Surface Reconstruction through Point Set Structuring

Florent Lafarge, Pierre Alliez

Surface Reconstruction through Point Set Structuring

Florent Lafarge and Pierre Alliez

Inria Sophia Antipolis - Méditerranée, France

Abstract

We present a method for reconstructing surfaces from point sets. The main novelty lies in a structure-preserving approach where the input point set is first consolidated by structuring and resampling the planar components, before reconstructing the surface from both the consolidated components and the unstructured points. The final surface is obtained through solving a graph-cut problem formulated on the 3D Delaunay triangulation of the structured point set where the tetrahedra are labeled as inside or outside cells. Structuring facilitates the surface reconstruction as the point set is substantially reduced and the points are enriched with structural meaning related to adjacency between primitives. Our approach departs from the common dichotomy between smooth/piecewise-smooth and primitive-based representations by gracefully combining canonical parts from detected primitives and free-form parts of the inferred shape. Our experiments on a variety of inputs illustrate the potential of our approach in terms of robustness, flexibility and efficiency.

1. Introduction

Reconstructing surfaces from defect-laden point sets is still one of the major challenges in geometry processing, computer vision and robotics. Driven by technological advances on sensors, data are progressively evolving in terms of scale and accuracy, ranging from defect-free point sets representing a single object, e.g., a statue, to large defect-laden point sets describing complex environments such as urban scenes [MWA*12]. This evolution motivates new surface reconstruction methods incorporating more structural and semantical considerations.

1.1. Previous work

Many different approaches have been proposed in the literature for reconstructing surfaces. In our context we divide these approaches into two main categories: smooth and primitive-based.

Smooth and piecewise-smooth reconstructions. Smooth approaches recover C_1-surfaces using either implicit or explicit representations. Implicit methods indirectly describe surfaces using level-sets [KBH06, HK06, LB07]. They commonly rely on two key elements: (i) a 3D function computed from the input points allowing to both approximate and smooth the surface, as signed distances or radial basis functions, and (ii) a solver for extracting the surface, e.g., linear least squares or graph-cuts. Implicit

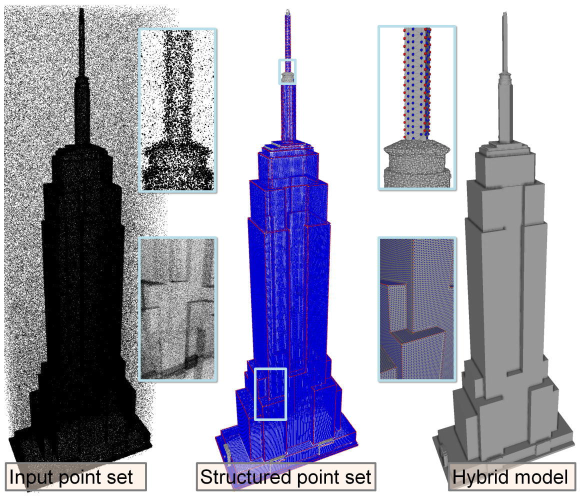


Figure 1: *Empire State Building. Our method structures defect-laden point sets from a configuration of planar primitives (bottom close-ups) while preserving the details everywhere else (top close-ups). The resulting model is a hybrid surface combining structures, accuracy and low complexity.*

methods are effective solutions but most of them require specific additional attributes associated to point locations such as oriented normals, lines of sight or measurement confidences. Explicit methods reconstruct surfaces using mesh-based structures such as Delaunay triangulations.

Relying on the idea that points close to the surface are also close in space, these methods provide convincing results when the sampling is dense enough and hampered with only little amount of noise [ABK98, LPK09]. Several methods have been proposed to preserve sharp features by either preliminarily detecting smooth regions [FCOS05] and sharp crease [SYM10] or inserting local shape priors into the reconstruction process [GSH*07].

Primitives-based reconstructions. Methods based on detected primitives have become popular in recent years. Efficient algorithms are now available for detecting geometric primitives [SWK07] and for readjusting them according to global relationships such as as coplanarity, coaxiality or parallelism [LWC*11]. Primitive-based methods are a relevant alternative to smooth reconstructions when the inferred surfaces contain many canonical parts such as planar components, and for large data sets as dealing with primitives may improve computational efficiency. Planar primitives are most commonly used. Chen et al. [CC08] and Chauve et al. [CLP10] use an arrangement of planes for approximating surfaces, which provides a rich solution space but only when all planes are perfectly detected. Missing planes may be completed by ghost components [CLP10], but this comes at the price of a lower surface accuracy. Vanegas et al. [VAB12] propose a method for reconstructing urban structures from laser range scans under the restrictive Manhattan-World assumption. Lafarge and Mallet [LM12] reconstruct buildings from Lidar scans by arranging planar and quadric surfaces, albeit this method is limited to 2.5D reconstructions. Schnabel et al. [SDK09] reconstruct surfaces while filling holes from incomplete point sets through graph-cut based primitive extension by assuming that each hole can be entirely described by primitive arrangments. Note also that some methods recover structures from surfaces [CSAD04] and Multi-View Stereo (MVS) images [LKBV13]. The latter methods however require to preliminarily extract an accurate mesh from the input point set.

1.2. Motivations

Primitive-based methods are particularly attractive when dealing with large scans containing canonical parts, but in general they remain less robust than smooth solutions. The first concern lies into the restrictive representation, as a complex scene can rarely be entirely described by a set of canonical primitives. The second concern lies into the reliability of the primitive detection step: an ideal primitive and primitive adjacency extraction with no under- nor over-detected parts cannot be guaranteed.
A common solution consists of inserting prior knowledge into the reconstruction procedure. Examples of such knowledge are a Manhattan-world assumption [VAB12] or the presence of symmetries [MPWC12]. These assumption models may greatly improve the reconstructions for specific types of scenes, but usually render the reconstruction algorithm more complex and parameter-dependent. Another solution is to interactively complete or correct primitive-based methods, such as a user-assisted snapping approach [ASF*12].
The proposed solution consists of not reconstructing the *entire scene* with primitives, but instead relying on robust smooth reconstruction for the non-canonical parts of the scene. We provide a flexible hybrid framework able to reconstruct altogether canonical parts by primitives and non-canonical part by free-form surfaces. Only planar primitives are considered. This choice is motivated by our application, *i.e.* the reconstruction of urban structures where non-planar primitives are relatively marginal compared to reverse engineering applications [AP10]. Our experiments show that this approach robustly generates faithful surfaces in spite of the limitations of primitive-based representations and of the defect-laden configurations of detected primitives. Such a hybrid framework suggests that it is possible to cumulate some advantages of both worlds (primitive and smooth), which are robustness, scalability and compaction.

