PolyFit: Polygonal Surface Reconstruction from Point Clouds

Liangliang Nan
Visual Computing Center, KAUST
liangliang.nan@gmail.com

Peter Wonka
Visual Computing Center, KAUST
pwonka@gmail.com

Abstract

We propose a novel framework for reconstructing lightweight polygonal surfaces from point clouds[1]*. Unlike traditional methods that focus on either extracting good geometric primitives or obtaining proper arrangements of primitives, the emphasis of this work lies in intersecting the primitives (planes only) and seeking for an appropriate combination of them to obtain a manifold polygonal surface model without boundary.*

We show that reconstruction from point clouds can be cast as a binary labeling problem. Our method is based on a hypothesizing and selection strategy. We first generate a reasonably large set of face candidates by intersecting the extracted planar primitives. Then an optimal subset of the candidate faces is selected through optimization. Our optimization is based on a binary linear programming formulation under hard constraints that enforce the final polygonal surface model to be manifold and watertight. Experiments on point clouds from various sources demonstrate that our method can generate lightweight polygonal surface models of arbitrary piecewise planar objects. Besides, our method is capable of recovering sharp features and is robust to noise, outliers, and missing data.

1. Introduction

Reconstructing 3D models from sampled points has been a major problem in both computer vision and computer graphics. Although it has been extensively researched in the past few decades [8, 13, 22, 11, 3, 17, 27, 25, 12, 2, 15, 20, 26, 16], obtaining faithful reconstructions of real-world objects from unavoidable noisy and possibly incomplete point clouds remains an open problem.

In this work, we focus on reconstructing piecewise planar objects (i.e., man-made objects such as buildings). Our inputs are point clouds sampled from real-world objects that could be captured by various means (e.g., drones, handheld scanners, and depth cameras). Our goal is to achieve lightweight polygonal surface models of the objects.

We consider a method to be effective to this problem if it meets the following requirements. First, the reconstructed model should be an oriented 2-manifold and watertight, ensuring physically valid models. This is essential for many applications, e.g., simulation and fabrication. Second, it should be able to recover sharp features of the objects. Third, the method should not closely follow surface details due to imperfections (i.e., noise, outliers, and missing data). Instead, a lightweight representation is more preferred and beneficial for many applications (e.g., Augmented Reality/Virtual Reality) that may involve many objects and large scenes. Besides, it is also helpful to provide a way to control the detail levels of the reconstructed models within the reconstruction pipeline. We proposed a novel reconstruction framework that all the above requirements are satisfied in a single optimization process.

Instead of fitting dense smooth polygonal surfaces [8, 13, 22, 11], we seek for a more compact representation (i.e., simple polygonal surfaces) for piecewise planar objects. Our method is based on a *hypothesizing and selection* strategy. Specifically, we chose an optimal set of faces from a large number of candidate faces to assemble a compact polygonal surface model. The optimal set of faces is selected through optimization that encourages good point support and compactness of the final model. The manifold and watertight requirements are enforced as hard constraints. Figure 1 shows an example of our reconstruction. Our work makes the following contributions:

- we cast polygonal surface reconstruction from point clouds as a binary labeling problem based on a *hypothesizing and selection* strategy.
- a surface reconstruction framework that is dedicated to reconstructing piecewise planar objects.
- a binary linear programming formulation for face selection that guarantees lightweight, manifold, and watertight reconstructions and meanwhile recovers sharp features of the objects.

[1] The source code of PolyFit is available under https://github.com/LiangliangNan/PolyFit.

