesh generation

The last stage allows us to generate a watertight and compact mesh from the structural information extracted previously.

Mesh initialization- An initial mesh is generated from the regular XY-grid used for computing the building footprints. The cells of the grid are labeled as *interior*, *dilated*, or *boundary* depending on whether the cell belongs to the inside, the dilated, or the boundary sectors of the building footprints (see Fig. 4.2). For each *interior* or *dilated* cell, we test whether the cell center is located inside the planimetrical projection of the roof sections extracted in Section 4.2.1. Two cases

have to be distinguished:

- The cell center belongs to one or more roof sections: For each concerned roof section, the cell center is vertically projected on the 3D-plane to create a potential vertex. The vertex having the highest Z value is kept to generate the mesh.
- The cell center does not belong to a roof section: The vertex is computed as the vertical projection of the cell center to the closest roof section. However, only *interior* cells are considered here in order to contain the propagation. This case allows to fill the holes inside the mesh without changing its topology.

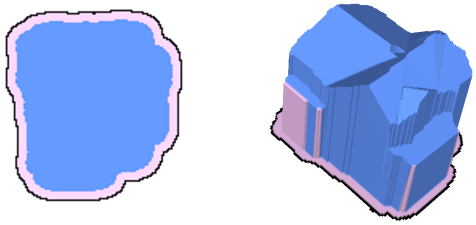


Figure 4.2: Mesh initialization - Building footprint (*left*) and initial mesh (*right*). The three classes *interior*, *dilated* and *boundary* are respectively represented in blue, pink and black.

Mesh simplification with topology preservation- A quadric edges collapse decimation based algorithm [Garland 1997] using topological preservation constraints is iteratively used to efficiently reduce the size of the mesh. This process preserves the mesh topology by using a quadratic error measure for the cost function and by minimizing the new location error for the placement function. Thus, planar based structures of the building are preserved during the mesh simplification. The decimation process is stopped when the number of not-*interior* vertices falls down under 1% of its initial number. As a post-process step, the remaining facets from the *boundary* are deleted. The result is illustrated on Fig. 4.1.

4.3 Approach with Multi-View Stereo data

4.3.1 Surface approximation

After the identification and the extraction of the urban elements from the scene (see Section 2.3.1), the next step consists in identifying the geometric shape of the objects in order to simplify it to compact and structured 3D-models. Buildings, trees, and ground are processed separately. The problem with surface approximation comes from the fact that the geometry of buildings is highly structured by definition. A volumetric labeling in a space partitioned by planes is adopted to guarantee compactness, multi levels of structures and preservation of semantical information.

Plane hypothesis. Planar primitives are extracted from the input mesh components labeled as *roof* and *facade*. Instead of detecting planes by standard techniques based on Ransac or region growing algorithms, we rely on the already existing curvature-based clustering performed in Section 2.3.1. f-clusters with a low planarity attribute a_p and with a high area A are eligible to generate planes. In our experiments, a f-cluster is selected if $a_p < 0.3$ and $A > 10m^2$. For each selected f-cluster, the plane minimizing the least square error to its vertices is then extracted.

Global regularization of planes. Imposing global regularities on plane hypotheses allows the models of buildings to be visually more consistent by re-adjusting plane positions and orientations. Discovering global regularities is also of interest for reducing the complexity of the subsequent space partitioning by the planes.

Several methods have been proposed in the literature for interactive geometric modeling [Habbecke 2012], Lidar-based building reconstruction [Zhou 2012], or more generally geometric shape fitting under regularization constraints [Li 2011]. These methods are very efficient with individual objects containing few primitives, but are not adapted to the scale of city in which thousands of planes are usually detected per km^2, leading to very high running times.

To address this problem, we propose a procedure based on barycentric operations. Four types of pairwise interactions between planes are considered. By

denoting P_1 and P_2, two planes having respective unit normals $\mathbf{n}_1$ and $\mathbf{n}_2$ and centroids c_1 and c_2, one can formulate these relationships under an orientation tolerance ε and an Euclidean distance tolerance d:

- *Parallelism.* P_1 and P_2 are ε-parallel if $|\mathbf{n}_1 \cdot \mathbf{n}_2| \geq 1 - \varepsilon$
- *Orthogonality.* P_1 and P_2 are ε-orthogonal if $|\mathbf{n}_1 \cdot \mathbf{n}_2| \leq \varepsilon$
- *Z-symmetry.* P_1 and P_2 are ε-Z-symmetric if $||\mathbf{n}_1 \cdot \mathbf{n}_z| - |\mathbf{n}_2 \cdot \mathbf{n}_z|| \leq \varepsilon$, where $\mathbf{n}_z$ is the unit vector along the vertical axis
- *Coplanarity.* P_1 and P_2 are d-ε-coplanar if they are ε-parallel and $|d_\perp(c_1, P_2) + d_\perp(c_2, P_1)| < 2d$, where $d_\perp(c, P)$ represents the orthogonal distance between point c and plane P

Three relationships, *i.e.* parallelism, orthogonality and Z-symmetry, are dedicated to plane orientation. Coplanarity is a particular case of parallelism, including a relative positioning constraint on planes. These four relationships constitute the most useful regularities that can be taken into account for modeling buildings with planar elements.

The procedure adopted for regularizing planes relies on four steps:

- Planes are first regrouped according to parallelism. We call a *parallel group*, a set of planes which are mutually ε-parallel.
- Relationships of orthogonality and Z-symmetry are then detected between the parallel groups. Two parallel groups are ε-orthogonal (respectively ε-Z-symmetric) if their respective area-weighted barycentric planes are ε-orthogonal (resp. ε-Z-symmetric) [1].
- The area-weighted barycentric plane of each parallel group is re-oriented according to orthogonality and Z-symmetry. We propagate the information of orthogonality and Z-symmetry in the orthogonal graph from the parallel group with the highest cumulated area to the parallel group with the lowest cumulated area. The orientation of each visited plane is adjusted by constraining its normal to be orthogonal, and eventually Z-symmetric, to the normal of the

1. The area of a plane corresponds to the area of its f-cluster.

parallel group diffusing the information. The re-adjusted normal is assigned to every plane of the considered parallel group. This scheme allows us to avoid the potential conflicts of relationships as the propagation does not loop in the orthogonality graph, priority being given to the parallel groups of higher cu-

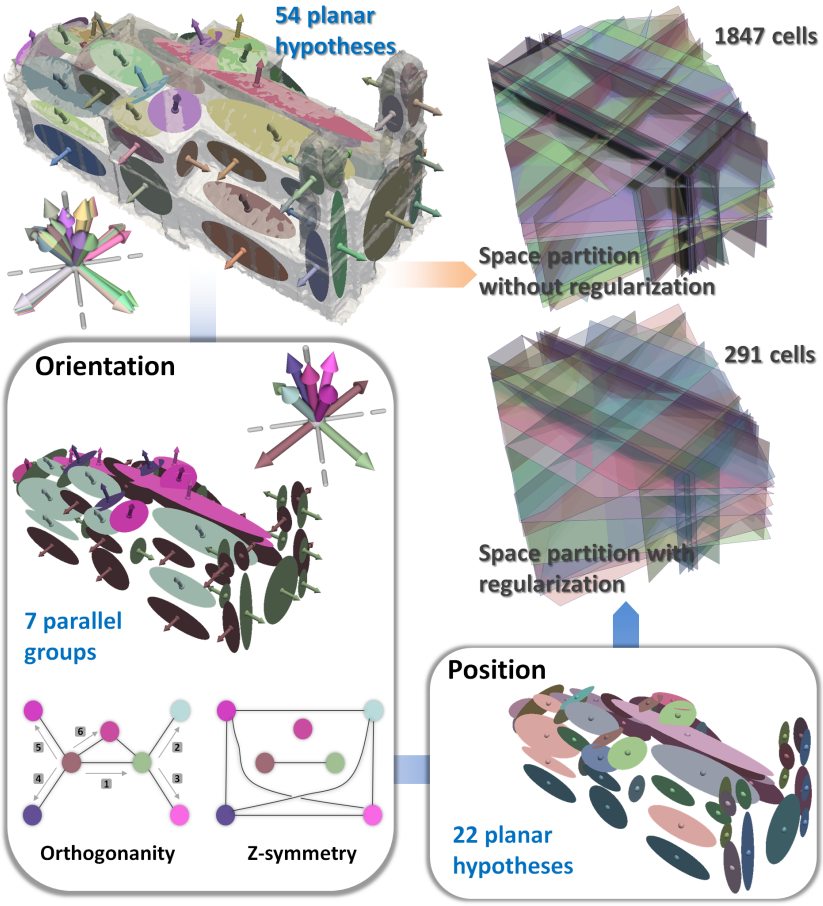


Figure 4.3: Plane regularization. Partitioning the space from the initial planar hypotheses (top left) leads to a dense and irregular set of cells (top right). Our plane regularization procedure both provides structural coherence within plane hypotheses and strongly reduces the number of cells in the space partition. Each color in the orientation step (respectively in the position step) represents a parallel group (resp. a set of coplanar planes). Plane normals are re-adjusted during the orientation step by propagating orthogonality and Z-symmetry constraints from the biggest parallel group to the smallest one (see numbers on the orthogonality graph).

mulated area. Some relationships between parallel groups can be broken, but this is the price to pay for linear complexity and fast running time.

– Sets of coplanar planes are extracted from each parallel group. Mutual d-ε-coplanar planes are regrouped, and their position is readjusted onto the area-weighted barycentric centroid of the planes.

As illustrated on Fig. 4.3, this procedure constitutes a fast solution to both reduce the plane hypotheses and reinforce regularities without iteratively re-fitting primitives as in [Zhou 2012] or [Li 2011]. Few seconds are required for regularizing several thousands of planes. In our experiments, the orientation tolerance ε and the distance tolerance d are fixed to 0.05 (*i.e.* approximately a 3 degree angle) and one meter respectively.

4.3.2 Surface extraction with discrete formulation.

Whereas the plane regularization is performed on the entire scene, the surface extraction from the space partition induced by the planes can be realized at a more local scale as building blocks or isolated houses are spatially disconnected. We therefore separate the connected components labeled as *facade* or *roof* into a set of sub-meshes so that they can be treated independently.

Computing the exact geometry of a 3D space partitioned by planes is a critical operation whose algorithmic complexity is very high. Existing works relying on plane arrangements try to reduce the algorithmic complexity. In [Furukawa 2009], plane hypotheses are restricted to the Manhattan-world assumption. This solution, leading the authors to consider a 3D-grid as a partition, is quite limited in practice. The method proposed by [Chauve 2010] consists in partitioning the space with polyhedral cell complex, one advantage being to only cut cells close to the planar primitive. Running time can however easily exceed half an hour when dealing with only few hundred planes. Instead, we propose an original discrete formulation of the problem by combining Binary Space Partitioning (BSP) trees with a volumetric occupancy grid. The key idea is to avoid the time-consuming computation of the exact geometry of the cells during the partitioning of the space.