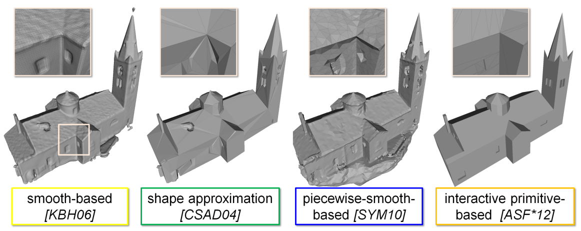

smooth-based [KBH06]
shape approximation [CSAD04]
piecewise-smooth-based [SYM10]
interactive primitive-based [ASF*12]

Figure 2: *Church model reconstructed by different methods. Note that the variational shape approximation (VSA) method (middle left) has been applied from the smooth model (left). The primitive-based model (right) recovers and completes the building structure but requires 15 minutes of snapping-based user interaction.*

1.3. Contributions

Our surface reconstruction algorithm provides the following contributions:

- *Structured point set:* The input point set is not used directly for reconstructing the inferred surface. Similarly in spirit to the concept of data consolidation [ZSW*10] for rectifying point sets with missing regions, the input point set is first analyzed by detecting planar primitives. From these primitives we extract a structure deriving from their adjacency relationships, and re-sample the primitives such that the structural elements, including sharp features, are preserved under a Delaunay triangulation.
- *Efficient min-cut formulation.* We propose a new efficient min-cut formulation which combines structure, geometry, and visibility considerations in order to guarantee intersection-free, 2-manifold surfaces as outputs.
- *Hybrid surface.* The final reconstructed surface combines

both structured canonical parts idealizing the planar elements, and free-form parts representing either the non-planar elements of the inferred scene or the undetected yet canonical parts of the scene. Even in presence of under-detection the reconstruction is handled gracefully as no parts are missing. On under-detected areas the final reconstruction is simply less structured and non-idealized.

Our algorithm requires as input a raw point set and a set of planar primitives preliminarily detected from this point set. The primitive extraction is performed by common procedures such as RANSAC [SWK07], region growing [LM12] or Gaussian sphere projection [CC08]. Primitives are detected under tolerance ε, which means that there are no outliers within ε distance to the primitives.
As depicted by Fig.1, we adopt a two step strategy. First, the input point set is structured from the extracted primitives (Section 2). Second, the surface is reconstructed from the structured point set using a min-cut formulation over a 3D Delaunay partitioning of the space (Section 3).

2. Structuring

Given a configuration of planar primitives extracted under a tolerance ε, the structuring process turns the input point set into another point set in which each point is associated to one of the four structural types, i.e., *planar*, *crease*, *corner* and *clutter*, as depicted by Fig. 3. A point labeled as *planar* (resp. *crease* and *corner*) is associated to one (resp. two and three or more) planar primitives. The structuring process acts on three key points:

- *Meaning insertion:* Each point is enriched with structural information related to its associated extracted primitives (zero, one or more) and their adjacencies. This information is used in subsequent reconstruction processes.
- *Structure preservation under space partitioning:* Points are sampled so that the structures induced from the extracted primitives are preserved when subdividing the space with a partitioning scheme, here a Delaunay 3D-triangulation.
- *Simplification:* The point set is re-sampled on canonical parts using the primitives, without losing the details on the free-form parts which are kept untouched.

Plane anchors. The first step of the structuring process consists in replacing the points fitted to the primitives by an ideal layout of points, both light and preserving the primitive surfaces in the Delaunay triangulation. To do so, an occupancy binary 2D-grid projected in the planar primitive is created. The width (side length) of a unitary square surface element of the grid is denoted by L_p. A surface element of the grid is marked occupied if at least one fitted point orthogonally projects within its domain or, subsequently, if it is surrounded by only occupied elements. The centers of all occupied elements form the new layout of points whose structural type is *planar*.

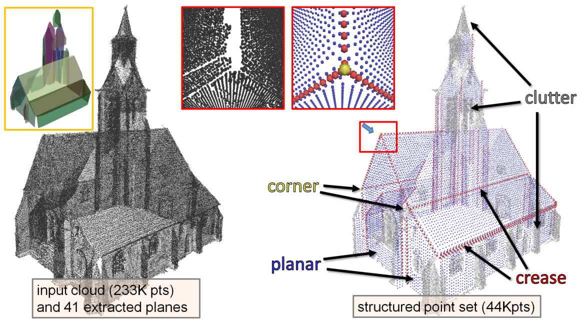


Figure 3: *Structuring principle. Plane anchors (blue), creases (red) and corners (yellow) are positioned in the new point set to describe the main structures of the building. The other components such as windows or doors are defined as clutter points (grey).*

In order for the detected planes to appear in the final model, the width L_p must be chosen so that the subsequent Delaunay triangulation will link the *planar* points with triangles. This linking condition is guaranteed when the equatorial circumsphere of these triangles is empty in spite of the presence of outliers. As illustrated by Fig. 4, this condition is matched when $L_p < \sqrt{2}\,\varepsilon$. In other words, the lower the tolerance ε, the denser the layout of *planar* points. Note that weighted Delaunay triangulation [CDR07] could be considered for obtaining a sparser layout of points. However it would require a complex analysis of the outlier positions to guarantee the existence of the facets supporting the primitive.

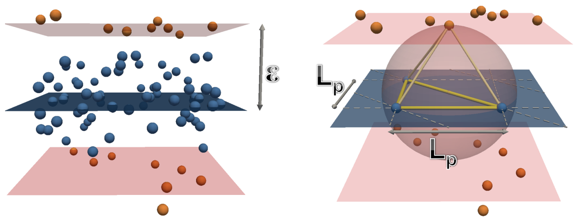


Figure 4: *Plane anchor sampling. Inliers to the planar primitive (blue dots) are removed from the point set and new points are created on the plane so that the induced facets exist in the Delaunay triangulation. The distance L_p between two successive added points is determined in function of the tolerance ε so that the equatorial circumsphere of an induced facet stays inside the ε-domain of the primitive. The right layout represents the critical case where the condition is not valid anymore.*

Creases. Creases are created between adjacent primitives in order to consolidate their connection. The adjacency relationship between two primitives is defined using the K-nearest neighbor (KNN) graph of the input points. Two primitives are said adjacent if at least two points fitted each to one of the two primitives are mutual neighbors in the KNN graph. Crease points are sampled uniformly along the intersection line of each pair of adjacent primitives. First, an occupancy 1D-grid supporting the intersection line is created, the length of an element being denoted by L_c. A crease is then generated at the center of a 1D-grid element if the ball of radius L_c contains at least one inlier point of each incident primitive.

