Robust Piecewise-Planar 3D Reconstruction and Completion from Large-Scale Unstructured Point Data

Anne-Laure Chauve Patrick Labatut Jean-Philippe Pons
IMAGINE, ENPC/CSTB/LIGM, Université Paris-Est, France
http://imagine.enpc.fr

Abstract

In this paper, we present a novel method, the first to date to our knowledge, which is capable of directly and automatically producing a concise and idealized 3D representation from unstructured point data of complex cluttered real-world scenes, with a high level of noise and a significant proportion of outliers, such as those obtained from passive stereo. Our algorithm can digest millions of input points into an optimized lightweight watertight polygonal mesh free of self-intersection, that preserves the structural components of the scene at a user-defined scale, and completes missing scene parts in a plausible manner. To achieve this, our algorithm incorporates priors on urban and architectural scenes, notably the prevalence of vertical structures and orthogonal intersections. A major contribution of our work is an adaptive decomposition of 3D space induced by planar primitives, namely a polyhedral cell complex. We experimentally validate our approach on several challenging noisy point clouds of urban and architectural scenes.

1. Introduction

1.1. Motivation

3D models of urban and architectural scenes are of increasing use in many applications: augmented reality, navigation, urban planning, physical simulations for environmental impact assessment, etc. Creating such models from range scanning and multi-view imagery constitute an interesting alternative to computer-aided design, both in terms of efficiency and cost. However, the scalability of these acquisition processes is still limited by their need for significant human assistance: purely-automatic 3D modeling from range or stereo data is still far inferior to its interactive counterpart as regards reliability, accuracy and compactness.

Whereas a large number of recent works have focused on the first two issues [24, 27], our paper tackles the third one. Indeed, automatic 3D modeling algorithms typically produce overly complex meshes which are cumbersome for storage, indexing, transmission and rendering. This problem is usually circumvented by a posteriori simplification. However, the latter may easily lose characteristic geometric features and worsen reconstruction defects.

This makes combined 3D reconstruction and idealization very desirable. Our work tackles the case of unstructured 3D points as input data, which encompasses merged range scans and stereovision depth maps, as well as several recent multi-view stereo techniques (*e.g.* [4, 12, 18]) explicitly based on point clouds.

1.2. Related work

The extensive work on segmentation of range images or height fields [5, 13, 32] and its application to automatic 2.5D city modeling [21, 33], and on reconstruction of piecewise-planar depth maps [1, 11, 20, 26] is not relevant to our problem due to its restrictive 2.5D assumption.

Segmentation of 3D point clouds into geometric primitives ([23] and references therein) only brings a partial solution, as it does not address the reconstruction of a well-behaved 3D surface.

A few recent works simultaneously address primitives detection and surface reconstruction [6, 14, 15, 17, 22] by leveraging a vast literature in computer-aided design. However, these methods only handle laser scans (or synthetic point clouds) of mechanical parts and simple architectural scenes, with a low level of noise and a small proportion of outliers. The tested scenes typically contain a few dozens of geometric primitives and never exceed 200.

1.3. Principles of our approach

Basically, our approach constructs a set of geometric primitives which faithfully approximates input data, then adequately assembles them into a well-behaved surface. To that extent, our work is related to [6, 14, 15, 17, 22]. But our instantiation of this general sketch obeys specific principles which allow to significantly push the state-of-the-art.

Order of approximation. A zero-order approximation is not sufficient in our context: slopes play an important semantic and physical role in architectural scenes, and surface normals have a major perceptual impact in rendering. Hence we maintain a first-order approximation throughout our approach. In particular, this involves estimating tangent planes to the input point cloud.

Degree of geometric primitives. Our approach uses planar primitives only. In contrast, some authors [14, 17, 23] have proposed to include higher-degree primitives such as spheres, cylinders, cones and tori. Our motivation is threefold. First, man-made environments are essentially piecewise-planar. Second, there is no loss of generality in restricting primitives to planes, since curved objects admit piecewise-linear approximations, at the expense of a higher number of primitives. Third, polygonal meshes are a widely established standard in computer vision, mesh processing [2] and computer graphics, to the point that the aforementioned higher-degree techniques [14, 17, 23] end up extracting a polygonal mesh as output. Our method *directly* computes an optimized polyhedral reconstruction.

Scale of analysis. Our 3D reconstruction algorithm aims at preserving structural parts of the scene. But, as extensively studied by the scale-space theory [19], the very notion of structure cannot be dissociated from the notion of scale. For instance, in architectural scenes, roof tiles and window moldings may be alternatively regarded as important structures or negligible perturbations depending on the scale of interest. Hence our whole approach is guided by a tolerance distance σ and a tolerance angle θ. The higher these scale parameters, the fewer primitives in the approximation and the more idealized the output reconstruction. Note that the input sampling density, accuracy and level of noise induce a lower bound on the scale of analysis.