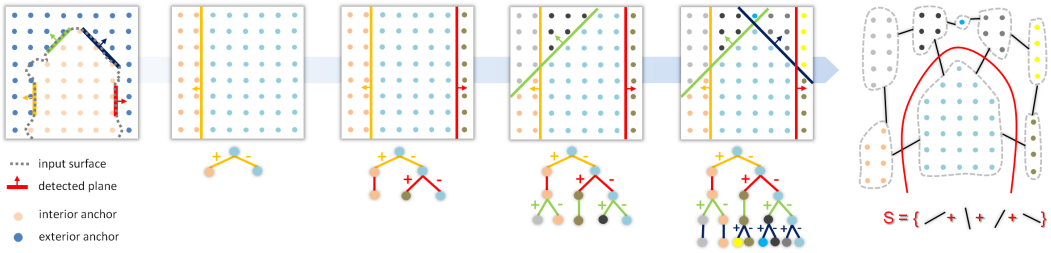


Figure 4.4: Discrete space partitioning. Anchors are labeled as interior or exterior to the input mesh by ray casting (left). At each plane insertion, both the anchors of the discrete space (top) and the BSP tree (bottom) are updated. After the last plane insertion, the anchor set is decomposed into discrete cells (right, colored points) from which one can compute a discrete volume or the ratio of interior/exterior anchors, as well as identify the adjacent cells. Once the optimal cut of our discrete problem is found, the surface can be extracted by computing the exact geometry of facets from the BSP tree (black edges crossed by the red cut).

We first create an oriented bounded box of the sub-mesh in which points are regularly sampled according to the three directions of the bounding box. Such a point, called an *anchor*, is associated to a cell index and to a binary number specifying whether the point is interior or exterior to the input mesh surface. As the mesh is not watertight, ray casting is used to distinguish interior from exterior anchors. The grid width specifying the distance between two neighboring anchors is a parameter of the algorithm. The lower the value, the more accurate the result. In our experiments, this value is fixed to 0.5 meter. As illustrated in Fig. 4.4, plane hypotheses are successively embedded in the bounding box, leading to the construction of a BSP tree. Instead of computing the exact geometry of the cells for every embedded plane, we only update the cell index associated to each anchor. This discrete scheme allows us to quickly estimate cell information such as volume, adjacency, facet areas and position with respect to the input mesh.

A min-cut formulation is used to find the inside/outside labeling of the cells, the surface being at the interface between inside and outside. Let us consider a graph $(\mathcal{C}, \mathcal{E})$. $\mathcal{C} = \{c_1, ..., c_n\}$ is the set of nodes corresponding to the cells induced

by the space partition. $\mathcal{E} = \{e_1, ..., e_m\}$ is the set of edges representing the facets separating two adjacent cells. A cut in the graph $(\mathcal{C}, \mathcal{E})$ consists in separating the set of cells $\mathcal{C}$ in two disjoint sets $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$. The set of edges between $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$ corresponds to a set of triangular facets forming a surface $\mathcal{S} \subset \mathcal{E}$.
In order to measure the quality of the surface $\mathcal{S}$ induced by the cut $(\mathcal{C}_{in}, \mathcal{C}_{out})$, we introduce a cost function C of the form

$$C(\mathcal{S}) = \sum_{c_k \in \mathcal{C}_{out}} V_{c_k}\, g(c_k) + \sum_{c_k \in \mathcal{C}_{in}} V_{c_k}\, (1 - g(c_k)) + \beta \sum_{e_i \in \mathcal{S}} A_{e_i} \tag{4.2}$$

where V_{c_k} is the discrete volume of cell c_k given by the number of anchors in c_k, $g(c_k)$ is a function measuring the coherence of the cell label with respect to the ratio of interior/exterior anchors, and A_{e_i} is the discrete area of facet e_i computed as the number of pairs of anchors (i) each belonging to one of the two cells, and (ii) adjacent in a standard 6-connexity neighborhood. The two first sums of the cost function C can be seen as a data term in the conventional energy formulation whereas the right sum weighted by the parameter $\beta \geq 0$ acts as a regularization term favoring small area surfaces. The optimal cut minimizing the cost $C(\mathcal{S})$ is obtained using the max-flow algorithm [Boykov 2004].

Function $g(c_k)$ is defined in the interval [0,1]. It checks whether assigning the label *inside* to cell c_k is coherent with the ratio r_{in} of interior anchors contained in c_k. It is given by

$$g(c_k) = \frac{(2r_{in} - 1) \times |2r_{in} - 1|^{\alpha} + 1}{2} \tag{4.3}$$

where α is a real value tuning the data sensitivity of function g, as illustrated on Fig. 4.5. In some cases, a cell can be free of anchors. This situation arises when the cell is thin compared to the width of the occupancy grid. Such a cell is not taken into account in the cost function C as its discrete volume is null. This approximation leads us to potentially omit cells which are small at the scale of the grid width. Note however that such situations are rare in practice thank to plane regularization which limits the number of thin cells.

The optimal cut corresponds to a subset of facets composing the surface, as illustrated on Fig. 4.4. The exact geometry of these facets is then computed from

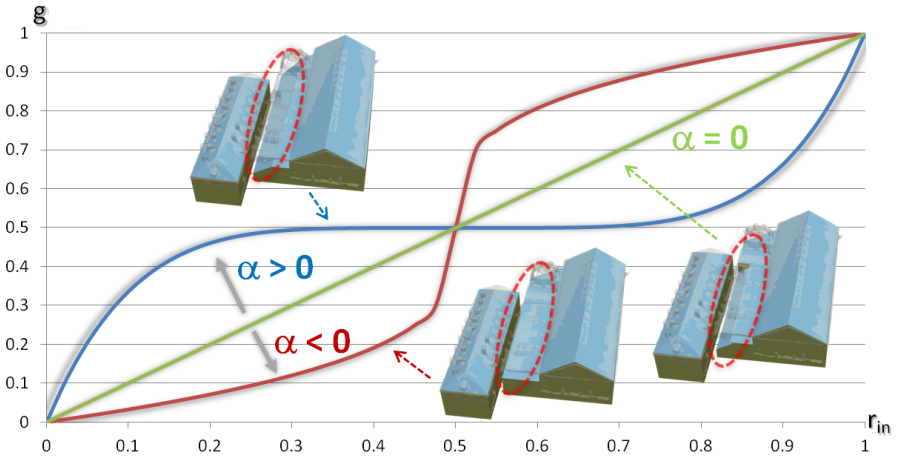


Figure 4.5: Behavior of the function g with respect to r_{in}. Choosing $\alpha = 0$ gives a linear penalization in function of the ratio r_{in}. Increasing α gives a more uniform penalization when the ratio r_{in} is close to 0.5. The impact of the data term in the cost function is reduced, favoring surfaces of low area (top model). If $\alpha < 0$, g strongly penalizes cells with a non consistent ratio.

the BSP tree, without operating the other facets. A continuous formulation of this space partition problem has been implemented using the Nef polyhedron data structure of [CGAL 2013]. The gain of time obtained by our discrete formulation is very high, running time being reduced by a factor close to 20 from the example of Fig. 4.5.

4.3.3 Modeling buildings at various Levels Of Details (LOD).

Depending on the application domain, the representation of buildings must contain a different amount of geometric details. As illustrated on Fig. 4.6, the users have the possibility to select between three different LOD.

– *LOD1:* It provides a simple description of buildings by representing facades by piecewise-planar components, and roofs by horizontal flat surfaces. Such a description is used for example to simulate the propagation of electro-magnetic waves to optimize the location of wireless antenna. We extract LOD1 surfaces by considering only vertical planes labeled as *facade* during the space parti-

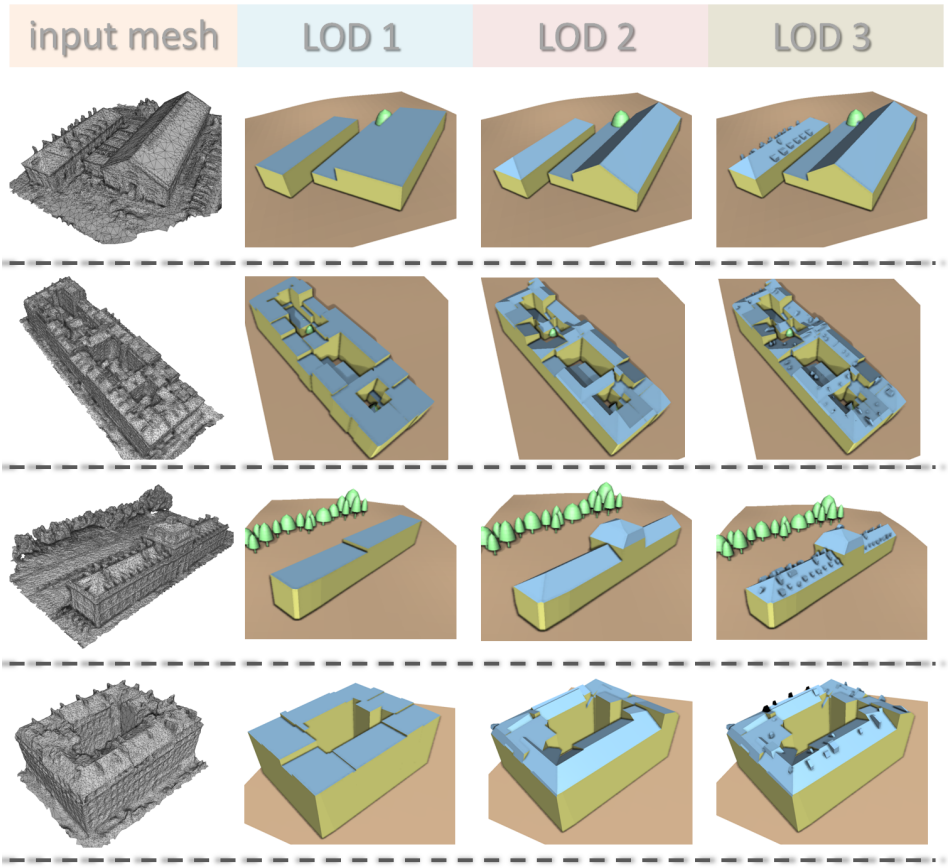


Figure 4.6: Levels Of Details (LOD). Our algorithm produces models of buildings at three different Levels Of Details (LOD). The LOD1 models provide a simplistic description of roofs, suitable to different types of urban areas such as typical financial districts or industrial complexes. The LOD2 and LOD3 models give a more detailed description of roofs using piecewise-planar structures. LOD2 and LOD3 models are well adapted to residential areas and dense downtowns.

tioning, the height of the horizontal roofs being given by the median z-value in the considered portion of the input mesh.

– *LOD2:* LOD2 models give a more refined description of the roofs by piecewise-planar components, breaking the LOD1 hypothesis of horizontal flat roofs. They are usually considered for urban planning applications. All the plane hypotheses are taken into account during the space partitioning for obtaining LOD2 surfaces.

– *LOD3:* This representation is similar to LOD2 in terms of facade components and roof sections, but also models the roof superstructures as chimneys or dormer-windows. LOD3 models are obtained by detecting small clutter components in the input mesh. More precisely, f-clusters labeled as *roof* but not flat enough to generate a plane hypothesis are selected and regrouped by connected components. A rectangular cuboid is then fitted on each f-cluster group, and integrated to the LOD2 model.