Crease points and corresponding planar points will be linked in triangles by the subsequent Delaunay triangulation if their circumspheres stay in the ε-domain of the two primitives (pink area on Fig. 5). By denoting θ, the angle between the two adjacent primitives, and h_c the distance between the intersection line and the first row of planar anchors, this linking condition requires placing the crease points so that

$$\begin{cases} L_c = 2\varepsilon \\ h_c = \varepsilon \times \cos\frac{\theta}{2} \end{cases} \quad (1)$$

Note that when the angle between the two primitives is close to 180°, the placement of the first row of anchor points becomes unstable. Although coplanar primitives are rarely adjacent in practice, we prevent this problem by disconnecting from the adjacency graph adjacent primitives with an angle close to 180° (> 170° in all examples shown).

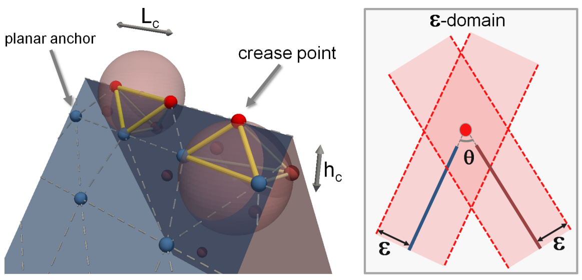


Figure 5: *Crease sampling. Crease points are sampled along the intersection line of two adjacent planar primitives, separated by the distance L_c and positioned at the distance h_c from the planar anchor points. The existence of the crease facets (yellow triangles) in the Delaunay triangulation is guaranteed when their circumspheres (red spheres) stay in the ε-domain of the two primitives (right). This leads to condition the choice of L_c and h_c according to ε and θ.*

Corners. Points are also positioned at the structure corners. The 3−cycles are first detected from the primitive adjacency graph. The potential n−cycles with $n \geq 3$ are then extracted from the detected 3−cycles. For $n = 3$, the corner location is computed as the intersection of the three planar primitives. For $n > 3$, it is computed as the barycenter of the points contained in its associated 3-cycles.

Clutter. The input points which have not been detected as belonging to planar primitives are inserted into the structured point set with the label *clutter*. Optionally, the small isolated components of *clutter* points may be removed using a clustering procedure based on the Euclidean distance to neighbors. We use this option to remove outliers and non-significant components from the structured point set.

The point set structuring is controlled by the tolerance parameter ε. As illustrated by Fig.6, increasing ε progressively structures the point set while reducing the amount of clutter points. When the point sets are ideally and fully structured (*e.g.*, second and fourth examples in Fig. 6), the surface can be straightforwardly extracted by a polygonalization procedure. However, such cases rarely occur from real-world data as free-form elements and under/over-detections of primitives and adjacencies are common. A robust procedure is required for extracting the surfaces from structured point sets.

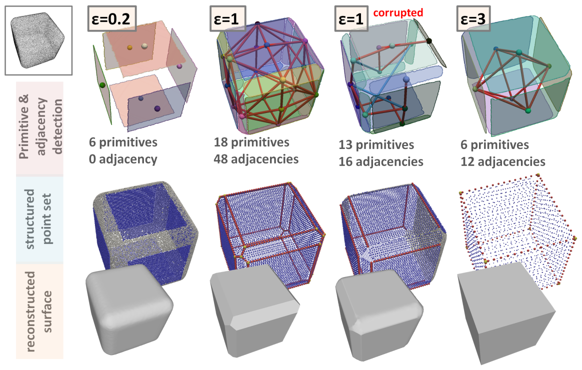


Figure 6: *Smooth cube structured with different ε-values. Increasing ε progressively structures the input point set (top left) while maintaining a coherent reconstructed surface. In the third example, both primitives and adjacencies have been randomly corrupted (5 primitives and 6 adjacencies are removed, and 3 wrong adjacencies depicted as blue segments are added). While under- and over-detection of primitives and adjacencies reduce the quality of the structuring, by either omitting or overly creating creases and corners, this does not fully hamper the reconstruction thanks to the free-form components.*

3. Surface extraction

The surface extraction step relies on the structured point set. The general framework builds on the creation of a space partition from which each volume element is labeled either inside or outside the inferred surface.

3.1. Labeling Delaunay tetrahedra

The space subdivision is obtained by constructing a 3D Delaunay triangulation from the structured point set. This partitioning procedure provides us with several relevant properties. Constructed from the structured point set, the triangulation preserves the structures both in terms of geometry (tetrahedra do not intersect the surfaces induced from the primitives) and in terms of meaning (each vertex of the 3D-triangulation inherits from a structural type assigned during structuring). As illustrated by Fig. 7, the partitioning also has the advantage of being light as in general the structured point set comprises fewer points than the input point set.

In order to extract the surface from the 3D Delaunay triangulation, a min-cut formulation is used to find the inside/outside labeling of the tetrahedra and to deduce the surface as the interface between inside and outside. This graph-cut method has been commonly used in surface reconstruction either from regular space partitions [HK06, LB07] or from data-driven partitions [LPK09]. This method guarantees a hole-free[†] and intersection-free surface, as well as low computation times.