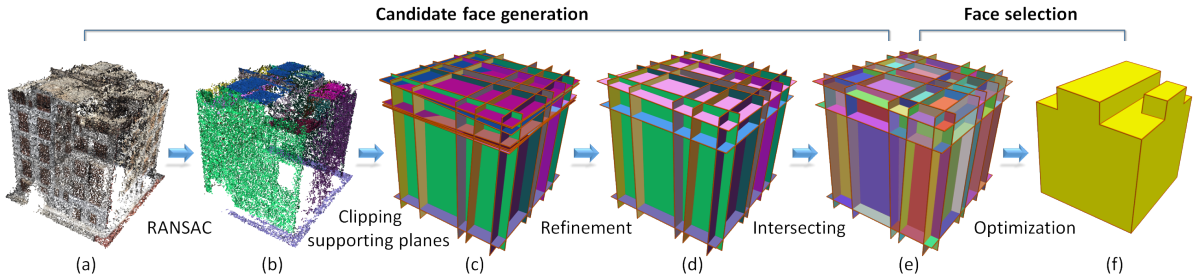

Figure 1. Pipeline. (a) Input point cloud. (b) Planar segments. (c) Supporting planes of the initial planar segments. (d) Supporting planes of the refined planar segments. (e) Candidate faces. (f) Reconstructed model. All planar segments and faces are randomly colored.

2. Related Work

In literature, a large body of research on polygonal surface reconstruction from point clouds aim at obtaining dense polygonal surfaces [8, 13, 22, 11]. In the last decades, extracting geometric primitives and identifying their combination to infer higher-level structures have become the most popular technique for reconstructing piecewise planar objects. In this section, we mainly review approaches that focus on primitive extraction, primitive regularization, and primitive- and hypothesis-based reconstruction.

Primitive extraction. Researches falling in this category aim at extracting high-quality instances of basic geometric primitives (e.g., plane, cylinder) from point clouds corrupted by noise and outliers. The common practice for this particular task is the Random Sample Consensus (RANSAC) algorithm [6] and its variants [28, 24, 14]. These methods are reliable when the input data is contaminated by noise and outliers. So in our work, we use an efficient implementation of the RANSAC algorithm [24] to extract initial planar primitives.

Primitive regularization. By exploiting the prior knowledge about the structure of an object, researchers further regularize the extracted primitives. Li et.al. [17] discover global mutual relations between basic primitives and use such information as constraints to refine the initial primitives base on local fitting and constrained optimization. Monszpart et. al. [19] further exploit this idea to extract regular arrangements of planes. These methods focus on inferring and regularizing potential relationship between structures. However, obtaining manifold and watertight surface models from the regularized primitives remains unsolved. In our work, we provide a promising framework to this challenging problem based on a *hypothesizing and selection* strategy.

Primitive-based reconstruction. In contrast with approaches that focus on obtaining dense polygonal surfaces, exploiting high-level primitives for man-made objects became popular in the last years [25, 12, 15, 20, 26, 16]. Arikan et al. [1] proposed an optimization-based interactive tool that can reconstruct architectural models from a sparse point cloud. Lin et al. [18] reconstruct coarse building models by decomposing and fitting a set of piecewise building blocks to the point clouds. By structuring and resampling planar primitives, Lafarge and Alliez [12] consolidate the point clouds and then obtain surface models by solving a graph-cut problem. Using a min-cut formulation, Verdie et. al. [26] reconstruct watertight buildings from an initial arrangement of planar segments. These approaches specialize in reconstructing urban buildings. Thus, they may not be suitable to handle general piecewise planar objects. Based on space partitioning and volumetric presentations, [3] and [2] obtain piecewise planar reconstruction by determining if cells are occupied or not. These two techniques require visibility information from the acquisition process as input while our approach does not require this information.

Hypothesis-based reconstruction. By making stronger assumptions on the objects, researchers further regularize the reconstruction problem and fit compound shapes (i.e., a combination of multiple basic primitives) to the point clouds [9, 16]. In Li et.al. [16], a set of axis-aligned boxes is assembled to approximate the geometry of a building. Their strategy is generating a non-uniform grid and then selecting a subset of its cells that have good data support and are smooth. In our work, we generalize this idea to reconstruct general piecewise planar objects, and our reconstruction is based on optimization under hard constraints that guarantee manifold and watertight polygonal surface models.