4.3.4 Modeling trees and ground.

The visual appearance of trees from MVS meshes is not appealing. As illustrated on Fig. 4.7, their shapes usually correspond to small domes with a wavy surface. A standard ellipsoidal tree model whose compaction and rendering is well adapted

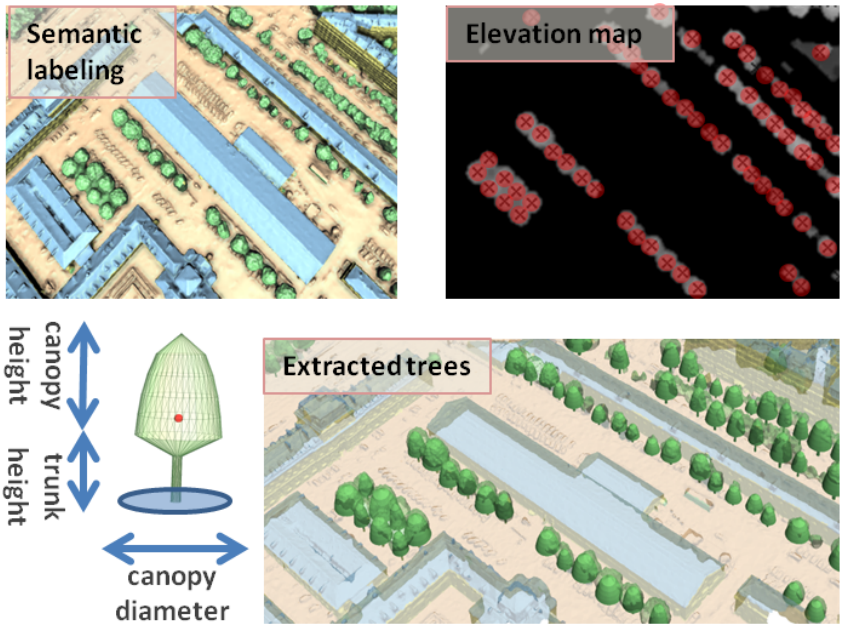


Figure 4.7: Detection and modeling of trees. Trees in MVS meshes (top left, green components) are detected from an elevation map by extracting the local maxima (top right, red dots representing the center of mass of the detected trees). Once the center of mass is located, three additional parameters (bottom left) are then found by fitting semi-ellipsoids to mesh vertices (bottom right).

to large urban scenes is considered to *idealize* these elements from the input mesh. As directly matching the tree models to a mesh is computationally expensive, the center of mass of each tree is first located by detecting the local maxima from an elevation map created from the parts of the input mesh labeled as *tree*. The other parameters of the template such as the height and the canopy radius are then found by minimizing the Euclidean distance from vertices to a semi-ellipse.

Ground contains many urban details, such as cars or traffic signals, which do not need to be preserved. A light mesh of the ground is produced by triangulating a set of points regularly positioned on a XY-grid, the Z-value of these points being given by the ground map used in Section 4.3 to compute the elevation attribute a_e.

4.4 Experiments

4.4.1 Experiments with Lidar data

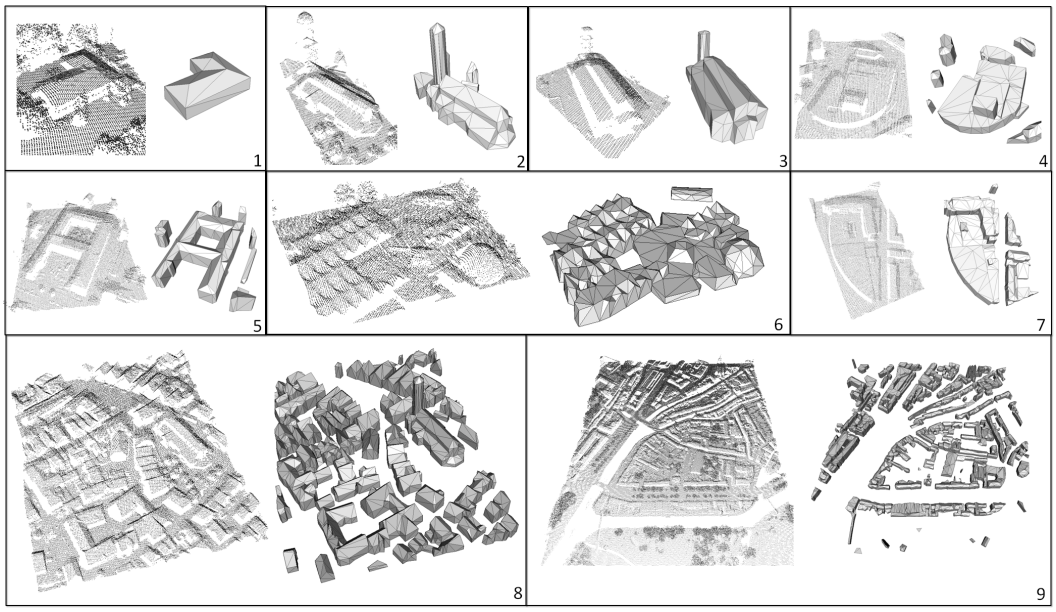


Figure 4.8: Reconstruction of various buildings and urban areas- Lidar point cloud (*left*), and our mesh-based model (*right*).

Our method has been tested on different point cloud densities. Fig 4.8 presents

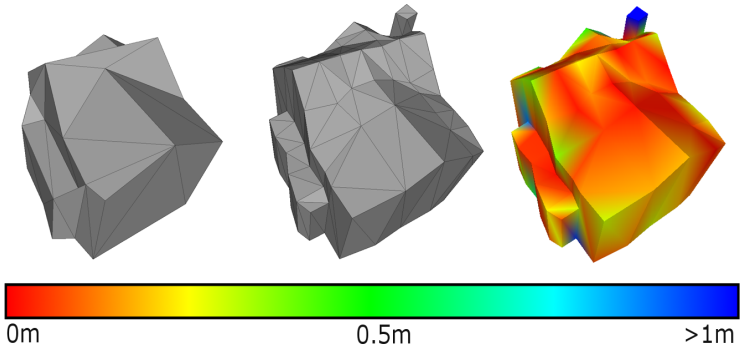


Figure 4.9: Compactness and accuracy evaluation- our mesh (*left*), mesh by Zhou et al. [Zhou 2010] (*middle*), and the Hausdorff distance of our mesh with respect to the reconstruction by [Zhou 2010] (*right*). Color scale: red (no error) to blue (error superior to 1 meter).

results obtained from various types of buildings and dense urban areas.

Visual considerations- The method proposes convincing 3D-models in which the building structure is correctly retrieved. As illustrated on the Fig 4.8 result #1, the building components are optimally represented by the mesh in terms of vertices and facets: each rectangular facade of the building is described by two complementary triangular facets, as well as each roof section is modeled by one or two facets. The non-planar roof sections are approximated by piecewise planar shapes: the visual rendering is not always satisfactory, as shown on the result #6 with the spherical roof.

Compactness and accuracy assessments- Our reconstruction is compared with the result obtained by [Zhou 2010] in Fig. 4.9. Our mesh has a better compactness (43 vertices / 69 facets VS 133 vertices / 228 facets) while having similar accuracy. Indeed, the average Hausdorff distance of our mesh with respect to the reconstruction by [Zhou 2010] is 0.16 meter. However, one can notice that a small chimney has been omitted in our reconstruction. The Hausdorff distance is maximal at this location, *i.e.* 1.12 meter. Our method focuses on the roof section reconstruction and is not designed for detecting the small planar

components such as dormer-windows and chimneys. Despite those small omitted structures, the average error is very satisfactory, particularly considering the gain in terms of compactness. Moreover, to make fair comparison with the result obtained by [Zhou 2010], we have chosen to use their method to generate a new mesh with similar average Hausdorff error of 0.16 meter. The mesh obtained still has a lower compactness (76 vertices / 124 facets) compared to our result.

Performances- The computing times are reasonable thank to the Computational Geometry Algorithms Library (CGAL) [CGAL 2013] used to perform our 3D-geometry operations. The result #8 on Fig. 4.8 representing a dense urban area (raw input: 139983 points) has been obtained in 9 minutes. The resulting mesh has 3450 vertices and 5788 facets.

4.4.2 Experiments with MVS data

Our algorithm has been tested from different datasets generated by [Hiep 2009] from Multi-View Stereo aerial images. The algorithm has been implemented in C++, using the Computational Geometry Algorithms Library [CGAL 2013] which provides the basic geometric tools for mesh analysis and generation. Several aspects have to be considered to evaluate our algorithm, in particular the semantic labeling correctness, the visual quality, the geometric accuracy and running times. Note that, as mentioned in [Musialski 2013], the evaluation of urban scene models is usually a difficult task as no benchmark nor Ground Truth (GT) exist in the field. Moreover most of existing automatic city modeling methods use Lidar inputs and cannot be adapted to the MVS mesh context. We instead compare with different standard surface approximation algorithms.

Robustness. As mentioned in Section 4.1, input meshes are dense and defect-laden. Results obtained during the object identification step are of good quality. Both regularizing term of the energy and the correction rules bring spatial consistency and limit identification errors, even in the presence of merged objects such as

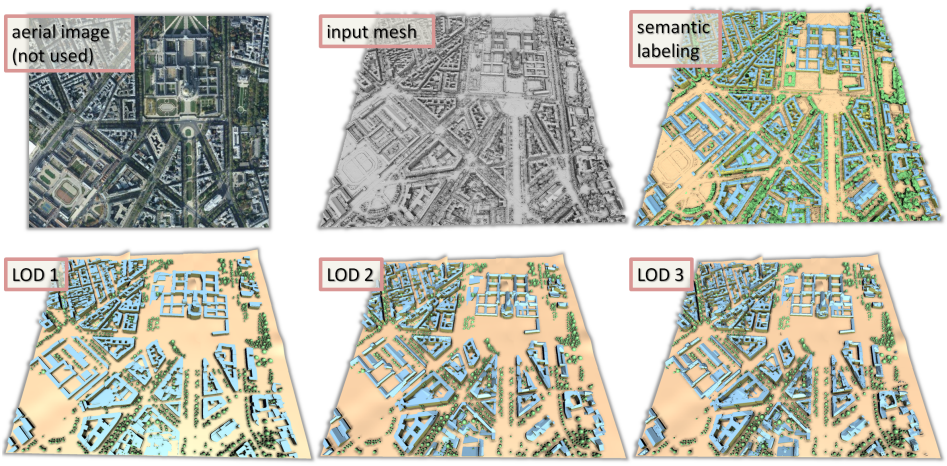


Figure 4.10: Large scale urban scene idealization. The 7^{th} district of Paris (approximately 1 km^2) is idealized with three levels of detail (bottom) from a dense MVS mesh (11M facets). The semantic labeling (top right) provides an accurate identification of the objects of interest (see visual comparison with the top left aerial image). The three LOD models are compact, geometrically-structured, and semantized. Image courtesy of *Google* [Google 2013].

facade/tree. Concerning surface approximation, the use of space partitions driven by planar hypotheses allow us to obtain compact and geometrically structured models of buildings with a better accuracy than the standard surface approximation techniques [Garland 1997, Cohen-Steiner 2004], as shown in Fig. 4.11. In particular, our models are not affected by holes and topological errors contained in the input mesh, as shown with the front facade of the church example. The LOD3 description has also an interesting advantage with respect to the standard surface approximation techniques as it gives compact models while preserving roof details such as chimneys and dormer-windows.