Topological guarantees. Our approach is based on a labeling of a decomposition of 3D space into empty or occupied regions. Thus it has in common with volumetric graph cuts (*e.g.* [18, 25]) and implicit methods for surface reconstruction (*e.g.* [16]) the ability to output watertight surfaces free of self-intersection. In contrast, most existing solutions to our problem [6, 14, 15] stitch primitives with local decisions based on proximity. Hence they are unable to fill gaps in input data and may easily produce invalid topology in ambiguous cases.

Visibility-consistency. In the vein of [8, 18], our approach maximizes the consistency of the surface with visibility information provided by the acquisition process: additionally to positions and tangent planes, lines-of-sight of inputs point are incorporated in a global optimization.

Prior-based completion. Due to the imperfection of the acquisition process, some important structural parts of the scene may appear only partially in the input data, or even be completely absent. For example, façades are frequently missing in aerial data, due to narrow streets and/or insufficiently oblique views, while window ledges, top of steps as well as sides of various protrusions are often occluded in ground data. Such incomplete data makes much harder the task of obtaining a faithful watertight reconstruction.

While partially missing structures can be recovered by prolongation of detected geometric primitives, as in [17, 22], a single absent structure, for example a small indentation between two close parallel planes, can jeopardize the entire reconstruction. This inevitably leads to augment the set of primitives using prior knowledge about the scene.

Here, we introduce such prior knowledge in the form of *ghost primitives*: they consist of hypothesized planar primitives that ensure the expected continuation of detected primitives and enforce the prevalence of vertical structures and orthogonal intersections in urban and architectural scenes.

1.4. Contributions of our approach

To our knowledge, our algorithm is the first to date which is capable of directly and automatically producing a concise and idealized 3D representation from unstructured point data of complex cluttered real-world scenes, with a high level of noise and a significant proportion of outliers, such as those obtained from passive stereo.

Our algorithm can digest millions of input points into an optimized lightweight watertight polygonal mesh free of self-intersection, that preserves the structural components of the scene at the user-defined scale, and completes missing scene parts in a plausible manner.

A major contribution of our work is an adaptive decomposition of 3D space induced by planar primitives, namely a polyhedral cell complex. It can be regarded as an extension of 3D arrangements of planes [10] which have only appeared once [28] in image processing, for automatic building reconstruction, in elementary cases of less than ten planes. The space decompositions used in the two works closest to ours also face serious limitations. [22] is based on a regular cubic grid, the computational and memory cost of which quickly becomes prohibitive: it has been be applied only to mechanical parts with tight bounding boxes and less than two hundred primitives. As for the generalized binary space partition proposed in [17], it is intractable: it must be approximated by a dense 3D Delaunay triangulation which loses the desirable compactness of the representation.

In contrast, we demonstrate the feasibility of polyhedral cell complexes of thousands of planar primitives. This constitutes an alternative in many applications to other widespread non-uniform space decompositions, such as octrees (*e.g.* [16]) or 3D triangulations (*e.g.* [18, 25]).

The rest of this paper is organized as follows. After describing in detail the different steps of our algorithm in Section 2, we discuss implementation aspects and we experimentally validate our approach on several challenging noisy point clouds of urban and architectural scenes in Section 3.

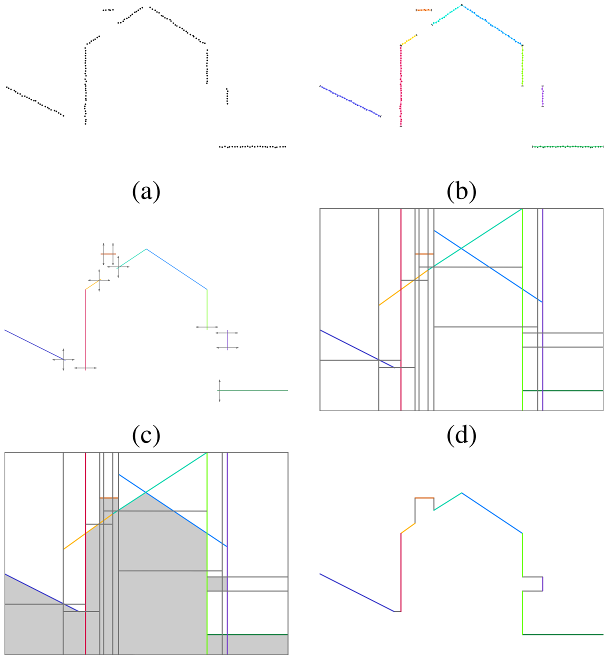

(a) (b) (c) (d) (e) (f)

Figure 1. 2D illustration of the outline of our algorithm: (a) input points, (b) detected primitives with boundary points, (c) detected and ghost primitives, (d) induced cell complex, (e) labeled cell complex and (f) output mesh.