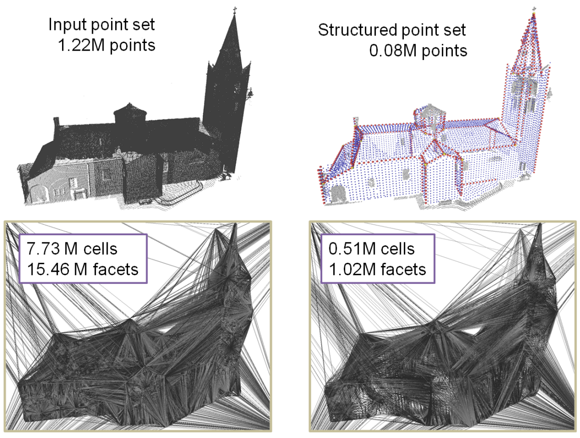


Figure 7: *Delaunay-based space partition. The Delaunay triangulation from the structured point set (right) provides a lighter space partition than from the input point set (left). Note that, in practice, the 8 bounding box points are also inserted in the partition.*

Let us consider a graph $(\mathcal{C},\mathcal{F})$. $\mathcal{C} = \{c_1,...,c_n\}$ is the set the cells (or tetrahedra) induced by the 3D Delaunay triangulation, corresponding to the nodes of the graph. $\mathcal{F} = \{f_1,...,f_m\}$ is the set of triangular facets existing between two cells of the Delaunay triangulation, representing the edges of the graph. A cut in the graph $(\mathcal{C},\mathcal{F})$ consists in separating the set of cells $\mathcal{C}$ in two disjoint sets $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$ such that $\mathcal{C} = \mathcal{C}_{in} + \mathcal{C}_{out}$ and $\mathcal{C}_{in} \cap \mathcal{C}_{out} = \emptyset$. The set of edges between $\mathcal{C}_{in}$ and $\mathcal{C}_{out}$ corresponds to a set of triangular facets forming a surface $\mathcal{S} \subset \mathcal{F}$.

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

$$C(\mathcal{S}) = \sum_{f_i \in \mathcal{S}} a(f_i)\, Q(f_i) + \sum_{c_k \in \mathcal{C}_{in}} P_{out}(c_k) + \sum_{c_k \in \mathcal{C}_{out}} P_{in}(c_k) \tag{2}$$

where $Q(f_i)$ is a non-negative quality function of the facet f_i weighted by its area $a(f_i)$. The product $a(f_i)\, Q(f_i)$ represents the weight put on the edge f_i in the graph. P_{in} and P_{out} are prediction functions penalizing unexpected cell labels according to visibility considerations. They allow the insertion of weights between the nodes of $\mathcal{C}$ and two artificial nodes called the source and the sink so that the optimal surface is not reduced to the trivial empty solution. These functions act as a data term in the conventional energy formulations. The optimal cut minimizing the cost $C(\mathcal{S})$ is obtained using the max-flow algorithm [BK04].

3.2. Visibility prediction

One of the key components of the min-cut formulation consists in efficiently weighting the edges between the nodes of the graph and both the source and the sink. To avoid the shrinking bias, the existing solutions usually rely on inserting a balloon force [LB07] or restricting the configuration space [HK06]. Visibility information are also used [LPK09] where each input point is associated to a line of sight and a visibility confidence coefficient. Both domain restrictions and a force attracting the surface to be close to points with a high visibility confidence are then imposed. This solution is however particularly time-consuming, and the attraction force tends to create spurious bumps on surfaces, especially in the presence of noise.

We propose a visibility-based approach consisting in detecting patches by shooting rays. This solution is both fast and robust to noise and outliers.

Patch detection. Visibility patches are first detected from the input point set. Rays are shot from a user-selected set of sides of the bounding box, which, in practice, corresponds to the scanning directions. The ray directions are chosen as being orthogonal to the selected sides of the box. The spatial density of the points is analyzed in the cylindrical domain of radius R_v along the ray. The user-specified parameter R_v must be large enough to prevent the ray from penetrating inside holes due to missing data. We set as default value $R_v = 2\widehat{d}$ where $\widehat{d}$ denotes the average distance between the 4-nearest neighbors computed from the input point set. This assumes a uniform sampling of the input point set. The first significant peak of the spatial density is detected when the point density becomes greater than $0.5\pi R_v^2 \widehat{d}^{-2}$, i.e., when the density exceeds half the average density estimated in the input point set. A visibility patch is created at the peak location under two conditions: (i) the point distribution is locally planar on this patch, as performed by [KTB07], and (ii) the angle between the ray and the local plane is not too large (in practice, we limit this angle to 45°).

Label prediction. The visibility patches and their associated rays are then embedded into the 3D Delaunay triangulation. Each Delaunay tetrahedron crossed by one of the visibility rays is potentially predicted as inside or outside according to its position with respect to the visibility patch, as

[†] The obtained surface is either boundary-free, or is composed of surface components with a single boundary which coincides with the graph boundary

illustrated by Fig. 8. An uncertainty margin M_v is computed so that the cells located close to the visibility patch are not predicted. The cells located before M_v are predicted as outside, whereas the cells located after M_v in a depth tolerance d_v are predicted as inside. In our experiments, M_v and d_v are fixed such that $M_v = 2d_v = \max(R_v, 2\varepsilon)$.

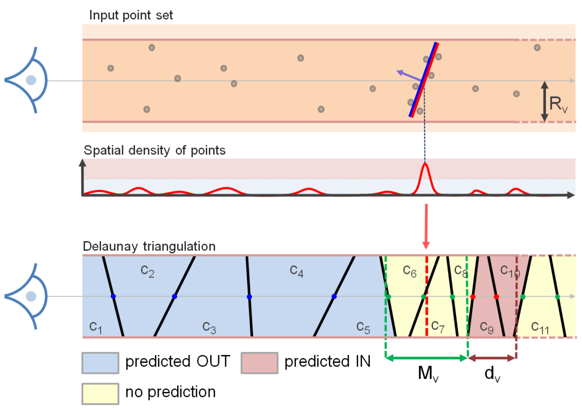


Figure 8: *Visibility prediction. Rays are shot from the input point set (top). A visibility patch is detected at the first high point density (middle) detected along the ray (under two additional conditions mentioned in the text). By embedding the visibility patches and their rays in the triangulation, a label prediction is assigned to the cells crossed by the rays according to their position with respect to the patch, an uncertainty margin M_v, and a depth tolerance d_v (bottom). Cells c_1 to c_5 (respectively c_9 and c_{10}) are predicted as outside (resp. inside). The other cells have no prediction.*

By denoting $\mathcal{P}_{out}$ (resp. $\mathcal{P}_{in}$) the set of cells predicted as *outside* (resp. *inside*) over all detected visibility patches, the prediction functions can be formulated as

$$\left\{ \begin{array}{l} P_{out}(c_k) = \beta \cdot 1_{\{c_k \in \mathcal{P}_{out}\}} \\ P_{in}(c_k) = \beta \cdot 1_{\{c_k \in \mathcal{P}_{in}\}} \end{array} \right. \tag{3}$$

where $1_{.}$ denotes the characteristic function, and β denotes a positive value greater than 1. Note that we impose $\mathcal{P}_{out} \cap \mathcal{P}_{in} = \emptyset$. In other words, if a cell is predicted as both outside and inside by two different visibility patches, we reject the cell from both $\mathcal{P}_{out}$ and $\mathcal{P}_{in}$. This operation allows reducing the prediction errors. As illustrated by Fig. 9, the visibility patches are correctly detected, even in presence of a significant amount of noise and outliers.