Performances. Performance aspects related to running time and scalability are important evaluation criteria as technical efforts have been made to design a fast algorithm able to idealize large urban scenes. Tab. 4.1 illustrates the performance of our algorithm from urban scenes of different sizes. The most time-consuming

operation is the surface extraction and more precisely, the computation of the exact geometry of the facets selected by the graph-cut. A block of buildings are fully processed in approximately 30 seconds for the LOD1 model and 3 minutes for the LOD2 model. Less than 20 minutes (respectively 2 hours) are necessary to idealize a $1km^2$ dense urban area at LOD1 (resp. LOD2). The model complexity is particularly appealing as less than 200K facets are required to model buildings at LOD2.

4.4.3 Comparison

It is difficult to compare the reconstruction results of the two methods as they process different types of data. Thus, we will compare the results of each method with state-of-the-art methods developed for the same data type. Moreover, we do not have the GT for the resulting 3D model. The evaluation of the accuracy is obtained by evaluating the Hausdorff distance of the reconstruction with the original data. This evaluation is done after isolating a single building from the scene.

Both approaches correctly reconstruct buildings mainly composed of piecewise

	Semantic labeling	*Plane regularization*	*LOD1*	*LOD2*	*Trees and ground*	*Complexity*
Church 59K facets, Fig. 4.11	5s	1.5s	41s	198s	2s	190 facets
Building blocks 170K facets, Fig. 4.6	7s	1.1s	21s	137s	1.1s	456 facets
Paris, 7^{th} district 11M facets, Fig. 4.10	55s	95s	17 min	112 min	36s	175K facets

Table 4.1: Running time and model complexity from urban scenes of different sizes. The tests have been performed on an Intel Core i7 clocked at 2GHz. Note that time required for extracting LOD2 and LOD3 models can be considered as similar as superstructure reconstruction time is negligible. The complexity refers the number of facets of the building models at LOD2.

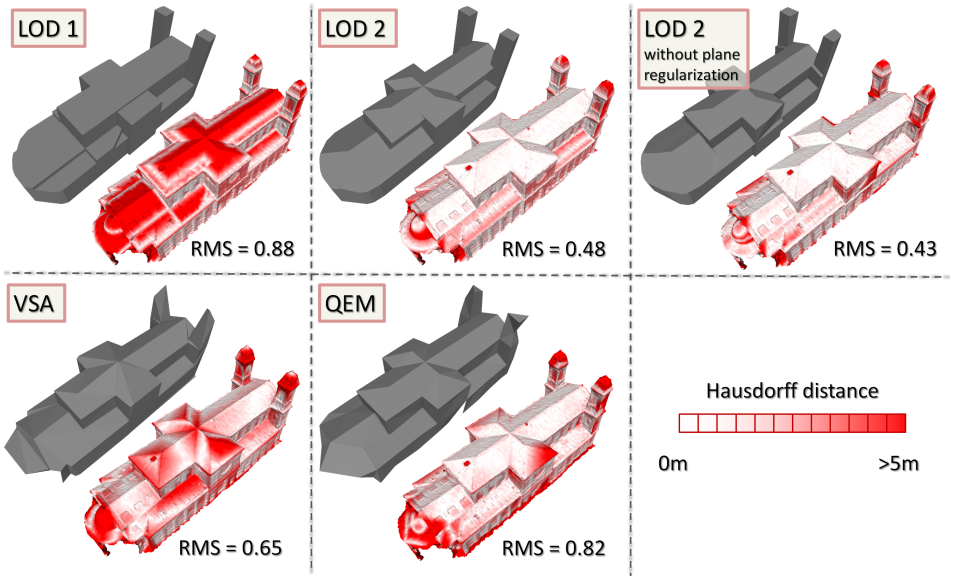


Figure 4.11: Geometric accuracy. Models produced by our system at different LODs are compared to standard surface approximation algorithms (QEM [Garland 1997] and VSA [Cohen-Steiner 2004]) by measuring the Hausdorff distance (scale from white to red) to the initial mesh. The complexity of the LOD2, QEM and VSA models are identical (approx. 190 facets) whereas LOD1 model is lower (175 facets). The Root Mean Square error (RMS) of the LOD1 model is higher than for QEM and VSA models, the roof being poorly described. The LOD2 model without plane regularization has a lower RMS than the LOD2 model with planar regularization but is visually less appealing and more time-consuming.

planar surfaces. We see the limits of our approaches in unusual cases of buildings with non-planar surfaces such as churches or domes. In these cases, small artifacts are visible on the resulting meshes but the overall reconstruction of the urban scene remains convincing.

Methodology: Concerning the methodologies of both mesh generation methods, we noticed that the method for Lidar data is more sensitive to missing data than its MVS counterpart. Indeed, with MVS data, a volumetric space partitioning is performed and cells are reconstructed when there is enough data to support this decision. This behavior is controlled by the smoothness term in the energy formu-

lation (β parameter, Eq. 4.2). The approach with MVS is thus, more resilient to missing data and noise than the approach for Lidar data.

Scalability: Both approaches are fully automatic and can handle large scenes. However, the approach for MVS takes longer to reconstruct an urban scene of same size compared to the approach for Lidar data. Indeed, the approach for MVS data uses a volumetric space partitioning that is costly and requires more computational resource than the other method. Note that during the experiments, both approaches were not limited by the memory usage and could successfully process urban area of large size (approximately 1km^2).

Reconstruction quality: With respect to the reconstruction quality, the approach for Lidar data generates buildings with reconstruction error (around 0.15 meter average) similar to those reported in the work of Zhou et al. [Zhou 2010] but with a better compactness. The approach for MVS data has a higher reconstruction error (around 0.46 meter average) but an even better compactness with a stronger regularization of the buildings. Compared to standard surface approximation methods ([Cohen-Steiner 2004] and [Garland 1997]), our approach for MVS generates more accurate meshes. Both our approaches with Lidar and MVS data generate accurate and convincing reconstruction compared to other state-of-the-art methods that employ similar data.

Flexibility: Finally, the approach for MVS data is flexible enough to arrive at several LODs: from LOD1 which are simple box-like building representations to LOD3 which are detailed representations with chimneys. Thus, this approach is more versatile than the approach for Lidar data which does not distinguish detailed elements of the buildings and fail to sufficiently take advantage of the semantics obtained after labeling the data.

Compared to each other, the approach with MVS data seems to be slightly better in terms of visual rendering, versatility and robustness.

4.5 Summary

4.5.1 Approach with Lidar data

We proposed an original method for reconstructing buildings from aerial lidar. Our method offers an accurate representation, and performs fast on wide range of data. The resulting mesh has its vertices labeled as *interior*, *boundary* or *dilated*, and is associated with the roof structures used to generated them. This semantic provides additional information that could be useful for texturing, urban planing, etc. Finally, our mesh-based model is extremely compact and accurate.

Contributions: Our work presents innovative solutions in the field by first, taking advantage of the mesh-based representations as well as the primitive-based approach, and second, providing both semantic information in the 3D-representation and a realistic reconstruction. We focused on reconstructing the buildings and did not consider the other urban elements which are excluded by the point cloud classification process. We demonstrated that our results use just a few vertices while remaining accurate. For generating the final mesh, we adopted a two-step strategy illustrated on Fig. 4.1. First, from the labeled point cloud (see Chapter 2.4.1), we extracted meaningful building structures from the points labeled as *building* (detailed in Section 4.2.1). Second, the generation of a compact mesh preserving the building structure is presented in Section 4.2.2. Experimental results are shown and discussed in Section 4.4.1.

Limitations: Our method has several limitations. The heuristic used for the quadric edge decimation stop criteria provides good results in most cases, but it may also remove too many edges. Furthermore, as we can see on Fig. 4.8, a sequence of small planar roofs is not always well suited for some non-planar roofs (*e.g* the church tower on result #2 and the spherical building on result #6).

Perspectives: In future work, it would be interesting to use a mesh decimation scheme that adapt the stop criteria to the type of building more accurately. Moreover, buildings with non-planar structures may not be well represented with our

current method and working on non-planar structure extraction would improve the results. Another interesting idea would be to substitute the regular XY-grid of our method by a quadtree in order to reduce the computation complexity.

4.5.2 Approach with MVS data

We presented an algorithm for idealizing urban scenes from dense semantized meshes generated by Multi-View Stereo system. Our approach used labeled data as *ground*, *vegetation*, *roof* and *facade* (see Chapter 2.4.2) before approximating their shapes by compact and geometrically structured models at different levels of detail. In addition to being efficient, the main advantage of our approach is its robustness to geometric and topological defects contained in the input meshes. Our plane regularization procedure and discrete formulation of the surface extraction problem constitute two technical contributions. Appealing performance in terms of computation time and scalability were demonstrated.

Contributions: To the best of our knowledge, the idea of inserting both semantical and structural information in a purely-geometric dense mesh is still unexplored for urban context. By considering a mesh as a digital image, mesh idealization would correspond to a semantic image segmentation with shape priors. Contrary to the existing city modeling methods, our algorithm provides a representation of buildings at various LODs so that the 3D-models can be used in different application contexts.

From a technical point of view, we proposed several contributions to the primitive-based surface reconstruction problem. Our aim is to arrive at high efficiency when processing massive data. Existing algorithms for detecting and regularizing geometric primitives as well as methods for extracting surfaces from planar-based partitions of the space provide good results in practice but they are not designed to tackle massive datasets under reasonable computation time. For instance, the number of planes detected from a $1km^2$ sized MVS mesh can easily exceed ten thousands in our experiments. Existing methods usually require as much as ten minutes to one hour to deal with just one hundred planes.

Limitations: Our algorithm is limited to modeling the buildings with piecewise planar components which becomes restrictive when dealing with atypical buildings with irregular shapes, such as the dome of the *Ecole militaire de Paris* in Fig. 4.10. A second notable limitation concerns the large-scale MVS meshes which are divided into several tiles for storage reasons: our algorithm processes the tiles independently without enforcing consistency between the neighboring tiles. Buildings located in between two tiles can thus contain geometric artifacts.

Perspectives: In the perspective, we plan to improve the accuracy of our models by considering geometric primitives that are not necessarily planar. We also envisage extending this approach in order to deal with more LODs for the modeling of buildings, in particular, in detecting facade components as doors or windows. This constitutes a challenging task as little geometric information is contained on facades.

From a technical point of view, the method processing MVS meshes uses a novel space partitioning. This space partition reports improved results by restraining the number of cells needed to be reconstructed to the strict minimum. However, the reconstruction of the selected cells remains costly since the computation of multiple plane-intersections are still needed. Indeed, each reconstruction relies on a high number of exact arithmetic operations which are time consuming. Using the method of Campen et al. [Campen 2010] will add inexact arithmetic to the system during the volume partitioning. This additional component should improve our approach in terms of performances.