2. Method Description

The input to our algorithm is a point set $\{p_i,\ i \in \{1,\dots,N\}\}$ in $\mathbb{R}^3$ along with visibility information. For clarity of the exposition, we assume that the latter consist of a single line-of-sight per input point, although 3D positions are typically triangulated from several emitters/sensors (except for time-of-flight range scanners). Let c_i be the 3D position of the reference emitter/sensor of p_i. Note that c_i may be different for every input sample, *e.g.* in the case of mobile LiDAR data. We also suppose that we have a reliable vertical direction since this is crucial for incorporating our priors on urban and architectural scenes.

Figure 1 illustrates the outline of our algorithm on a simple 2D case. Its steps are described in the next subsections.

2.1. Definition of adjacency relations

To be able to assess the connectedness of a primitive, we have to equip the input point cloud with some adjacency relations. While this is straightforward for range images or height fields [5, 21, 32, 33], unstructured 3D data require more elaborate tools.

Specifically, we use the *mutual k-nearest neighbors graph*: p_i and p_j are considered neighbors if and only if p_j belongs to the k-nearest neighbors of p_i and p_i belongs to the k-nearest neighbors of p_j. As demonstrated in [30], the reciprocity of the k-NN relationship is an effective criterion for outliers detection. Consequently, our definition of adjacency tends to isolate outliers into small connected components, thereby uncluttering inlier primitives.

2.2. Estimation of oriented tangent planes

In order to ensure a first-order approximation of input data, an estimation of tangent planes at the user-defined scale is required.

To achieve this, for each input point p_i, we perform a principal component analysis (PCA) on the ball neighborhood of radius 2σ centered on p_i. First, the eigenvector associated to the smallest eigenvalue yields the best fitting plane in least-squares sense, which constitutes our initial estimate T_i^0 of the tangent plane at p_i. Second, the ratio of the two smallest eigenvalues gives a local indicator q_i of the complexity of the neighborhood in terms of curvature, noise level and outlier proportion.

As the initial estimate T_i^0 smoothes out normal discontinuities, we refine it by iteratively *(i)* partitioning points contained in the ball neighborhood as inliers or outliers depending on whether their distance to T_i^l exceeds $\sigma/2$, *(ii)* fitting T_i^{l+1} to these inliers using PCA. This process usually converges in two or three iterations and produces a final tangent plane estimate T_i. Moreover we give T_i a consistent orientation by making sure that c_i is on its positive side.

We can now make an important definition. An input point p_i is an *inlier* of an oriented plane P if and only if the three following conditions are met: *(i)* **visibility:** c_i is on the positive side of P, *(ii)* **position:** the distance from p_i to P is less than σ, *(iii)* **tangency:** the angle between T_i and P is less than θ.

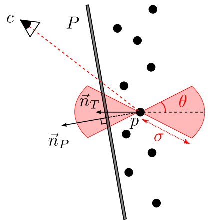


Figure 2. Inlier point p of the plane P.

2.3. Detection of planar primitives

This crucial step of our algorithm constructs a first-order approximation of the input data by a set of planar primitives. A planar primitive consists in *(i)* an oriented 3D plane and *(ii)* a connected subset of inlier points which encodes its spatial extent.

We could have resorted to RANSAC as in [23, 17], but a simpler deterministic region growing approach has proven sufficient, even on challenging data. A well-known limitation of this approach is the critical influence of seed choice, so we have designed a careful seeding strategy, based on the following observation: region growing easily gets trapped in erroneous solutions whenever the seed point is close to sharp edges, corners, high-curvature or high-noise regions. Hence our algorithms exploits the indicator of local complexity q_i: we iteratively grow planar primitives from the unassigned input point of better planarity. This way, easy decisions are made first, while difficult ones are postponed to a later stage of the algorithm in which we expect that most ambiguities be resolved.

After a primitive has been computed, its constituting points are discarded from further consideration. Hence the main detection loop terminates, and every input point is assigned to exactly one primitive. Finally, we discard planar primitives which are significantly smaller than the scale of analysis, as they unnecessarily increase the size of the approximation. Specifically, we check whether their standard deviation along their second principal axis exceeds $\sigma/2$.

We now detail the extraction of a planar primitive from an adequately chosen seed point p_i. It again follows a classical alternate segmentation/fitting approach. Given an oriented plane P^l, we unambiguously define a planar primitive as the maximal connected subset of inliers of P^l that contains p_i. Reciprocally, a PCA of these inliers yields a novel estimation P^{l+1} of the supporting plane of the primitive. We bootstrap this process with the tangent plane T_i as the initial estimate P^0 and iterate until convergence (or cycle).

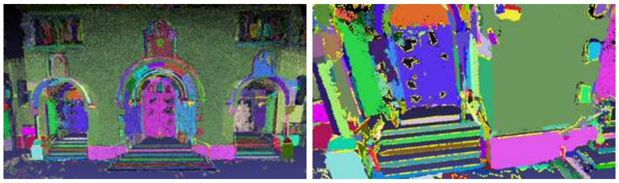


Figure 3. Detected planar primitives (left, color-coded) and zoom on boundaries (left, yellow points) for *Herz-Jesu-P25*.