3. Overview

Our method takes as input a point cloud (possibly noisy, incomplete, and with outliers) of a piecewise planar object and outputs a lightweight polygonal surface model. Our method consists of two main parts (see Figure 1):

Candidate face generation. We first extract a set of planar segments from the point cloud using RANSAC [24]. Considering the detected planar segments may contain undesired elements due to noise, outliers, and missing data, we refine these planar segments by iteratively merging plane pairs and fitting new planes. We then clip these

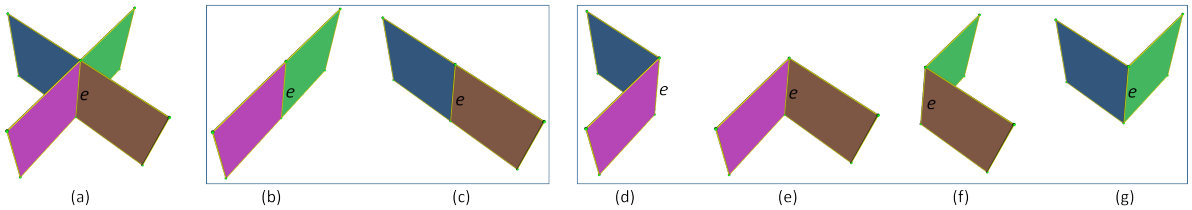

Figure 2. Two planes intersect with each other resulting in four parts (a). (b) to (g) show all possible combinations to ensure a 2-manifold surface. The edge in (b) and (c) connects two co-planar parts, while in (d) to (g) it introduces sharp edges in the final model.

planes by an enlarged bounding box of the point cloud and compute pairwise intersections of the clipped planes to hypothesize of the object's faces.

Face selection. We choose an optimal subset of the candidate faces to assemble a manifold and watertight polygonal surface model. To do so, we formulate the face selection as a binary linear programming problem. Our objective function combines three terms that favor data fitting, point coverage, and model complexity, respectively. We also formulate hard constraints that ensure the final model is manifold and watertight.

4. Candidate Face Generation

The input to this step is a point cloud P of a piecewise planar object, and the goal is to generate a reasonably large number of candidate faces.

Plane extraction. We use the RANSAC-based primitive detection method proposed by Schnabel et al. [24] to detect a set of initial planar segments $S = \{s_i\}$ from the point cloud P. Here s_i is a set of points whose distances are smaller than a threshold ε to a plane and each points can be assigned to no more than one plane. This plane is denoted as the *supporting plane* of s_i.

Plane refinement. Due to the presence of noise and outliers (especially for point clouds computed from Multi-view Stereo), RANSAC may produce a few undesired planar segments. We observed that the undesired planar segments usually have arbitrary orientations and poor point support. Although the later optimization-based face selection procedure is designed to be capable of handling such outliers, there may still cause problems. First, these arbitrarily oriented planes may result in long but very thin faces and small bumps in the final model. In some extreme cases, it may also introduce degenerate faces due to the limit of floating point precision. This degeneracy usually makes the *manifold* and *watertight* hard constraints (see Section 5.2) impossible to be satisfied. Second, the undesired candidate faces results in a larger optimization problem that is more expensive to be solved.

To tackle this issue, we iteratively refine the initial planar segments using an improved plane refinement algorithm proposed in [15]. Specifically, we first compute the angle of the supporting planes for each pair of planar segments. Then, starting from the pair (s_i, s_j) with the smallest angle, we test if the following two conditions are met. First, the angle between the two planes is lower than a threshold, i.e., $angle(s_i, s_j) < \theta_t$. Second, more than a specified number (denoted as N_t) of points lie on the supporting planes of both segments. If both conditions are satisfied, we merge the two planar segments and fit a new supporting plane using PCA [10]. We iterate this process until no more segment pair can be merged. In our experiments, we choose $\theta_t = 10°$ and $N_t = min(|s_i|, |s_j|)/5$, where $|s_i|$ denotes the number of supporting points of s_i. Figure 1 (c) and (d) show the extracted planes before and after refinement respectively. It should be noted that an alternative approach, e.g., [17], can also be used to extract planar segments.