3.3. Surface quality

Function Q measures the quality of a facet by taking into account its structural meaning and its geometry. As each vertex of a facet is characterized by a structural type, i.e., *planar, crease, corner* or *clutter*, and is potentially associated to one or more planar primitives, the plausibility of a facet can be checked by combining the information of its three vertices. Different groups of configurations can be distinguished:

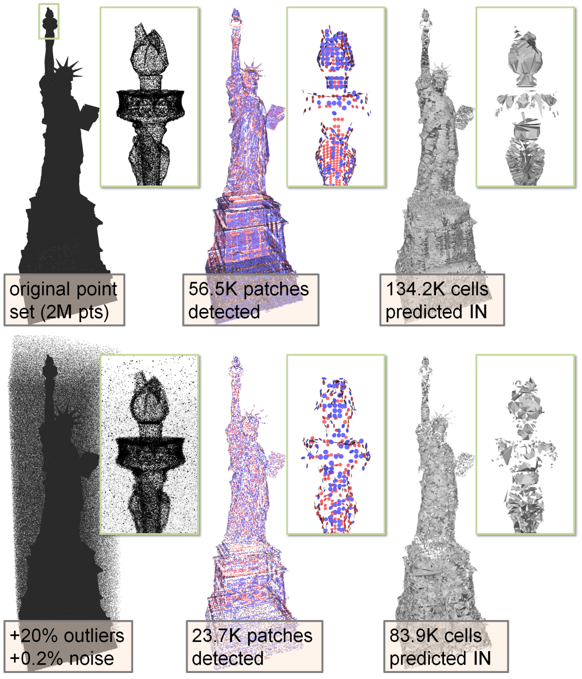

original point set (2M pts)
56.5K patches detected
134.2K cells predicted IN
+20% outliers +0.2% noise
23.7K patches detected
83.9K cells predicted IN

Figure 9: *Visibility patches are represented by bi-colored disks (blue for outside, red for inside). The visibility patch detection is more selective in the presence of defect laden data. The number of cells predicted as IN remains high even with a non-negligible amount of outliers and noise.*

- *Structurally-coherent* facets, denoted by *S-coherent* facets, have their three vertices in coherence with the structural relations induced during the point set structuring. As illustrated by Fig. 10, it corresponds to facets describing planar components, creases, and corners.
- *Free-Form-coherent* facets, denoted by *FF-coherent* facets, are plausible in the scene as a portion of a free-form shape. The type of their vertices is *clutter*, and potentially *planar* when the facet is used to connect a free-form part to a structure.
- *Incoherent* facets constitute all the remaining cases, i.e., all facets neither *S-coherent* nor *FF-coherent*. These facets may break the structures, for example, by linking two *planar* vertices associated to different primitives.

Function Q is then expressed as a penalization score of the form:

$$Q(f_i) = \left\{ \begin{array}{ll} 0 & \text{if } f_i \text{ } \textit{S-coherent} \\ g(f_i) & \text{if } f_i \text{ } \textit{FF-coherent} \\ \gamma & \text{if } f_i \text{ } \textit{incoherent} \end{array} \right. \tag{4}$$

where γ is a positive value greater than 1 so that *incoherent* facets are highly penalized. $g(f_i)$ is a function measuring the geometric quality of a *free-form-coherent* facet. It is defined by using the β-skeleton criterion [ABE98]. The principle consists in favoring the facets incident to tetrahedra with

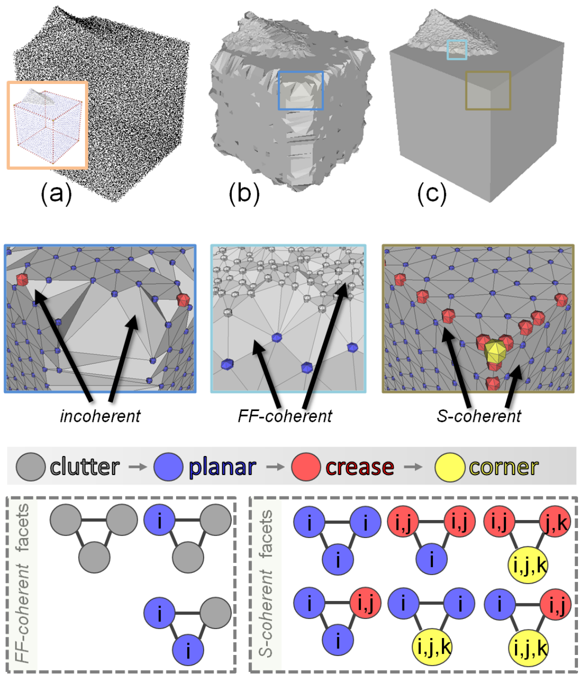


Figure 10: *Facet coherence. Starting from a noisy input point set (a, depicted with its structured point set in the closeup), we reconstruct two models: one with many incoherent facets (b), and one with all facets either FF-coherent or S-coherent (c). i, j and k each corresponds to a primitive index in the FF-coherent and S-coherent facet description.*

large circumspheres. $g(f_i)$ is formulated as

$$g(f_i) = \alpha - \min(\cos(\phi_{in}), \cos(\phi_{out})) \tag{5}$$

where ϕ_{in} (respectively ϕ_{out}) is the angle between the plane supporting f_i and the tangent of the inside (respectively outside) circumsphere of the inside (respectively outside) at the intersection with f_i. α is a parameter belonging to $[1,2]$ and relating to the smoothness of the free-form shapes.
By scoring the *S-coherent* facets to zero, we assume that the structural components extracted during the data structuring are very plausible, and should be part of the surface. *FF-coherent* facets cannot benefit from such a favor as most of them are geometrically irrelevant.

3.4. Model parameters

The general cost function C defined in Eq. 2 includes three parameters α, β, and γ. β and γ have similar roles by strongly penalizing unexpected configurations, i.e., the cells not labeled in accordance to the visibility prediction for β, and *incoherent* facets for γ. In our experiments β is set to 10^5 such that the visibility prediction becomes a hard constraint. When choosing to be more flexible with the visibility prediction by attributing a much lower value, we may correct some potential prediction errors due to good local geometry of the facets. However such choice may favor solutions which skip small components of the objects. γ is also set to a high value (10^3 in all experiments shown) so as to prevent shrinking the surface around sharp creases. Conversely, α is a soft parameter set to 1 by default. It is used to trade smoothness (and robustness in presence of noise) for faithfulness.