2.4. Addition of ghost primitives

A single, small missing plane between two closely parallel ones can jeopardize the entire reconstruction if constrained to the set of detected primitives (see Figure 4). We make up for these gaps by incorporating additional, unseen, ghost primitives.

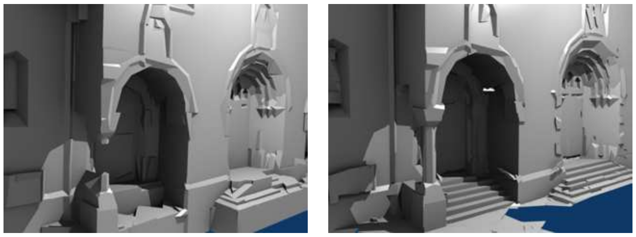


Figure 4. Importance of ghost primitives. Left: reconstruction with detected primitives only. Right: reconstruction with detected plus ghost primitives.

In practice, we derive these ghost primitives from boundaries of detected primitives. First, we delineate the boundary of a planar primitive by computing the 2D α-shape of the projections of its constituting points [9], with a characteristic radius equal to the scale ($\alpha = \sigma^2$). Second, we approximate this boundary up to a distance σ with a set of line primitives, obtained by iterative merging. Third, we discard boundary lines adjacent to another primitive (compared with the scale σ), thus retaining only true boundaries (see Figure 3). Finally, for each of these boundary lines, we add the following two ghost planes, provided they are not parallel to the original primitive, up to the tolerance angle θ: the vertical plane containing the line and the plane orthogonal to the former and passing through the line as well.

2.5. Construction of the cell complex

A 3D arrangement of the supporting planes of all the primitives contains the entire structure of the scene and could be computed using the algorithm of [10]. However, this naive solution suffers from a very high algorithmic – $\Theta(n^3)$ – and combinatorial complexity. Indeed, each additional planar primitive involves to split all current polyhedral cells across its supporting plane; in addition, the spatial extent of a primitive is not taken into account: it makes no sense that, for instance, a chimney on a roof may intervene in the reconstruction of a building at the other end of the scene. A much more local modification of the partition would be desirable to keep the number of cells, and thus the computational effort and the memory cost, sustainable. We propose a more elaborate algorithm to cope with these limitations.

First, we use a two-level hierarchy in the decomposition. The first level of the hierarchy consists of a coarse uniform rectilinear volumetric grid which decomposes the domain into cubical super-cells of medium size, typically 10 times the scale σ. This grid forms an initial, trivial arrangement of equally-spaced axis-aligned planes that subsequently localizes the reach of the primitives, hence decreasing the complexity of the complex. The second level of the hierarchy consists of a partition of each cubical super-cell into convex polyhedral cells. The way this partition is built lies in-between a 3D arrangement and a binary space partition: our algorithm splits a subset of the cells traversed by the supporting plane, whereas an arrangement would split all cells, and a binary space partition would split a single cell. Specifically, we only split the cells spanned by the planar primitive being inserted, *i.e.* cells intersected by the supporting plane and by a σ-ball centered on a point of this primitive. This expansion guarantees a greater robustness of the reconstruction and ensures that the intersecting primitives (as defined in the analysis of boundary segment) actually intersect in the complex.

Please note that, contrarily to the case of an arrangement, the final partition depends on the order in which the primitives are inserted. We design this order to minimize the complexity of the final partition, using two heuristics. The first heuristic exploits the fact that there are many quasi-horizontal and quasi-vertical primitives (compared to the tolerance angle θ), with two variants for aerial and ground data respectively. In the case of aerial data, we insert all quasi-vertical primitives first, which is likely to lead to a low-complexity 2D-like partition that contains the layout of buildings. We then insert all quasi-horizontal primitives, which leads to simple partitions in the existing cells while limiting the extent of these primitives to relevant zones. Finally, we insert slanted primitives. However, for ground data, we insert first quasi-horizontal primitives (which lead to trivial partitions in the different cubical super-cells), then quasi-vertical ones. The dominant horizontal direction of lines of sight, which yields a majority of horizontal missing primitives rather than horizontal ones, accounts for this reverse order. The second heuristic we use is to insert primitives in decreasing order of size. Indeed, smaller primitives are likely to traverse few of the cells induced by larger primitives, contrarily to what would happen in reverse order.

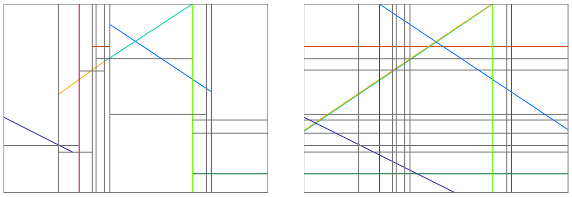


Figure 5. 2D illustration of the better conciseness of our polyhedral cell complex (left) over a complete arrangement (right).