Chapter 5

Conclusion and future work

5.1 Conclusion

This Ph.D thesis presented explored novel techniques in the urban reconstruction field using Lidar and MVS meshes for an automatic and accurate modeling of cities. For both types of data, a two-step pipeline was designed which (i) labels the input data into urban-related classes such as building, vegetation, ground, or small element and then (ii) automatically generates compact meshes of the urban scene. Top-down and bottom-up approaches were evaluated to reconstruct urban elements. The design of both pipelines was constrained by important considerations which are (1) automatic processing, (2) efficiency, (3) semantic labeling, (4) accuracy, and (5) sparse representation. These considerations were defined in details in Section 1.5.2.

The semantic labeling (3) is part of the first step of each pipeline. This important process was presented in Chapter 2 where we described labeling methods based on an energy minimization formalism for Lidar and MVS data. This labeling showed to be meaningful and to contribute to the scene understanding and reconstruction. We showed the benefits of using MVS meshes over aerial Lidar data for scene labeling since the former also describes the vertical structures of the scene. Thus, the labeling of MVS data is more complete and informative. Although the labeling method for Lidar data remains simple, the experiments have shown we obtained accurate results. We defined four simple geometric features computed for each point which are part of a novel energy formulation. The solution, which is the labeling that has the lowest energy, is quite stable and relies on few assumptions on the scene. Moreover, we can take advantage of the extra information brought by the Lidar technology such as

the number of echoes returned, the amplitude of the Lidar data, etc. to design an energy that is resilient to labeling errors. Concerning the MVS mesh labeling, we obtained an even more precise semantic labeling than with aerial Lidar data. The experiments proved the MVS labeling to be accurate and consistent while using less tuning parameters than needed for labeling Lidar data. However, the approach is composed of a post-processing step consisting in basic semantic rules which improves the quality of the results but makes it less general and flexible. Altogether, both labeling methods are unsupervised and fully automatic (2). They contribute as novel and simple but effective labeling approaches for urban scene understanding. To our best knowledge, we are the first to describe a new challenge in the field which is to insert both semantical information and geometric structures into MVS meshes of urban scenes.

The second step of the pipeline is presented in the subsequent two chapters - (Chapter 3 and Chapter 4) - where the reconstruction of the urban elements into meshes is detailed. Both described methods - one for each type of data - used labeled data and applied different strategies for each type of urban elements.

From aerial Lidar data (Section 4.2), we proposed two methods to mesh different type of urban elements. One method relies on a library of simple 3D templates and is capable, among other things, of detecting trees in the Lidar point cloud and generating parametric shapes, i.e meshes of trees (Chapter 3). The technique relies on a mathematical model which has the advantage of being general and robust. It is known to deliver convincing results as long as a proper modeling of the problem into an energy minimization formalism is given. While this mathematical model already existed, a major contribution in this thesis is the novel optimization scheme exploiting parallel techniques to converge faster. We showed improvements compared to previous optimization methods through various type of experiments and illustrated the potential of our method on problems with complexity never attended before. The second method - free of any a-priori on shapes - reconstructed more complex and unique structures which are common with buildings. This is realized by detecting planar patches inside the point clouds. It followed a coarse mesh reconstruction which is then completed by an edge collapse decimation (Chapter 4). While this

approach stays simple and straightforward, it gives good results in term of accuracy (4) and light-weight reconstruction (5) which was discussed during the experiments and comparisons with the work of Zhou et al. [Zhou 2010].

From MVS meshes (Section 4.3), a method for reconstructing all the elements of the scene was described which follows different modeling strategies for each type of elements, i.e. buildings, vegetation and ground (Chapter 4). The modeling of the buildings is flexible enough to generate multiple Level of Details (LOD) allowing a precise control on the idealization of the urban scene. This was achieved by using an novel discrete formulation of the space partitioning problem combined with a global regularization scheme for urban structures. Both discrete formulation and global regularization contributed in the field by heading towards a more precise urban scene idealization. Experiments on complex urban structures illustrated the robustness and scalability of our system. The reconstruction method has also been tested for accuracy (4) and light-weight reconstruction (5) by experiencing single building reconstruction and comparing the results obtained with other known reconstruction techniques and particularly the Variational Shape Approximation [Cohen-Steiner 2004].

Note that special care has been taken to design both pipelines to be fully automatic (1) and efficient (2). Both pipelines were validated by experiments on large and complex urban scenes.

5.2 Outlook

In spite of many contributions to automatic urban scene reconstruction, still numerous important problems remain unsolved.

Assessing the state-of-the-art, one can notice that many solutions rely on assumptions on the scene or on the data which limits their flexibility. Many works are dealing with very narrow-scoped problems whose proposed solutions can be hardly transferred to other problems or type of data. In addition, most of them use only a single type of data at the time while we have shown in this thesis each type of

data has unique advantages which are very useful for urban scene reconstruction. We believe there is potential in developing more general and flexible solutions which would use different types of data simultaneously.

In computer vision and machine learning, much effort have been made towards automatic learning techniques which can be applied for urban semantization and reconstruction. Two directions seem promising. First, we believe that a hybrid system mixing robust supervised procedural approaches with flexible generic mesh reconstruction is a direction for future researches. Second, impressive work on automatic processing of large collections of pictures from online web services have recently impacted the field. These two directions are surely just the precursors for new research projects which will probably lead to important contributions.

In geometry processing, a clear trend for structure-aware methods has been observed [Mitra 2013]. Much research effort aim at enriching 3D models with specific semantic and topological relationships which are useful in urban modeling. Regularization, symmetry detection, functional analysis, are but a few of many aspects which would contribute to a better idealization of urban scenes. This seems to be a promising direction for urban scene reconstruction.

In the remote sensing community, more and more techniques use data fusion approaches to benefit from the difference characteristics of the data. For example, researchers in satellite imagery obtained excellent results by using multi-spectral analysis which mixes various sources of data simultaneously. Similarly, the archaeological survey community already acknowledged the benefits of using simultaneously laser scanners, range imaging and sonar sensors. While only at its early stage of research, fusing various types of data and using them simultaneously seems a very promising technique and we expect it to become even more important in the near future.

Altogether, many works from various communities can contribute to the same goal of idealizing and reconstructing large urban scenes. By easily producing 3D models of cities with a better semantic, the field of research could evolve to the next interesting problem of a complete functional analysis of large urban scenes which,

we believe, will attract a wide audience in multiple domains and has a substantial potential in the area of academic research and industry.

Appendix A

Appendix

Population counting model

Let x denote a configuration of ellipses for which the center of mass of an ellipse is contained in the compact set K supporting the input image (see Fig. A.1). The energy follows the form specified by Eq. 3.5. The unitary data term $D(x_i)$ and the potential $V(x_i, x_j)$ are given by:

$$D(x_i) = \begin{cases} 1 - \frac{d(x_i)}{d_0} & \text{if } d(x_i) < d_0 \\ exp(\frac{d_0 - d(x_i)}{d_0}) - 1 & \text{otherwise} \end{cases} \tag{A.1}$$

$$V(x_i, x_j) = \beta \frac{A(x_i \cap x_j)}{\min(A(x_i), A(x_j))} \tag{A.2}$$

where

- $d(x_i)$ represents the Bhattacharyya distance between the radiometry inside and outside the object x_i:

$$d(x_i) = \frac{(m_{in} - m_{out})^2}{4(\sigma_{in}^2 + \sigma_{out}^2)} - \frac{1}{2} ln(\frac{2\sigma_{in}\sigma_{out}}{\sigma_{in}^2 + \sigma_{out}^2}) \tag{A.3}$$

  where m_{in} and σ_{in} (respectively m_{out} and σ_{out}) are the intensity mean and standard deviation in S_{in} (respectively in S_{out}).
- d_0 is a coefficient fixing the sensitivity of the object fitting. The higher the value of d_0, the more selective the object fitting. In particular, d_0 has to be high when the input images are corrupted by a significant amount of noise.
- $A(x_i)$ is the area of object x_i.
- β is a coefficient weighting the non-overlapping constraint with respect to the data term.

Note that a basic mathematical dilatation is used in practice to roughly extract the class of interest from the image of birds for creating a space-partitioning tree.

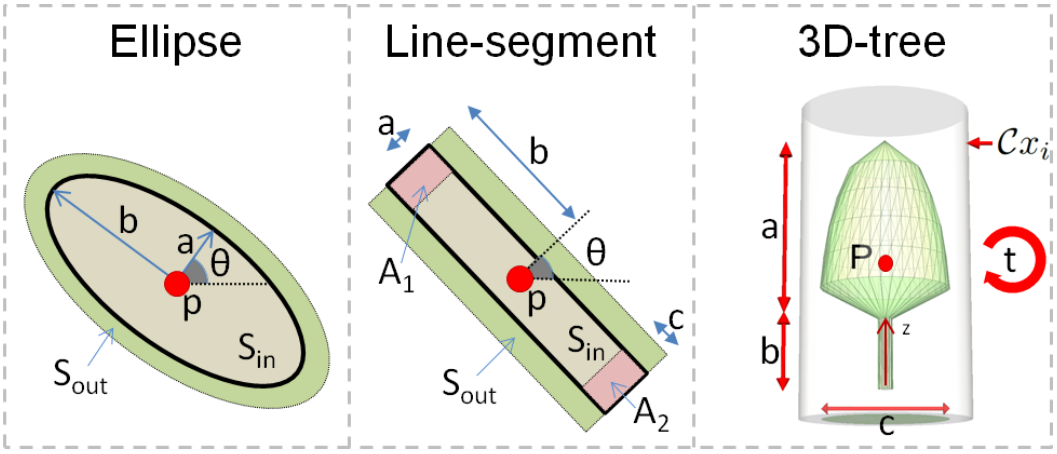


Figure A.1: Objects and their parameters for the various presented models. (left) Ellipses and (middle) line-segments are defined by a 2D-point $p \in K$ (center of mass of the object) and some marks. These additional parameters are the semi-major axis b, the semi-minor axis a, and the angle θ for an ellipse, and the semi-length b, the semi-width a, the orientation θ, and the anchor length c for a line-segment. The inside (respectively bordering) volume of the object is denoted by S_{in} (respectively S_{out}). The anchors are denoted by A_1 and A_2. (right) 3D-trees are defined by a 3D-point $p \in K$ (center of mass of the object), a type $t \in$ {conoidal, ellipsoidal, semi-ellipsoidal} illustrated on Fig. A.2, and 3 additional parameters which are the canopy height a, the trunk height b and the canopy diameter c. The cylindrical volume $\mathcal{C}x_i$ represents the attraction space of object x_i in which the input points are used to measure the quality of this object.