Pairwise intersecting. To hypothesize the object's faces, we crop the supporting planes of all planar segments by the bounding box of the point cloud. Then, the candidate faces (i.e., a superset of the faces of the object) can be obtained by intersecting the cropped planes. For simplicity, we compute pairwise intersections of the cropped planes (see Figure 1 (e)). It should be noted that pairwise intersections introduce redundant candidate faces. Since most of the redundant faces do not represent actual structures of an object, they are supported by no or very few (due to noise and outliers) point samples. The subsequent optimization-based face selection is designed to favor choosing the most confident faces while satisfying certain constraints.

The pairwise intersections maintain incidence information of the faces and edges. Each edge of a candidate face is either connecting four neighboring candidate faces or representing a boundary. For example in Figure 2 (a), edge e connects four faces while others are boundaries. We rely on such incidence information to formulate our *manifold* and *watertight* constraints for face selection (see Section 5.2).

5. Face Selection

Given N candidate faces $F = \{f_i | 1 \leq i \leq N\}$ generated in the previous step, we select a subset of these candidate faces that can best describe the geometry of the object and ensure that the chosen faces form a manifold and watertight polygonal surface. This is achieved through op-

timization. We define multiple energy terms that constitute our objective function.

5.1. Energy terms

Let variable x_i encode if a candidate face f_i is chosen (i.e., $x_i = 1$) or not chosen (i.e., $x_i = 0$), our objective function consists of three energy terms: data-fitting, model complexity, and point coverage.

Data-fitting. This term is intended to evaluate the fitting quality of the faces to the point cloud while accounting for an appropriate notion of confidence [21]. It is defined to measure a confidence-weighted percentage of points that do not contribute to the final reconstruction

$$E_f = 1 - \frac{1}{|P|}\sum_{i=1}^{N} x_i \cdot support(f_i), \tag{1}$$

where $|P|$ is the total number of points in P. $support(f_i)$ accounts for a notion of confidence and is defined at each point considering its local neighborhood

$$support(f) = \sum_{p,f|dist(p,f)<\varepsilon} (1 - \frac{dist(p,f)}{\varepsilon}) \cdot conf(p), \tag{2}$$

where $dist(p, f)$ measures the Euclidean distance between a point p and a face f. We consider only points that are ε-close to f, i.e., points p satisfying $dist(p, f) < \varepsilon$. The confidence term $conf(p)$ measures the local quality of the point cloud at a point p. It is computed by examining the local covariance matrices defined at p, as in [23]. In this work, we compute for p the covariance matrices of its local neighbors at three static scales (i.e., different neighborhood sizes). Then, $conf(p)$ is defined as

$$conf(p) = \frac{1}{3}\sum_{i=1}^{3}(1 - \frac{3\lambda_i^1}{\lambda_i^1 + \lambda_i^2 + \lambda_i^3}) \cdot \frac{\lambda_i^2}{\lambda_i^3}, \tag{3}$$

where $\lambda_i^1 \leq \lambda_i^2 \leq \lambda_i^3$ are the three eigenvalues of the covariance matrix at scale i. $conf(p)$ encode two geometric properties of the point samples near point p. The first property $1 - 3\lambda^1/(\lambda^1 + \lambda^2 + \lambda^3)$ evaluates the quality of fitting a local tangent plane at p, whose value of 0 indicates the worst point distribution and value of 1 indicates a perfectly fitted plane. The second property λ^2/λ^3 evaluates the local sampling uniformity. Each eigenvalue ratio takes on a value in the range $[0, 1]$ with boundary values 0 and 1 corresponding to a perfect line distribution and a uniform disc distribution correspondingly.