2.6. Labeling of the cell complex

We formulate the surface reconstruction problem as an optimal binary labeling of the complex, according to an energy function designed to maximize the visibility-consistency of the surface. Each cell is labeled as empty or occupied, and we extract a surface as the union of all oriented facets from an occupied cell to an empty one. This surface bounds the 3D volume of the union of occupied cells, hence is watertight and intersection-free.

This labeling problem is handled within the framework of minimum $s-t$ cut on the cell-adjacency graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ of the partition: the vertices $\mathcal{V}$ are the cells of the polyhedral complex while the edges $\mathcal{E}$ link adjacent cells, *i.e.* correspond to the facets of the complex. More precisely, we consider the directed adjacency graph with a (directed) edge corresponding to each oriented facet of the complex. $\mathcal{V}$ is augmented with two additional seeds, a source s and a sink t, with edges from s to each cell and from each cell to t. All edges have non-negative weights w.

A $s-t$ cut $(\mathcal{S}, \mathcal{T})$ is a partition of $\mathcal{V}$ into two disjoint sets $\mathcal{S}$ and $\mathcal{T}$ such that $s \in \mathcal{S}$ and $t \in \mathcal{T}$. The cost of an $s-t$ cut is the sum of the weights of the edges from $\mathcal{S}$ to $\mathcal{T}$. Efficient algorithms with low-polynomial complexity exist to find the $s-t$ with minimal cost, allowing a global minimization of the energy. A graph partitioning $(\mathcal{S}, \mathcal{T})$ corresponds to a binary labeling of cells – where cells in $\mathcal{S}$ and $\mathcal{T}$ are respectively empty and occupied – and the cost of the cut to the energy of the associated surface. Weights of edges joining the source or the sink penalize the associated cells, while weights of edges between two cells penalize the associated oriented facets.

The energy is composed of three terms, a visibility term $E_{vis}(\mathcal{S})$, an orientation term $E_{orient}(\mathcal{S})$ and a regularization one $E_{regul}(\mathcal{S})$:

$$E(\mathcal{S}) = E_{vis}(\mathcal{S}) + E_{orient}(\mathcal{S}) + E_{regul}(\mathcal{S}). \quad (1)$$

The visibility term $E_{vis}(\mathcal{S})$ penalizes the oriented facets crossed by the lines of sight of the fitted points. In order to account for the scale precision σ, a line of sight goes from the camera center, to an ending point behind the fitted point, shifted to a distance of σ in this direction. As shown in Figure 6, along this line, all oriented facets in the opposite direction are penalized with a weight α_{vis}, except for facets for which this point is an inlier at this scale σ (see Section 2.2). Moreover, the cells containing the camera center and the ending point are respectively linked to the source s and the sink t, with the same weight α_{vis}.

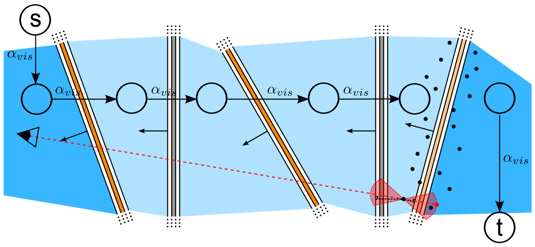


Figure 6. Visibility. A fitted point penalizes the oriented facets of the grid (grey facets) and of the primitives of which it is not an inlier (darker orange) along its line of sight (red dashed line), and links its terminal cells (darker blue).

The orientation term $E_{orient}(\mathcal{S})$ penalizes facets corresponding to detected primitives but of opposite orientation with an infinite weight, preventing genuine primitives – which are oriented 3D planes – to be used in the reverse direction in the reconstructed surface. Note that neither ghosts primitives nor grid planes appear here because they have no orientation – hence can be used in both directions.

Finally, the regularization term $E_{regul}(\mathcal{S})$ penalizes the area of the surface compared to the mean surface density d of the detected primitives, with different weights α^{prim}_{area} and α^{grid}_{area} for facets belonging respectively to a primitive (detected or ghost) and to a plane of the initial grid:

$$E_{regul}(\mathcal{S}) = \alpha^{prim}_{area} d\, \mathcal{A}(\mathcal{S}_{prim}) + \alpha^{grid}_{area} d\, \mathcal{A}(\mathcal{S}_{grid}). \quad (2)$$

The parameters α_{vis}, α_{area}^{prim} and α_{area}^{grid} balance the various terms of the energy which is defined up to a factor, so we can fix one of them, *e.g.* $\alpha_{vis} = 1$. The other parameters α_{area}^{prim} and α_{area}^{grid}, are easily set (see Section 3.2).

2.7. Surface extraction

The boundary of the occupied cells of the polyhedral partition forms a watertight polygonal mesh, free of self-intersection. However this mesh does not fulfill all our needs: it is verbose, since many facets of the polyhedral partition may belong to the same planar primitive. These redundant coplanar facets have to be identified and merged into a single polygonal facet. Moreover, this polygonal facet possibly features concave parts, holes and multiple connected components, so it has to be triangulated to meet display or post-processing requirements.