Line-network extraction model

A line-segment is defined by five parameters, including the 2D point corresponding to the center of mass of the object (Fig. A.1). Similarly to the population counting model detailed in Appendix A, the fitting quality with respect to the data is based on the Bhattacharyya distance: the unitary data term $D(x_i)$ of the energy is given by Eq. A.1. The potential $V(x_i, x_j)$ penalizes strong object overlaps (see Eq. A.2), but also takes into account a connection interaction in order to favor the

linking of the line-segments. The potential term is thus given by:

$$V(x_i, x_j) = \beta_1 \frac{A(x_i \cap x_j)}{\min(A(x_i), A(x_j))} + \mathbf{1}_{x_i \sim_{nc} x_j} \times \beta_2 f(x_i, x_j) \tag{A.4}$$

where

- β_1 and β_2 are two coefficients weighing respectively the non-overlapping and connection constraints with respect to the data term.
- $\sim_{nc}$ is the non-connection relationship between two objects. $x_i \sim_{nc} x_j$ if the anchor areas of x_i and x_j (see Fig. A.1) do not overlap.
- $\mathbf{1}_{condition}$ is the indicative function returning one when *condition* is valid, and zero otherwise.
- $f(x_i, x_j)$ is a symmetric function weighting the penalization of two non-connected objects x_i and x_j with respect to their average fitting quality. The function f is introduced to slightly relax the connection constraint when the two objects are of very good quality.

As for the bird counting problem, a basic mathematical dilatation has been used to roughly extract the class of interest from the aerial image shown on Fig. 3.14. Indeed the pixels corresponding to the class road in this image are relatively bright compared to the background. The segmented result is obviously not optimal, but sufficient to create an efficient space-partitioning tree.

Tree recognition model formulation

Let x represent a configuration of 3D-models of trees from a template library described in Fig. A.2. The center of mass p of a tree is contained in the compact set K supporting the 3D bounding box of the input point cloud (Fig. A.1). We denote by ∂x_i the surface of the object x_i, and by $\mathcal{C}x_i$ the cylindrical volume having a vertical axis passing through the center of mass of x_i, in which the input points are considered to measure the quality of x_i. The unitary data term $D(x_i)$ and the

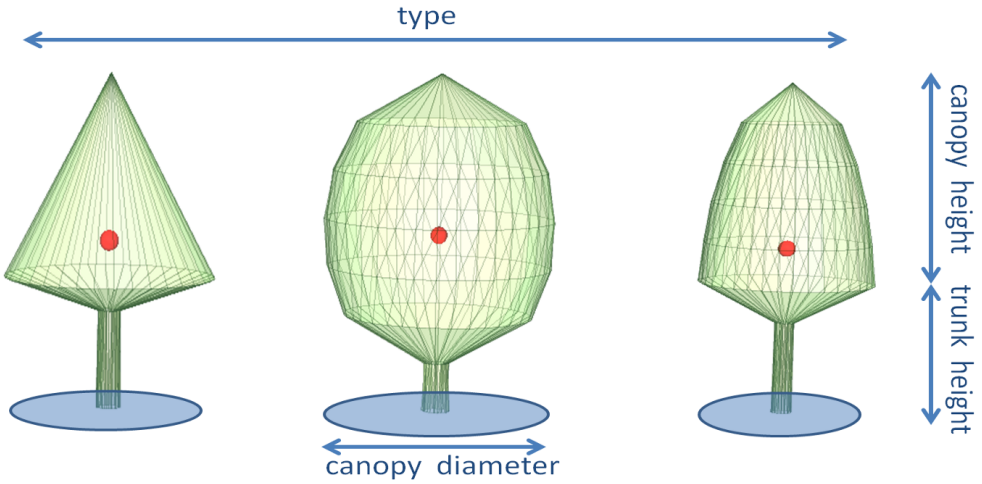


Figure A.2: Library of tree models - the objects are specified by a 3D point (center of mass illustrated by a red dot) and additional parameters (blue arrows) including the canopy type whose shape can be conoidal (*e.g.* pine or fir), ellipsoidal (*e.g.* poplar or tilia) or semi-ellipsoidal (*e.g.* oak or maple).

pairwise potential $V(x_i, x_j)$ are given by:

$$D(x_i) = \frac{1}{|\mathcal{C}x_i|} \prod_{p_c \in \mathcal{C}x_i} \gamma(d(p_c, \partial x_i)) \tag{A.5}$$

$$V(x_i, x_j) = \beta_1 V_{overlap}(x_i, x_j) + \beta_2 V_{competition}(x_i, x_j) \tag{A.6}$$

where

- $|\mathcal{C}x_i|$ is a coefficient normalizing the unitary data term with respect to the number of input points contained in $\mathcal{C}x_i$.
- $d(p_c, \partial x_i)$ is a distance measuring the coherence of the point p_c with respect to the object surface ∂x_i. d is not the traditional orthogonal distance from point to surface because, as real trees do not describe ellipsoidal/conoidal shapes, input points are not homogeneously distributed on the object surface. Here, d is defined as the combination of the planimetric distance, *i.e.* the projection in the plane of equation $z = 0$ of the Euclidean distance, and the altimetric variation such that points outside the object are more penalized than inside

points. Note that d is invariant by rotation around the Z-axis.

- $\gamma(.) \in [-1, 1]$ is a quality function which is strictly increasing.
- $V_{overlap}$ is the pairwise potential penalizing strong overlapping between two objects, and given by:

$$V_{overlap}(x_i, x_j) = \frac{A(x_i \cap x_j)}{\min(A(x_i), A(x_j))} \tag{A.7}$$

  where $A(x_i)$ is the area of the object x_i projected onto the plane of equation $z = 0$.

- $V_{competition}$ is the pairwise potential favoring a similar tree type t in a local neighborhood:

$$V_{competition}(x_i, x_j) = \mathbf{1}_{t_i \neq t_j} \tag{A.8}$$

  where $\mathbf{1}_{.}$ is the indicative function.

- β_1 and β_2 are two coefficients weighting respectively the non-overlapping constraint and the competition term with respect to the data term.

In order to roughly extract the class of interest from the point clouds, the scatter descriptor proposed by [Lafarge 2012] is used to identify the points which potentially correspond to trees.

Bibliography

[Agarwal 2009] S. Agarwal, N. Snavely, I. Simon, B. Curless, S.M. Seitz and R. Szeliski. *Building Rome in a day.* In ICCV, 2009. (Cited on pages 3 and 14.)

[Al-Durgham 2010] M. Al-Durgham, G. Fotopoulos and C. Glennie. *On the Accuracy of LiDAR Derived Digital Surface Models.* In Stelios P. Mertikas, editor, Gravity, Geoid and Earth Observation, volume 135 of *International Association of Geodesy Symposia*. Springer, 2010. (Cited on page 17.)

[Baddeley 1993] A.J. Baddeley and M.V. Lieshout. *Stochastic geometry models in high-level vision.* Journal of Applied Statistics, vol. 20, no. 5-6, 1993. (Cited on pages 47 and 55.)

[Becker 2009] S. Becker. *Generation and application of rules for quality dependent facade reconstruction.* Journal of Photogrammetry and Remote Sensing, vol. 64, no. 6, 2009. (Cited on page 17.)

[Ben-Ameur 2004] W. Ben-Ameur. *Computing the Initial Temperature of Simulated Annealing.* Computational Optimization and Applications, vol. 29, no. 3, 2004. (Cited on page 84.)

[Ben Hadj 2010] S. Ben Hadj, F. Chatelain, X. Descombes and J. Zerubia. *Parameter estimation for a marked point process within a framework of multidimensional shape extraction from remote sensing images.* In ISPRS Commission WG III/4, 2010. (Cited on page 84.)

[Benchmark 2013] Benchmark. Datasets, results and evaluation tools available at http://www-sop.inria.fr/members/Florent.Lafarge/benchmark/evaluation.html, 2013. (Cited on page 74.)

[Besag 1986] J.E. Besag. *On the Statistical Analysis of Dirty Pictures.* Journal of the Royal Statistical Society, vol. 48, no. 3, 1986. (Cited on page 81.)

[Botsch 2010] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez and B. Levy. AK Peters, 2010. (Cited on page 34.)

[Boykov 2001] Y. Boykov, O. Veksler and R. Zabih. *Fast Approximate Energy Minimization via Graph Cuts.* IEEE Trans. PAMI, vol. 23, no. 11, 2001. (Cited on pages 16, 38, 41 and 81.)

[Boykov 2004] Y. Boykov and V. Kolmogorov. *An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision.* IEEE Trans. PAMI, vol. 26, no. 9, 2004. (Cited on page 96.)

[Byrd 2010] J. Byrd, S. Jarvis and A. Bhalerao. *On the parallelisation of MCMC-based image processing.* In IEEE International Symposium on Parallel and Distributed Processing, 2010. (Cited on page 51.)

[Campen 2010] M. Campen and L. Kobbelt. *Exact and Robust (Self-) Intersections for Polygonal Meshes.* Computer Graphics Forum, vol. 29, no. 2, 2010. (Cited on page 109.)

[CGAL 2013] CGAL. Computational Geometry Algorithms Library, 2013. www.cgal.org. (Cited on pages 97 and 102.)

[Chai 2012] D. Chai, W. Forstner and M.Y. Yang. *Combine Markov Random Fields and Marked Point Processes to Extract Building from Remotely Sensed Images.* In ISPRS Commission ICWG III/VI, 2012. (Cited on page 49.)

[Chai 2013] D. Chai, W. Forstner and F. Lafarge. *Recovering line-networks in images by junction-point processes.* In CVPR, 2013. (Cited on page 49.)

[Chauve 2010] A.-L. Chauve, P. Labatut and J.-P. Pons. *Robust Piecewise-Planar 3D Reconstruction and Completion from Large-Scale Unstructured Point Data.* In CVPR, 2010. (Cited on page 94.)

[Cipolla 1999] R. Cipolla and D. Robertson. *3D models of architectural scenes from uncalibrated images and vanishing points.* In Image Analysis and Processing, 1999. (Cited on pages 3 and 13.)

[Cohen-Steiner 2004] D. Cohen-Steiner, P. Alliez and M. Desbrun. *Variational shape approximation.* In SIGGRAPH, 2004. (Cited on pages 20, 87, 103, 105, 106 and 113.)

[Collins 1996] R.T. Collins. *A Space-Sweep Approach to True Multi-Image Matching.* In CVPR, 1996. (Cited on page 15.)

[Cornelis 2008] N. Cornelis, B. Leibe, K. Cornelis and L. Van Gool. *3D Urban Scene Modeling Integrating Recognition and Reconstruction.* IJCV, vol. 78, no. 2-3, 2008. (Cited on page 14.)

[Debevec 1996] P.E. Debevec, C.J. Taylor and J. Malik. *Modeling and rendering architecture from photographs: a hybrid geometry- and image-based approach.* In SIGGRAPH, 1996. (Cited on page 13.)

[Denis 2008] P. Denis, J.H. Elder and F.J. Estrada. *Efficient Edge-Based Methods for Estimating Manhattan Frames in Urban Imagery.* In ECCV, 2008. (Cited on page 20.)

[Descombes 2009] X. Descombes, R. Minlos and E. Zhizhina. *Object Extraction Using a Stochastic Birth-and-Death Dynamics in Continuum.* Journal of Mathematical Imaging and Vision, vol. 33, no. 3, 2009. (Cited on pages 20, 48, 51, 65, 66 and 68.)