Intuitively, the *data-fitting* term favors choosing faces that are close to the input points and are supported by densely sampled uniform regions. This term has a value in the range [0, 1] where a value of 1 indicates a noise-free and outlier-free input data.

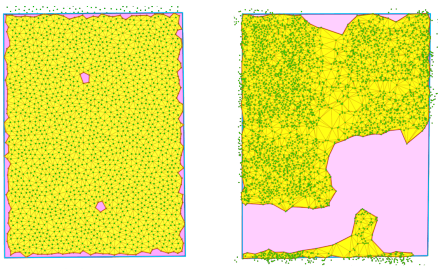

Figure 3. Two examples of point coverage. The α-shape meshes are in yellow and the candidate faces are in purple. The value below each figure indicates the coverage ratio of each face.

Model complexity. Given incomplete point clouds due to occlusions, the *data-fitting* term defined in Equation 1 tends to stubbornly comply with the incomplete data, resulting in gaps in the final model. Besides, noise and outliers also tend to introduce gaps and protrusions in the reconstructed models. To avoid these defects, we introduce a *model complexity* term to encourage simple structures (i.e., large planar regions). Considering gaps and protrusions result in additional sharp edges, we define the *model complexity* term as the ratio of sharp edges in the model

$$E_m = \frac{1}{|E|}\sum_{i=1}^{|E|} corner(e_i), \tag{4}$$

where $|E|$ denotes the total number of pairwise intersections in the candidate face set. $corner(e_i)$ is an indicator function whose value is determined by the configuration of the two selected faces connected by an edge e_i. The intersecting edges can be either planar or sharp. For example, the intersecting edges e in Figure 2 (b) and (c) are planar edges yielding larger planar polygons, but edges in (d) to (g) are sharp edges that will introduce sharp features (if they are selected in the final model). So $corner(e_i)$ will have a value of 1 if the faces associated with e_i introduce a sharp edge in the final model. Otherwise, $corner(e_i)$ has a zero value meaning the two faces are coplanar.

Point coverage. To handle missing data caused by occlusions, the unsupported regions (i.e., regions not covered by points) of the final model should be as small as possible. To measure the coverage of a face f_i, we first project all ε-close points onto f_i. We then extract a mesh M_i^α by constructing a 2D α-shape [5] from the projected points (see Figure 3). We use only the points whose projections are inside f_i for α-shape construction. We call M_i^α an α-shape mesh. Intuitively, the α-shape mesh of a set of points ensures that any three points with a circumradius $r_c \leq \sqrt{\alpha}$ are spanned by a triangle face. Given an appropriate value of r_c, the area of the α-shape mesh surface can provide us a reliable measure of the coverage of a candidate face by the

input points. Thus, our *point coverage* energy is defined as the ratio of the uncovered regions in the model

$$E_c = \frac{1}{area(M)} \sum_{i=1}^{N} x_i \cdot (area(f_i) - area(M_i^\alpha)), \quad (5)$$

where $area(M)$, $area(f_i)$, and $area(M_i^\alpha)$ denote the surface areas of the final model, a candidate face f_i, and the α-shape mesh M_i^α of f_i, respectively. In our implementation, we chose the radius r_c to be $5 \cdot density(P)$, where $density(P)$ denotes the average spacing of all the points to their k nearest neighbors, k being set to 6.

Using the exact model area, i.e., $\sum_{i=1}^{N} x_i \cdot area(f_i)$, as the denominator ensures that the value of the *point coverage* term is within the range [0, 1]. However, this results in a non-linear objective function that is difficult to optimize. Considering the actual surface area of the final model is comparable to the area of the point cloud's bounding box, we replace the exact model area with the area of the point cloud's bounding box, i.e., $area(M) \approx area(bbox(P))$.

5.2. Optimization

With the above energy terms, the optimal set of faces can be obtained by minimizing a weighted sum of these terms under certain hard constraints enforcing the final model to be manifold without boundary.