We reach both goals using a *constrained 2D Delaunay triangulation* [7]: for each planar primitive that appears in the reconstruction, we build a 2D triangulation of the vertices of the corresponding polygonal facet that tries to be as Delaunay as possible, while including among its edges the boundaries of this polygonal facet, although the latter do not necessarily constitute Delaunay edges. Interestingly, this close-to-Delaunay property discourages badly-shaped triangles.

Figure 7 illustrates the surface extraction step. It demonstrates the higher conciseness and quality of the output mesh over the raw boundary of the polyhedral partition.

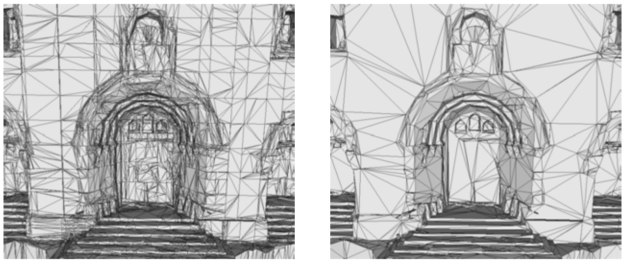


Figure 7. Surface extraction. Left: boundary of the polyhedral cell complex. Right: final mesh obtained by a constrained 2D Delaunay triangulation on each planar primitive.

3. Experimental Validation

3.1. Implementation aspects

Our C++ implementation is based upon the *Computational Geometry Algorithms Library* (CGAL: `http://www.cgal.org`). CGAL provides all the needed basic geometric operations, as well as robust and efficient algorithms for nearest neighbors search, principal component analysis and 2D triangulations.

As there is currently no publicly available code for computing 3D plane arrangements, and *a fortiori* our more general polyhedral cell complex, we have developed our own source code on top of the CGAL kernel. The use of exact geometric computation [31] has proven essential: the frequent occurrence of quasi-degenerate configurations in large real-world data make rounding errors of floating-point arithmetic prohibitive.

To perform the $s-t$ cut, we use the Boost Graph Library implementation of Kolmogorov's max-flow algorithm [3].

3.2. Parameters

A major strength of our method is that the user basically needs to set only one parameter, the scale distance σ, depending on the scale of the data, the noise level and the desired precision. For all the other parameters, we used the constant values of Table 1 throughout our experiments.

k	number of nearest neighbors	10
θ	scale angle	25°
α_{area}^{prim}	regularization for primitives	0.05
α_{area}^{grid}	regularization for the grid	0.5

Table 1. Values of constant parameters.

As for the initial cubic decomposition, we used cubes of size $10\,\sigma$ for ground data, and squared columns of base size $10\,\sigma$ and of infinite height for aerial data, to allow the vertical completion of buildings up to the ground.

3.3. Results

We tested our algorithm on datasets of various type and size. We show our results in Figure 8.

The first dataset, *block*, is a synthetic, mechanical part, typical of CAD models, that allows us to check our algorithm on a simple example. It has already been reconstructed in [17] for instance. We obtain a concise mesh, with a good approximation of cylindric parts by planes. Since the data is complete, the reconstruction does not resort to any ghost primitive.

The other datasets were obtained from multi-view imagery by winner-take-all matching based on multi-scale normalized cross correlation. Hence our input point clouds present a large proportion of outliers, as clearly visible in Figure 8(b) for instance.

entry-P10, *fountain-P11* and *Herz-Jesu-P25* are ground, architectural scenes, taken from the publicly available benchmark of [27]. We obtained concise, simplified models of these scenes that highlight their major planar features and preserve sharp edges. Note how we recovered window moldings in *entry-P10* or steps in *Herz-Jesu-P25*.

Cluny and *Marseille* have been obtained from aerial (resp. helium balloon and airborne) imagery. Our algorithm successfully infers the overall structure of the scene despite many missing façades in the input data. It also recovers some roof superstructures, in particular chimneys and dormer windows of the Cluny Abbey.

Table 2 gathers quantitative data about our experiments: number of input points, of detected primitives, of ghost primitives and of output mesh facets, as well as timings measured on a 2.66 Ghz Intel Xeon PC. Notably, the computation time does not exceed 35 minutes on our largest datasets of several millions of points.

We would like to emphasize that the main objective and achievement of our work are the conciseness and idealization of the output models, and not a fine accuracy. We obviously do not intend to compete with dense multiview stereo reconstruction algorithms for example, [29] reports 2M facets on *entry-P10* and 1.6M facets on *fountain-P11*. On the other hand, we do not intend to compete with dense multiview stereo reconstruction algorithms as regards accuracy, as our main goals are conciseness and idealization.

4. Conclusion

We have proposed a novel method for automatic simplified piecewise-planar 3D reconstruction and completion from massive point clouds corrupted by noise, outliers, and severe occlusions. We have demonstrated the benefits of our approach on challenging urban and architectural scenes.