[Descombes 2011] X. Descombes. Stochastic geometry for image analysis. Wiley-ISTE, 2011. (Cited on page 55.)

[Dick 2004] A.R. Dick, P.H.S. Torr and R. Cipolla. *Modelling and Interpretation of Architecture from Several Images.* IJCV, vol. 60, no. 2, 2004. (Cited on pages 3 and 14.)

[du Valat 2013] Tour du Valat. `http://www.tourduvalat.org/`, 2013. (Cited on pages 61 and 66.)

[Du 2011] H. Du, P. Henry, X. Ren, M. Cheng, D.B. Goldman, S.M. Seitz and D. Fox. *Interactive 3D modeling of indoor environments with a consumer depth camera.* In UbiComp, 2011. (Cited on page 16.)

[Earl 2005] D. Earl and M. Deem. *Parallel tempering: Theory, applications, and new perspectives.* Physical Chemistry Chemical Physics, vol. 23, no. 7, 2005. (Cited on pages 51, 66, 67 and 68.)

[Fruh 2004] C. Fruh and A. Zakhor. *An Automated Method for Large-Scale, Ground-Based City Model Acquisition.* IJCV, vol. 60, no. 1, 2004. (Cited on pages 10 and 18.)

[Furukawa 2009] Y. Furukawa, B. Curless, S.M. Seitz and R. Szeliski. *Manhattan-world stereo.* In CVPR, 2009. (Cited on pages 14, 20 and 94.)

[Furukawa 2010] Y. Furukawa, B. Curless, S.M. Seitz and R. Szeliski. *Towards Internet-scale multi-view stereo.* In CVPR, 2010. (Cited on pages 3 and 4.)

[Garland 1997] M. Garland and P. Heckbert. *Surface Simplification Using Quadric Error Metrics.* In SIGGRAPH, 1997. (Cited on pages 90, 103, 105 and 106.)

[Ge 2009] W. Ge and R. Collins. *Marked point processes for crowd counting.* In CVPR, 2009. (Cited on pages 48, 50 and 82.)

[Geoinformatics 2007] Geoinformatics. *Geoinformatics webpages .* `http://www.geoinformatics.com/blog/online-articles/geokosmos`, 2007. [Online; accessed 19-July-2013]. (Cited on page 12.)

[Goesele 2006] M. Goesele, B. Curless and S.M. Seitz. *Multi-View Stereo Revisited.* In CVPR, 2006. (Cited on page 4.)

[Goesele 2007] M. Goesele, N. Snavely, B. Curless, H. Hoppe and S.M. Seitz. *Multi-View Stereo for Community Photo Collections.* In ICCV, 2007. (Cited on pages 3 and 14.)

[Gonzalez 2011] J. Gonzalez, Y. Low, A. Gretton and C. Guestrin. *Parallel Gibbs Sampling: From Colored Fields to Thin Junction Trees.* Journal of Machine Learning Research, vol. 15, 2011. (Cited on page 51.)

[Google 2013] Earth Google. `http://earth.google.com/`, 2013. (Cited on pages 1, 26, 29, 34, 73, 79 and 103.)

[Green 1995] P.J. Green. *Reversible Jump Markov Chains Monte Carlo computation and Bayesian model determination.* Biometrika, vol. 82, no. 4, 1995. (Cited on pages 48, 50, 54, 55, 66, 67 and 68.)

[Grenander 1994] U. Grenander and M.I. Miller. *Representations of Knowledge in Complex Systems.* Journal of the Royal Statistical Society, vol. 56, no. 4, 1994. (Cited on page 50.)

[Haala 2008] N. Haala, M. Peter, J. Kremer and G. Hunter. *Mobile LiDAR mapping for 3D point cloud collection in urban areas: A Performance Test.* The

International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 37, pages 1119–1127, 2008. (Cited on pages 10 and 11.)

[Habbecke 2012] M. Habbecke and L. Kobbelt. *Linear analysis of Nonlinear constraints for interactive geometric modeling.* In Eurographics, 2012. (Cited on page 91.)

[Han 2004] F. Han, Z.W. Tu and S.C. Zhu. *Range Image Segmentation by an Effective Jump-Diffusion Method.* IEEE Trans. PAMI, vol. 26, no. 9, 2004. (Cited on page 50.)

[Harkness 2000] M. Harkness and P. Green. *Parallel chains, delayed rejection and reversible jump MCMC for object recognition.* In BMVC, 2000. (Cited on page 51.)

[Hastings 1970] W.K. Hastings. *Monte Carlo sampling using Markov chains and their applications.* Biometrika, vol. 57, no. 1, 1970. (Cited on page 48.)

[Hengel 2006] A.V.D. Hengel, A. Dick, T. Thormahlen, B. Ward and P.H.S. Torr. *Building models of regular scenes from structure and motion.* In BMVC, 2006. (Cited on page 14.)

[Hiep 2009] V.H. Hiep, R. Keriven, P. Labatut and J.-P. Pons. *Towards high-resolution large-scale multi-view stereo.* In CVPR, 2009. (Cited on pages 4, 5, 7, 15, 22 and 102.)

[Hornung 2006] A. Hornung and L. Kobbelt. *Hierarchical Volumetric Multi-view Stereo Reconstruction of Manifold Surfaces based on Dual Graph Embedding.* In CVPR, 2006. (Cited on page 4.)

[Izadi 2011] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison and A. Fitzgibbon. *KinectFusion: real-time 3D reconstruction and interaction using a moving depth camera.* In User interface software and technology, 2011. (Cited on page 12.)

[Jancosek 2011] M. Jancosek and T. Pajdla. *Multi-view reconstruction preserving weakly-supported surfaces.* In CVPR, 2011. (Cited on pages 3 and 15.)

[Jiang 2009] N. Jiang, P. Tan and L. Cheong. *Symmetric architecture modeling with a single image.* TOG, vol. 28, no. 5, 2009. (Cited on page 13.)

[Kada 2009] M. Kada and L. McKinley. *3D building reconstruction from lidar based on a cell decomposition approach.* The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science, vol. 38, no. 3/W4, 2009. (Cited on page 16.)

[Kalogerakis 2010] E. Kalogerakis, A. Hertzmann and K. Singh. *Learning 3D Mesh Segmentation and Labeling.* In SIGGRAPH, 2010. (Cited on page 33.)

[Kang 2001] S.B. Kang, R. Szeliski and J. Chai. *Handling occlusions in dense multi-view stereo.* In CVPR, 2001. (Cited on page 15.)

[Kazhdan 2006] M. Kazhdan, M. Bolitho and H. Hoppe. *Poisson surface reconstruction.* In SGP, 2006. (Cited on page 4.)

[Kim 2011] E. Kim and G. Medioni. *Urban scene understanding from aerial and ground LIDAR data.* Machine Vision and Applications, vol. 22, no. 4, 2011. (Cited on page 8.)

[Lacoste 2005] C. Lacoste, X. Descombes and J. Zerubia. *Point Processes for Unsupervised Line Network Extraction in Remote Sensing.* IEEE Trans. PAMI, vol. 27, no. 10, 2005. (Cited on pages 49 and 75.)

[Lafarge 2010a] F. Lafarge, X. Descombes, J. Zerubia and M. Pierrot-Deseilligny. *Structural approach for building reconstruction from a single DSM.* IEEE Trans. PAMI, vol. 32, no. 1, 2010. (Cited on pages 16, 49, 50 and 75.)

[Lafarge 2010b] F. Lafarge, G. Gimel'farb and X. Descombes. *Geometric Feature Extraction by a Multi-Marked Point Process.* IEEE Trans. PAMI, vol. 32, no. 9, 2010. (Cited on pages 83 and 86.)

[Lafarge 2010c] F. Lafarge, R. Keriven and M. Bredif. *Insertion of 3D-Primitives in Mesh-Based Representations: Towards Compact Models Preserving the Details.* IEEE Trans. on IP, vol. 19, no. 7, 2010. (Cited on page 87.)

[Lafarge 2012] F. Lafarge and C. Mallet. *Creating large-scale city models from 3D-point clouds: a robust approach with hybrid representation.* IJCV, vol. 99, no. 1, 2012. (Cited on pages 16, 17, 28, 86 and 121.)

[Lato 2010] M.J. Lato, M.S. Diederichs and D.J. Hutchinson. *Bias Correction for View-limited Lidar Scanning of Rock Outcrops for Structural Characterization.* Rock Mechanics and Rock Engineering, vol. 43, no. 5, 2010. (Cited on page 9.)

[Lehmussola 2007] A. Lehmussola, P. Ruusuvuori, J. Selinummi, H. Huttunen and O. Yli-Harja. *Computational framework for simulating fluorescence microscope images with cell populations.* IEEE Trans. on MI, vol. 26, no. 7, 2007. (Cited on pages 70, 71 and 72.)

[Lempitsky 2010] V. Lempitsky and A. Zisserman. *Learning To Count Objects in Images.* In NIPS, 2010. (Cited on pages 67, 71 and 72.)

[Li 2001] S.Z. Li. Markov random field modeling in image analysis. Springer, 2001. (Cited on page 81.)

[Li 2011] Y. Li, X. Wu, Y. Chrysathou, A. Sharf, D. Cohen-Or and N.J. Mitra. *GlobFit: Consistently Fitting Primitives by Discovering Global Relations.* In SIGGRAPH, 2011. (Cited on pages 18, 87, 91 and 94.)

[Lin 2013] H. Lin, J. Gao, Y. Zhou, G. Lu, M. Ye, C. Zhang, L. Liu and R. Yang. *Semantic Decomposition and Reconstruction of Residential Scenes from LiDAR Data.* In SIGGRAPH, 2013. (Cited on page 86.)

[Liu 2001] J. Liu. Monte Carlo strategies in scientific computing. Springer, 2001. (Cited on page 48.)

[Liu 2010] T. Liu, M. Carlberg, G. Chen, J. Chen, J. Kua and A. Zakhor. *Indoor localization and visualization using a human-operated backpack system.* In Indoor Positioning and Indoor Navigation, 2010. (Cited on pages 10 and 12.)

[Maas 2006] J. Maas, R.A. Verheij, P.P. Groenewegen, S. de Vries and P. Spreeuwenberg. *Green space, urbanity, and health: how strong is the relation?* Journal of Epidemiology and Community Health, vol. 60, no. 7, 2006. (Cited on page 27.)

[Mallet 2010] C. Mallet, F. Lafarge, M. Roux, U. Soergel, F. Bretar and C. Heipke. *A Marked Point Process for Modeling Lidar Waveforms.* IEEE Trans. on IP, vol. 19, no. 12, 2010. (Cited on pages 9, 49 and 50.)

[Mastin 2009] A. Mastin, J. Kepner and J. Fisher. *Automatic registration of LIDAR and optical images of urban scenes.* In CVPR, 2009. (Cited on page 18.)

[Microsoft 2013] Virtual Earth Microsoft. http://www.microsoft.com/maps/, 2013. (Cited on page 1.)

[Mitra 2013] N.J. Mitra, M. Wand, H. Zhang, D. Cohen-Or and M. Bokeloh. *Structure-Aware Shape Processing.* In Eurographics, 2013. (Cited on page 114.)