3.5. Surface simplification with structure preserving

The complexity of the obtained hybrid surface depends on both the free-form/canonical area ratio and the tolerance parameter ε fixing the size of the *Structurally-coherent* facets. However the surface can be easily simplified by exploiting the semantic information of the vertices. An edge-collapse procedure is then specified by attributing either an edge length based cost to the edges connecting two identical planar or crease vertices, the created vertex being determined by the edge mid-point, or an infinite cost value to the other edges. As illustrated by Fig. 11, this procedure allows us to reduce the complexity without any loss of accuracy as the free-form components are preserved.

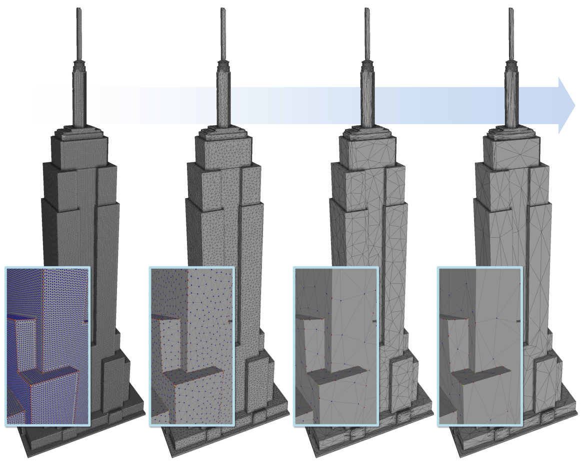


Figure 11: *Surface simplification. Our edge collapse procedure exploits the structural type of vertices so that the surface is simplified without loss of accuracy (see close-ups).*

4. Experiments

The algorithm has been implemented in C++, using the Computational Geometry Algorithms Library [CGA12] which provides the basic geometric tools for the analysis of point clouds and for 3D Delaunay triangulation. The planar primitives are extracted using either a RANSAC implementation [SWK07] on defect-laden point sets, or an efficient region growing procedure [LM12] on noise-free point sets.

Flexibility. The algorithm has been tested on a variety of data ranging from simple objects (*Distorted cube*, Fig. 10)

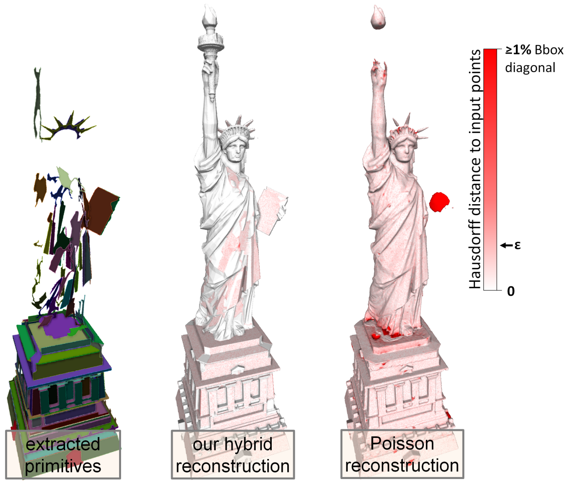


Figure 12: *Statue of Liberty. The extracted primitives mainly corresponds to planar components of the basement of the statue and to some parts of the dress. Our algorithm structures these elements while accurately reconstructing the free-form parts such as the face. The Hausdorff distance to the input points is nearly zero at free-form locations, and is lower than ε otherwise. The Hausdorff distance measured over the Poisson reconstructed surface [KBH06] (octree depth=10) is higher almost everywhere.*

to large urban scenes (Fig. 16) through complex buildings (*Church* and *Empire State*). Different types of acquisition systems have been used to generate the input point sets, ranging from terrestrial laser to MVS imagery through airborne Lidar. The hybrid surface we produce brings flexibility and allows the modeling of urban scenes in a general context. As illustrated by Fig. 16, the details such as facade ornaments or roof superstructures (chimneys, dormer-windows..) are captured whereas the principal structure of the scene is recovered under the user-specified tolerance ε. Even if the algorithm is devoted to reconstruct urban scenes and cultural heritage buildings, it is also suited to reconstruct mechanical parts, by hybrid surfaces combining canonical as well as free-form parts. On models such as the *blade* comprising semi-sharp creases, the reconstructed surface may contain either truly sharp creases for high error tolerance or planar parts connected by smooth free-form parts for low error tolerances (Fig. 15).

Robustness. The algorithm is designed to generate consistent results even in case of defect-laden primitive detection. Through the hybrid aspect, the reconstructed surface remains coherent even in high under-detection situations where only a few primitives are detected. In presence of noise, the canonical parts are nicely preserved, albeit the free-form parts are hampered with noise. In particular, the main planar components, *i.e.* the walls, are structured on *MVS2* and *MVS3* models in Fig. 16 whereas the details above the tolerance ε, *i.e.* the ornaments, are recovered.

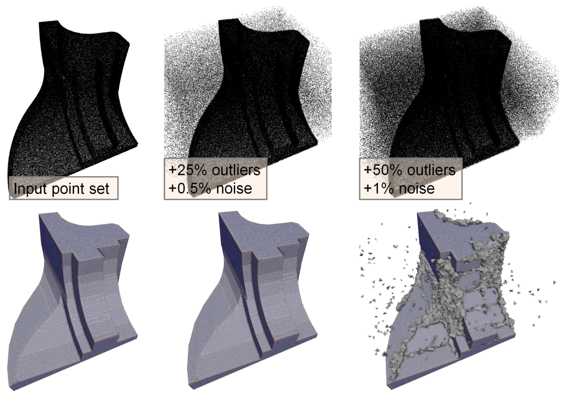


Figure 13: *Fandisk. Even if our solution is dedicated to urban reconstruction, it is also suited to model mechanical objects, assuming the curve parts can be approximated by sets of planes. The method provides noise and outlier robustness through the robust primitive extraction strategy.*

Note that these two models are very challenging in our context as they are usually dedicated to reconstruction methods exploiting photo-consistency information. Noise robustness usually comes at the price of selecting a high ε value making the model more simplified. The algorithm also performs well in presence of outliers and heterogeneous point densities (see Fig. 14) without, however, being able to ignore a huge amount of outliers (see Fig. 13, right case).