Future work includes incorporating verticality, horizontality and orthogonality priors since the primitives detection step, in order to improve the visual acceptability of our 3D models. Also, we plan to explore the feasibility of a streaming approach for out-of-core 3D reconstruction of entire cities from ground or oblique aerial imagery.

References

[1] S. Baker, R. Szeliski, and P. Anandan. A layered approach to stereo reconstruction. In *ICCV*, 1998.

[2] M. Botsch, M. Pauly, L. Kobbelt, P. Alliez, B. Lévy, S. Bischoff, and C. Rössl. Geometric modeling based on polygonal meshes. In *SIGGRAPH*, 2007.

[3] Y. Boykov and V. Kolmogorov. An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Vision. *PAMI*, 26, 2004.

[4] D. Bradley, T. Boubekeur, and W. Heidrich. Accurate multiview reconstruction using robust binocular stereo and surface meshing. In *CVPR*, 2008.

[5] C. Chen and I. Stamos. Range image segmentation for modeling and object detection in urban scenes. In *3DIM*, 2007.

[6] J. Chen and B. Chen. Architectural Modeling from Sparsely Scanned Range Data. *IJCV*, 78, 2008.

[7] L. P. Chew. Constrained Delaunay triangulations. In *SCG*, 1987.

[8] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, 1996.

[9] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the Shape of a Set of Points in the Plane. *TIT*, 29, 1983.

[10] H. Edelsbrunner, J. O'Rourke, and R. Seidel. Constructing arrangements of lines and hyperplanes with applications. *SICOMP*, 15, 1986.

[11] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Manhattan-World Stereo. In *CVPR*, 2009.

[12] Y. Furukawa and J. Ponce. Accurate, Dense, and Robust Multi-View Stereopsis. In *CVPR*, 2007.

[13] A. Hoover, G. Jean-Baptiste, X. Jiang, P. J. Flynn, H. Bunke, D. B. Goldgof, K. Bowyer, D. W. Eggert, A. Fitzgibbon, and R. B. Fisher. An experimental comparison of range image segmentation algorithms. *PAMI*, 18, 1996.

[14] P. Jenke, B. Krückeberg, and W. Straßer. Surface Reconstruction from Fitted Shape Primitives. In *VMV*, 2008.

[15] P. Jenke, M. Wand, M. Bokeloh, A. Schilling, and W. Straßer. Bayesian Point Cloud Reconstruction. *CGF*, 25, 2006.

[16] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. In *SGP*, 2006.

[17] P. Labatut and J.-P. Pons. Hierarchical shape-based surface reconstruction for dense multi-view stereo. In *3DIM*, 2009.

[18] P. Labatut, J.-P. Pons, and R. Keriven. Efficient Multi-View Reconstruction of Large-Scale Scenes using Interest Points, Delaunay Triangulation and Graph Cuts. In *ICCV*, 2007.

[19] T. Lindeberg. Scale-space theory: A basic tool for analysing structures at different scales. *JAS*, 21, 1994.

[20] B. Micusik and J. Kosecka. Piecewise Planar City 3D Modeling from Street View Panoramic Sequences. In *CVPR*, 2009.

[21] C. Poullis and S. You. Automatic Reconstruction of Cities from Remote Sensor Data. In *CVPR*, 2009.

[22] R. Schnabel, P. Degener, and R. Klein. Completion and Reconstruction with Primitive Shapes. *CGF*, 28, 2009.

[23] R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for Point-Cloud Shape Detection. *CGF*, 26, 2007.

[24] S. Seitz, B. Curless, J. Diebel, D. Scharstein, and R. Szeliski. A comparison and evaluation of multi-view stereo reconstruction algorithms. In *CVPR*, 2006.

[25] S. N. Sinha, P. Mordohai, and M. Pollefeys. Multi-View Stereo via Graph Cuts on the Dual of an Adaptive Tetrahedral Mesh. In *CVPR*, 2007.

[26] S. N. Sinha, D. Steedly, and R. Szeliski. Piecewise Planar Stereo for Image-based Rendering. In *ICCV*, 2009.

[27] C. Strecha, C. von Hansen, L. V. Gool, P. Fua, and U. Thoennessen. On benchmarking camera calibration and multi-view stereo for high resolution imagery. In *CVPR*, 2008.

[28] F. Taillandier and R. Deriche. Automatic Buildings Reconstruction from Aerial Images: a Generic Bayesian Framework. In *IAPRSSIS*, volume 35, 2004.