[Musialski 2013] P. Musialski, P. Wonka, D.G. Aliaga, M. Wimmer, L. van Gool and W. Purgathofer. *A Survey of Urban Reconstruction.* Computer Graphics Forum, 2013. (Cited on pages 13, 85, 86 and 102.)

[Nan 2010] L. Nan, A. Sharf, H. Zhang, D. Cohen-Or and B. Chen. *SmartBoxes for Interactive Urban Reconstruction.* In SIGGRAPH, 2010. (Cited on page 15.)

[NCALM 2013] NCALM. *National Center for Airborne Laser Mapping.* http://www.ncalm.cive.uh.edu/, 2013. [Online; accessed 19-July-2013]. (Cited on page 10.)

[Nguyen 2010] H.-G. Nguyen, R. Fablet and J.M. Bouchet. *Spatial statistics of visual keypoints for texture recognition.* In ECCV, 2010. (Cited on page 49.)

[NOAA 2013] NOAA. *NOAA webpage .* http://csc.noaa.gov/digitalcoast/dataregistry/, 2013. [Online; accessed 19-July-2013]. (Cited on page 10.)

[Oesau 2013] S. Oesau, F. Lafarge and P. Alliez. *Indoor Scene Reconstruction using Primitive-driven Space Partitioning and Graph-cut.* In Eurographics Workshop on Urban Data Modelling and Visualisation, 2013. (Cited on page 12.)

[Oliensis 2000] J. Oliensis. *A Critique of Structure-from-Motion Algorithms.* Computer Vision and Image Understanding, vol. 80, no. 2, 2000. (Cited on page 3.)

[Opentopography 2013] Opentopography. *Opentopography webpage .* http://www.opentopography.org/, 2013. [Online; accessed 19-July-2013]. (Cited on pages 9 and 10.)

[Ortner 2008] M. Ortner, X. Descombes and J. Zerubia. *A Marked Point Process of Rectangles and Segments for Automatic Analysis of Digital Elevation Models.* IEEE Trans. PAMI, vol. 30, no. 1, 2008. (Cited on pages 49, 50 and 83.)

[Park 2006] J. Park, I. Lee, Y. Choi and Y.J. Lee. *Automatic extraction of large complex buildings using lidar data and digital maps.* The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Science, 2006. (Cited on page 16.)

[Poullis 2009] C. Poullis and S. You. *Automatic reconstruction of cities from remote sensor data.* In CVPR, 2009. (Cited on pages 16 and 86.)

[Pu 2009] S. Pu and G. Vosselman. *Knowledge based reconstruction of building models from terrestrial laser scanning data.* Journal of Photogrammetry and Remote Sensing, vol. 64, no. 6, 2009. (Cited on page 17.)

[Rochery 2006] M. Rochery, I. Jermyn and J. Zerubia. *Higher Order Active Contours.* IJCV, vol. 69, no. 1, 2006. (Cited on page 75.)

[Rottensteiner 2005] F. Rottensteiner, J. Trinder, S. Clode and K. Kubik. *Automated delineation of roof planes from lidar data.* International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, vol. 36, no. 3/W19, 2005. (Cited on page 16.)

[Rottensteiner 2012] F. Rottensteiner, G. Sohn, J. Jung, M. Gerke, C. Baillard, S. Benitez and U. Breitkopf. *The ISPRS Benchmark On Urban Object Classification And 3d Building Reconstruction.* Journal of Photogrammetry and Remote Sensing, vol. I-3, no. 2012, 2012. (Cited on page 19.)

[Sack 2000] J.R. Sack and J. Urrutia. Handbook of computational geometry. Elsevier, 2000. (Cited on pages 88 and 89.)

[Salamon 2002] P. Salamon, P. Sibani and R. Frost. Facts, conjectures, and improvements for simulated annealing. SIAM Monographs on Mathematical Modeling and Computation, 2002. (Cited on page 55.)

[Satari 2012] M. Satari. *Recognition of Dormers from lidar data using support vector machine.* In IGARSS, 2012. (Cited on page 28.)

[Schnabel 2007] R. Schnabel, R. Wahl and R. Klein. *Efficient RANSAC for Point-Cloud Shape Detection.* Computer Graphics Forum, vol. 26, no. 2, 2007. (Cited on page 17.)

[Seitz 2006] S.M. Seitz, B. Curless, J. Diebel, D. Scharstein and R. Szeliski. *A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms.* In CVPR, 2006. (Cited on pages 4, 15 and 87.)

[Sinha 2008] S.N. Sinha, D. Steedly, R. Szeliski, M. Agrawala and M. Pollefeys. *Interactive 3D Architectural Modeling from Unordered Photo Collections.* SIGGRAPH Asia, 2008. (Cited on pages 3 and 14.)

[Srivastava 2002] A. Srivastava, U. Grenander, G. Jensen and M. Miller. *Jump-Diffusion Markov processes on orthogonal groups for object pose estimation.* Journal of Statistical Planning and Inference, vol. 103, no. 1-2, 2002. (Cited on page 50.)

[Stoica 2007] R.S. Stoica, V. Martinez and E. Saar. *A three dimensional object point process for detection of cosmic filaments.* Journal of the Royal Statistical Society, vol. 56, no. 4, 2007. (Cited on page 49.)

[Strasdat 2011] H. Strasdat, A.J. Davison, J.M.M. Montiel and K. Konolige. *Double window optimisation for constant time visual SLAM.* In ICCV, 2011. (Cited on page 16.)

[Sun 2007] K. Sun, N. Sang and T. Zhang. *Marked point process for vascular tree extraction on angiogram.* In EMMCVPR, 2007. (Cited on pages 49 and 50.)

[Szeliski 2008] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen and C. Rother. *Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors.* IEEE Trans. PAMI, vol. 30, no. 6, 2008. (Cited on pages 54 and 79.)

[Toshev 2010] A. Toshev, P. Mordohai and B. Taskar. *Detecting and parsing architecture at city scale from range data.* In CVPR, 2010. (Cited on page 86.)

[Tu 2002] Z. Tu and S.C. Zhu. *Image Segmentation by Data-Driven Markov Chain Monte Carlo.* IEEE Trans. PAMI, vol. 24, no. 5, 2002. (Cited on pages 51, 60, 66, 67, 68 and 82.)

[UN 2012] UN. *World Urbanization Prospects: The 2011 Revision Highlights (New York: United Nations Department of Economic and Social Affairs Population Division).* http://esa.un.org/unup/pdf/WUP2011_Highlights.pdf, 2012. [Online; accessed 19-July-2013]. (Cited on page 1.)

[UNAVCO 2008] UNAVCO. *UNAVCO TLS instrument .* http://facility.unavco.org/project_support/tls/tls.html, 2008. [Online; accessed 19-July-2013]. (Cited on page 11.)

[USGS 2013] USGS. *USGS webpage .* http://seamless.usgs.gov/, 2013. [Online; accessed 19-July-2013]. (Cited on page 10.)

[Utasi 2011] A. Utasi and C. Benedek. *A 3-D Marked Point Process Model for Multi-View People Detection.* In CVPR, 2011. (Cited on pages 49, 50 and 51.)

[van Lieshout 2008] M.N.M. van Lieshout. *Depth Map Calculation for a Variable Number of Moving Objects Using Markov Sequential Object Processes.* IEEE Trans. PAMI, vol. 30, no. 7, 2008. (Cited on page 49.)

[Vanegas 2010a] C. Vanegas, D. Aliaga and B. Benes. *Building Reconstruction using Manhattan-World Grammars.* In CVPR, 2010. (Cited on page 20.)

[Vanegas 2010b] C. Vanegas, D. Aliaga, P. Wonka, P. Muller, P. Waddell and B. Watson. *Modeling the Appearance and Behavior of Urban Spaces.* In Eurographics STAR, 2010. (Cited on pages 85 and 86.)

[Varanelli 1999] J.M. Varanelli and J.P. Cohoon. *A fast method for generalized starting temperature determination in homogeneous two-stage simulated annealing systems.* Computers & Operations Research, vol. 26, no. 5, 1999. (Cited on page 84.)

[Verdie 2011] Y. Verdie, F. Lafarge and J. Zerubia. *Generating compact meshes under planar constraints: an automatic approach for modeling buildings from aerial LiDAR.* In ICIP, 2011. (Cited on pages 23 and 24.)

[Verdie 2012] Y. Verdie and F. Lafarge. *Efficient Monte Carlo sampler for detecting parametric objects in large scenes.* In ECCV, 2012. (Cited on page 23.)

[Verdie 2013] Y. Verdie and F. Lafarge. *Detecting parametric objects in large scenes by Monte Carlo sampling.* IJCV, 2013. (Cited on page 23.)

[Verma 2006] V. Verma, R. Kumar and S. Hsu. *3D Building Detection and Modeling from Aerial LIDAR Data.* In CVPR, 2006. (Cited on page 16.)

[Wahl 2008] R. Wahl, R. Schnabel and R. Klein. *From detailed digital surface models to city models using constrained simplification.* Photogrammetrie, Fernerkundung, Geoinformation, 2008. (Cited on page 16.)

[Wang 2011] R. Wang. Towards urban 3D modeling using mobile LiDAR and images. McGill University Libraries, 2011. (Cited on pages 8 and 18.)

[Weiss 2001] Y. Weiss and W.T. Freeman. *On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs.* IEEE Trans. on Information Theory, vol. 47, no. 2, 2001. (Cited on page 81.)

[Wendel 2001] H.E.W. Wendel. *An Examination of the Impacts of Urbanization on Green Space Access and Water Resources: A Developed and Developing World Perspective.* PhD thesis, USF University South California, 2001. (Cited on page 27.)

[Zaharescu 2007] A. Zaharescu, E. Boyer and R. Horaud. *TransforMesh: a topology-adaptive mesh-based approach to surface evolution.* In ACCV, 2007. (Cited on page 4.)

[Zebedin 2008] L. Zebedin, J. Bauer, K. Karner and H. Bischof. *Fusion of Feature- and Area-Based Information for Urban Buildings Modeling from Aerial Imagery.* In ECCV, 2008. (Cited on pages 18 and 86.)

[Zhou 2008] Q.-Y. Zhou and U. Neumann. *Fast and extensible building modeling from airborne LiDAR data.* In Advances in geographic information systems, 2008. (Cited on page 16.)

[Zhou 2010] Q.-Y. Zhou and U. Neumann. *2.5D Dual Contouring: A Robust Approach to Creating Building Models from Aerial LiDAR Point Clouds.* In ECCV, 2010. (Cited on pages 20, 101, 102, 106 and 113.)

[Zhou 2012] Q.-Y. Zhou and U. Neumann. *2.5D Building Modeling by Discovering Global Regularities.* In CVPR, 2012. (Cited on pages 16, 86, 91 and 94.)

[Zhou 2013] Q.Y. Zhou and V Koltun. *Dense Scene Reconstruction with Points of Interest.* In SIGGRAPH, 2013. (Cited on page 12.)

[Zhu 2005] S.C. Zhu, C.E. Guo, Y.Z. Wang and Z.J. Xu. *What are Textons?* IJCV, 2005. (Cited on page 49.)