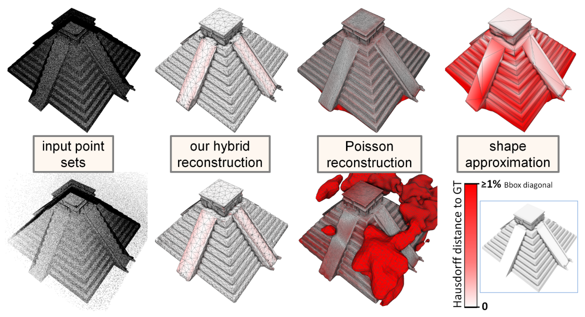


Figure 14: *Maya temple. A defect-free (top) and a defect-laden (bottom, outliers +10%, variable point density ranging from a factor 1 to 50 from left to right) point sets are generated from a mesh (bottom right). The obtained hybrid surfaces are close to the original mesh in terms of accuracy and structure. The Poisson surfaces are less accurate, especially from the defect-laden input, and have no structure. The use of a shape recovery algorithm [CSAD04] from the Poisson surface provides a rough structured mesh while, however, amplifying the geometric error to Ground Truth.*

Performances. Running times are provided in Fig. 16 and 15 on two models as a function of ε. Our algorithm takes an order of 30 seconds to reconstruct a surface from one million points with a high ε-tolerance, after consuming few seconds to extract the planar primitives. The surface extraction step is often both more time and memory consuming than the structuring step. The running times strongly depend on the structuring level of the input point set. For instance, the *Church* model in its highly structured version is obtained six times faster than without structuring it, and requires five times less memory. The performances from the airborne Lidar scan are lower due to the disproportion between the low point density ($< 2\text{pts/m}^2$) and the large scene area (1km^2), which makes the structuring relatively inefficient.

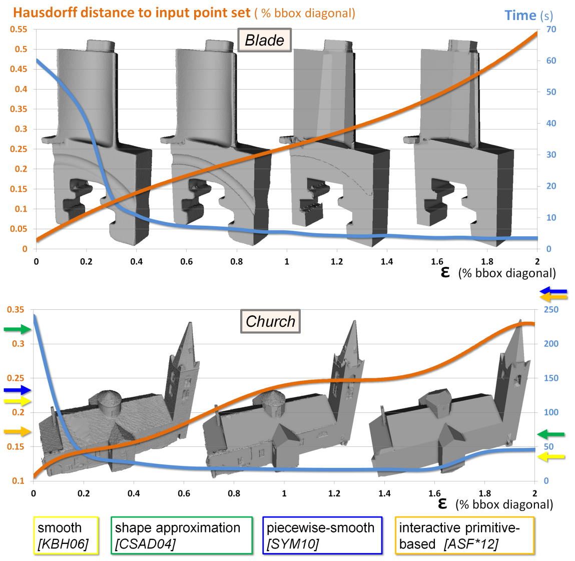


Figure 15: *Impact of* ε*. When* ε *is close to zero, few primitives are detected; the obtained surfaces being similar to smooth reconstructions. Increasing* ε *progressively structures the surface while preserving the details over an* ε *tolerance. The computation time decreases as the structured point set becomes lighter. For smooth shapes, i.e. Blade, the Hausdorff distance to the input point set increases relatively proportionally to* ε*. In case of urban scenes, i.e. Church, the distance evolves in function of the different urban scales, the stagnation around* $\varepsilon = 1$ *occurring after that the minor elements as windows or doors have been digested in the major components as walls or roof sections. The hybrid model at* $\varepsilon = 0.2$ *is particularly interesting, competing well with existing approaches in terms of accuracy and running time (see colored arrows).*

Limitations. Our algorithm is not designed to reconstruct non-manifold surfaces nor surfaces with complex occlusions or invisible parts from the scanning directions. Our algorithm is not suited to missing data (large holes in the input point set) where the reconstruction problem is even more ill-posed. The two main reasons are as follows. First, the Delaunay-based space partitioning does not allow an optimal or smooth interpolation of the surface on holes. Second, the parameter R_v must be high to avoid that visibility rays penetrate inside holes. Increasing R_v however comes at a cost as the set of predicted inside/ outside cells are then less consistent, which in turn may skip some small parts of the surface.

5. Conclusion

We described an algorithm for reconstructing a surface from a raw point set and a set of detected planar primitives which may cover only a subset of the input points. Our approach first proceeds by extracting structuring relationships between the primitives in order to identify sharp features and adjacencies. It then consolidates the point set on planar areas by re-sampling the detected primitives and sharp features, and constructs a 3D Delaunay triangulation from both planar and free-form areas. The final reconstructed surface is obtained through a graph-cut method formulated on the graph of the triangulation enriched with attributes derived from the previous structuring step. In addition to being both feature-preserving and efficient, the main strength of our approach is its robustness to imperfect primitive detection: the reconstructed surfaces gracefully range from perfectly canonical for perfect primitive detection to purely free-form surfaces in absence of primitive detection. Using a hybrid representations allow mixing canonical components and free-form parts so as to improve the structure without loosing the fine details on the free-form areas.

In its current form the user-specified error tolerance provides a way to vary the level of details, but we wish to research on ways to construct a scale-space with smooth transitions between the scales. We also wish to extend this approach in order to deal with heterogeneous data such as triangle soups and slices, in addition to point sets.

Acknowledgments. This work was funded by the European Research Council (ERC Starting Grant "Robust Geometry Processing", Grant agreement 257474). The authors thank the French Mapping Agency (IGN), the AIM@Shape project, M. Arikan, C. Strecha, M. Yvinec, R. Keriven, L. De Luca and Y. Verdié for providing datasets and 3D-models.