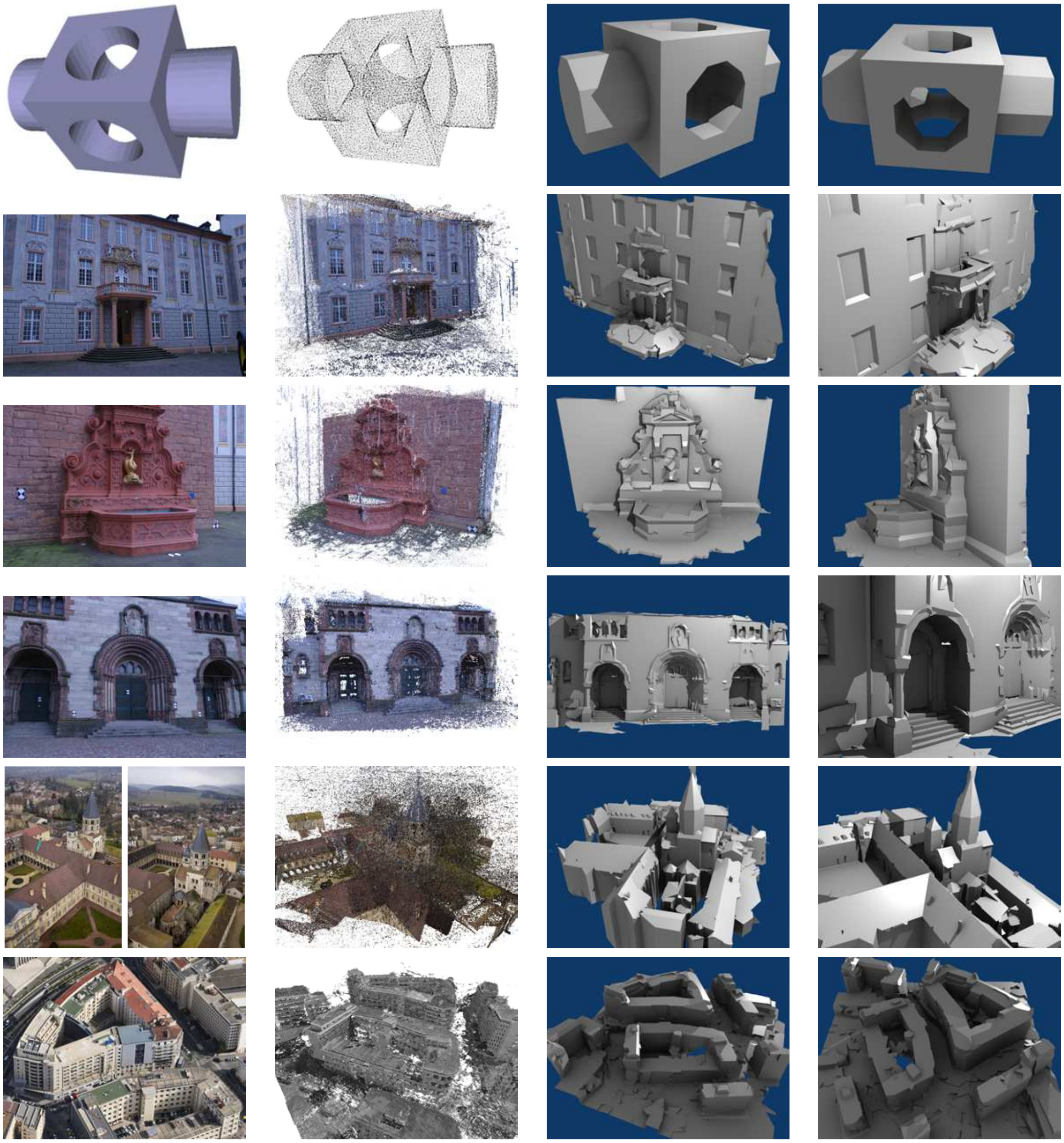
[29] H. Vu, R. Keriven, P. Labatut, and J.-P. Pons. Towards high-resolution large-scale multi-view stereo. In *CVPR*, 2009.

[30] T. Weyrich, M. Pauly, R. Keiser, S. Heinzle, S. Scandella, and M. Gross. Postprocessing of scanned 3D surface data. In *PBG*, 2004.

[31] C.-K. Yap. Towards exact geometric computation. *CGTA*, 7, 1997.

[32] G. Yu, M. Grossberg, G. Wolberg, and I. Stamos. Think Globally, Cluster Locally: A Unified Framework for Range Segmentation. In *3DPVT*, 2008.

[33] Q.-Y. Zhou and U. Neumann. Fast and extensible building modeling from airborne LiDAR data. In *GIS*, 2008.



(a) picture of the scene (b) input point cloud (c) Output polygonal mesh

Figure 8. Piecewise-planar reconstructions on various datasets.

Datasets	# points	σ (m)	# primitives	# ghosts	# output facets	time (s)
block	49283	1	52	0	208	7
entry-P10	1393012	0.1	452	1681	5376	550
fountain-P11	1526957	0.05	491	1439	4139	730
Herz-Jesu-P25	737990	0.05	895	4821	15025	265
Cluny	3019228	0.2	1241	6285	19889	2050
Marseille	3313834	1	1099	2415	11078	1400

Table 2. Statistics on processed models.