References

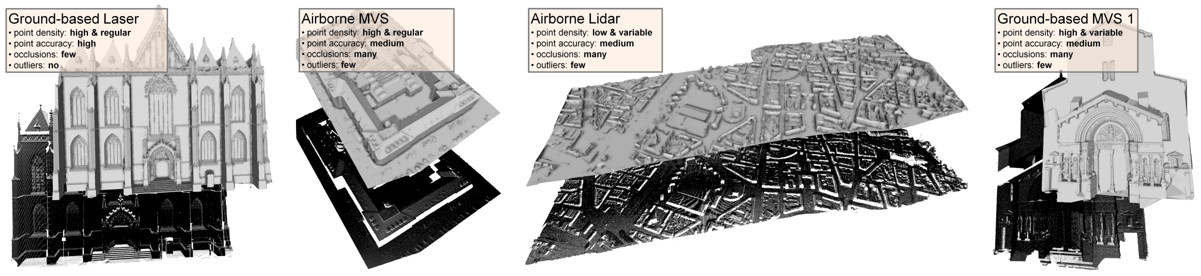
[ABE98] AMENTA N., BERN M., EPPSTEIN D.: The crust and the beta-skeleton: combinatorial curve reconstruction. *Graphical Models and Image Processing 60*, 2 (1998). 6

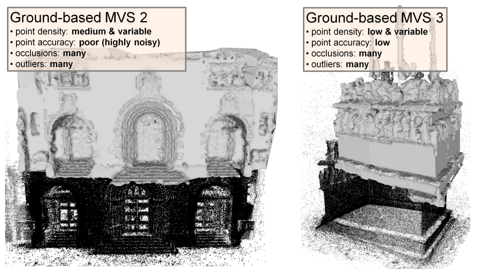
[ABK98] AMENTA N., BERN M., KAMVYSSELIS M.: Crust: A new Voronoi-based surface reconstruction algorithm. In *SIGGRAPH* (1998). 2

[AP10] ATTENE M., PATANE G.: Hierarchical structure recovery of point- sampled surfaces. *CGF 29*, 6 (2010). 2

[ASF*12] ARIKAN M., SCHWARZLER M., FLORY S., WIMMER M., MAIERHOFER S.: O-snap: Optimization-based snapping for modeling architecture. *TOG* (2012). 2

[BK04] BOYKOV Y., KOLMOGOROV V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI 26*, 9 (2004). 5





	Ground Laser	*Airborne MVS*	*Airborne Lidar*	*Ground MVS 1*	*Ground MVS 2*	*Ground MVS 3*	*Distorted cube*	*Empire State*
# input points	1.49M	1.97M	1.69M	2.43M	0.57M	0.75M	50K	1.2M
# primitives	166	154	3257	55	4	18	6	114
# structured points	1.19M	0.24M	0.87M	1.26M	0.36M	0.39M	5.4K	0.15M
Structuring time (sec)	19	15.4	51.6	18.9	3.4	4.03	0.25	13.5
extraction time (sec)	216	31.6	353.1	434	40.3	41.2	1.0	71.3
Memory peak (Mb)	3,300	1,120	2,220	3,700	990	1,080	29	715

Figure 16: *Reconstruction of urban scenes from different acquisition systems. The obtained hybrid surfaces recover the main structure of the scenes while preserving the details. The performances in terms of running time and memory are given in the bottom right table. The tests have been performed on an Intel Core i7 clocked at 2GHz. We exclude from the running times data loading and computation of the Riemannian graph.*

[CC08] Chen J., Chen B.: Architectural modeling from sparsely scanned range data. *IJCV 78*, 2-3 (2008). 2, 3

[CDR07] Cheng S.-W., Dey T., Ramos E.: Delaunay refinement for piecewise smooth complexes. In *SODA* (2007). 3

[CGA12] CGAL:. Computational Geometry Algorithms Library (www.cgal.org), 2012. 7

[CLP10] Chauve A.-L., Labatut P., Pons J.-P.: Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *CVPR* (San Francisco, US, 2010). 2

[CSAD04] Cohen-Steiner D., Alliez P., Desbrun M.: Variational shape approximation. In *SIGGRAPH* (2004). 2, 8

[FCOS05] Fleishman S., Cohen-Or D., Silva C.: Robust moving least-squares fitting with sharp features. In *SIGGRAPH* (2005). 2

[GSH*07] Gal R., Shamir A., Hassner T., Pauly M., Cohen-Or D.: Surface reconstruction using local shape priors. In *SGP* (2007). 2

[HK06] Hornung A., Kobbelt L.: Robust reconstruction of watertight 3D models from non-uniformly sampled point clouds without normal information. In *SGP* (2006). 1, 5

[KBH06] Kazhdan M., Bolitho M., Hoppe H.: Poisson surface reconstruction. In *SGP* (2006). 1, 8

[KTB07] Katz S., Tal A., Basri R.: Direct visibility of point sets. In *SIGGRAPH* (2007). 5

[LB07] Lempitsky V., Boykov Y.: Global optimization for shape fitting. In *CVPR* (2007). 1, 5

[LKBV13] Lafarge F., Keriven R., Bredif M., Vu H.: A hybrid multi-view stereo algorithm for modeling urban scenes. *PAMI 35*, 1 (2013). 2

[LM12] Lafarge F., Mallet C.: Creating large-scale city models from 3d-point clouds: a robust approach with hybrid representation. *IJCV 99*, 1 (2012). 2, 3, 7

[LPK09] Labatut P., Pons J.-P., Keriven R.: Robust and efficient surface reconstruction from range data. *CGF 28*, 8 (2009). 2, 5

[LWC*11] Li Y., Wu X., Chrysathou Y., Sharf A., Cohen-Or D., Mitra N. J.: Globfit: Consistently fitting primitives by discovering global relations. In *SIGGRAPH* (2011). 2

[MPWC12] Mitra N., Pauly M., Wand M., Ceylan D.: Symmetry in 3d geometry: Extraction and applications. In *EUROGRAPHICS STARs* (2012). 2

[MWA*12] Musialski P., Wonka P., Aliaga D., Van Gool L., Purgathofer W.: A survey of urban reconstruction. In *EUROGRAPHICS STARs* (2012). 1

[SDK09] Schnabel R., Degener P., Klein R.: Completion and reconstruction with primitive shapes. In *Eurographics* (2009). 2

[SWK07] Schnabel R., Wahl R., Klein R.: Efficient RANSAC for point-cloud shape detection. *CGF 26*, 2 (2007). 2, 3, 7

[SYM10] Salman N., Yvinec M., Merigot Q.: Feature preserving mesh generation from 3D point clouds. In *SGP* (2010). 2

[VAB12] Vanegas C., Aliaga D., Benes B.: Automatic extraction of manhattan-world building masses from 3d laser range scans. *T-VCG 18*, 10 (2012). 2

[ZSW*10] Zheng Q., Sharf A., Wan G., Li Y., Mitra N. J., Cohen-Or D., Chen B.: Non-local scan consolidation for 3D urban scenes. In *SIGGRAPH* (2010). 2