Remember that the candidate faces are obtained by the pairwise intersection of planes. Thus an edge is connected to either one (for boundary faces) or four faces (for inner faces). See Figure 2 (a) for an example. It is quite obvious that the necessary and sufficient condition for manifold and watertight polygonal surfaces is that each edge of the model connects only two adjacent faces. Thus, the final formulation for face selection can be written as

$$\begin{aligned} \min_{\mathbf{x}} \quad & \lambda_f \cdot E_f + \lambda_m \cdot E_m + \lambda_c \cdot E_c \\ \text{s.t.} \quad & \begin{cases} \sum_{j \in \mathcal{N}(e_i)} x_j = 2 \quad \text{or} \quad 0, & 1 \le i \le |E| \\ x_i \in \{0, 1\}, & 1 \le i \le N \end{cases} \end{aligned} \quad (6)$$

where $\sum_{j \in \mathcal{N}(e_i)} x_j$ counts the number of faces connected by an edge e_i. This value is enforced to be either 0 or 2, meaning none or two of the faces are selected (see Figure 2 (b) - (g) for possible selections). These hard constraints guarantee that the final model is manifold and closed.

The problem defined in Equation 6 is a binary linear program. We solve it using the Gurobi solver [7]. After optimization, the union of the selected faces (i.e., faces having a corresponding variable $x_i = 1$) comprises a polygonal surface model approximating the object.

6. Results and Discussion

We implemented our method using C++. The α-shapes are constructed using the CGAL library [5]. We tested our method on various real-world objects of different complexity. Experiments demonstrated the advantages of our *hypothesizing and selection* strategy.

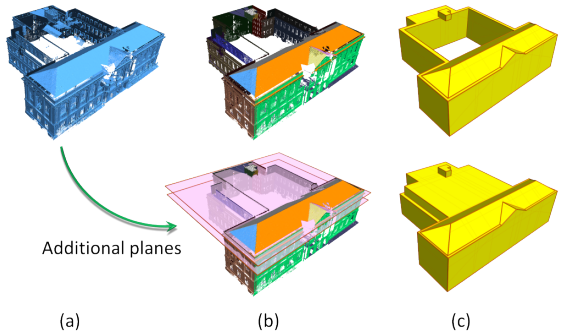

Figure 5. Reconstruction of a building with completely missing planes (top row). By adding an extra plane, model with different structures can be obtained (bottom row).

Reconstruction results. Our method can reconstruct piecewise planar objects such as buildings and other man-made objects. Figures 1 and 4 (a) - (d) show the reconstruction of five buildings of different styles. The input of these buildings are point clouds computed from images using Multi-View Stereo techniques. Although the point clouds are quite noisy and contain significant amounts of outliers and missing data, our method faithfully reconstructed these buildings. In Figures 4 (e) and (f), two indoor scenes consisting of multiple rooms are reconstructed. These two scenes were captured by Google Tango tablets. Due to the cluttered nature of indoor scenes, the datasets contains large holes. Our *hypothesizing and selection* strategy fills in the missing regions and reconstructed the permanent structures (i.e., walls and roofs) of these scenes.

Besides the buildings, we also tested our method on other man-made objects. In Figure 4 (g), our method took the laser scan of a small packing foam box as input and faithfully reconstructed all its structures with sharp features. We consider this as the *first* advantage of our approach. In (h) and (i), two sofas of different styles are reconstructed. These pieces of furniture were scanned by PrimeSense RGB-D cameras in the form of dense triangular meshes [4]. Our method took the vertices of the meshes as input and produced lightweight polygonal surface models.

In rare cases where few planes are completely missing, our method may still generate plausible reconstructions. In Figure 5 (top), a building with its back roof completely missing is reconstructed. It is interesting to see that by adding few extra planes, models with distinct structures can also be obtained (bottom row). Thanks to the manifold and watertight constraints, our method guarantees the reconstructed models to be manifold without boundary. We consider this as the *second* advantage of our approach.

In Table 1, we give statistics of our quantitative results.

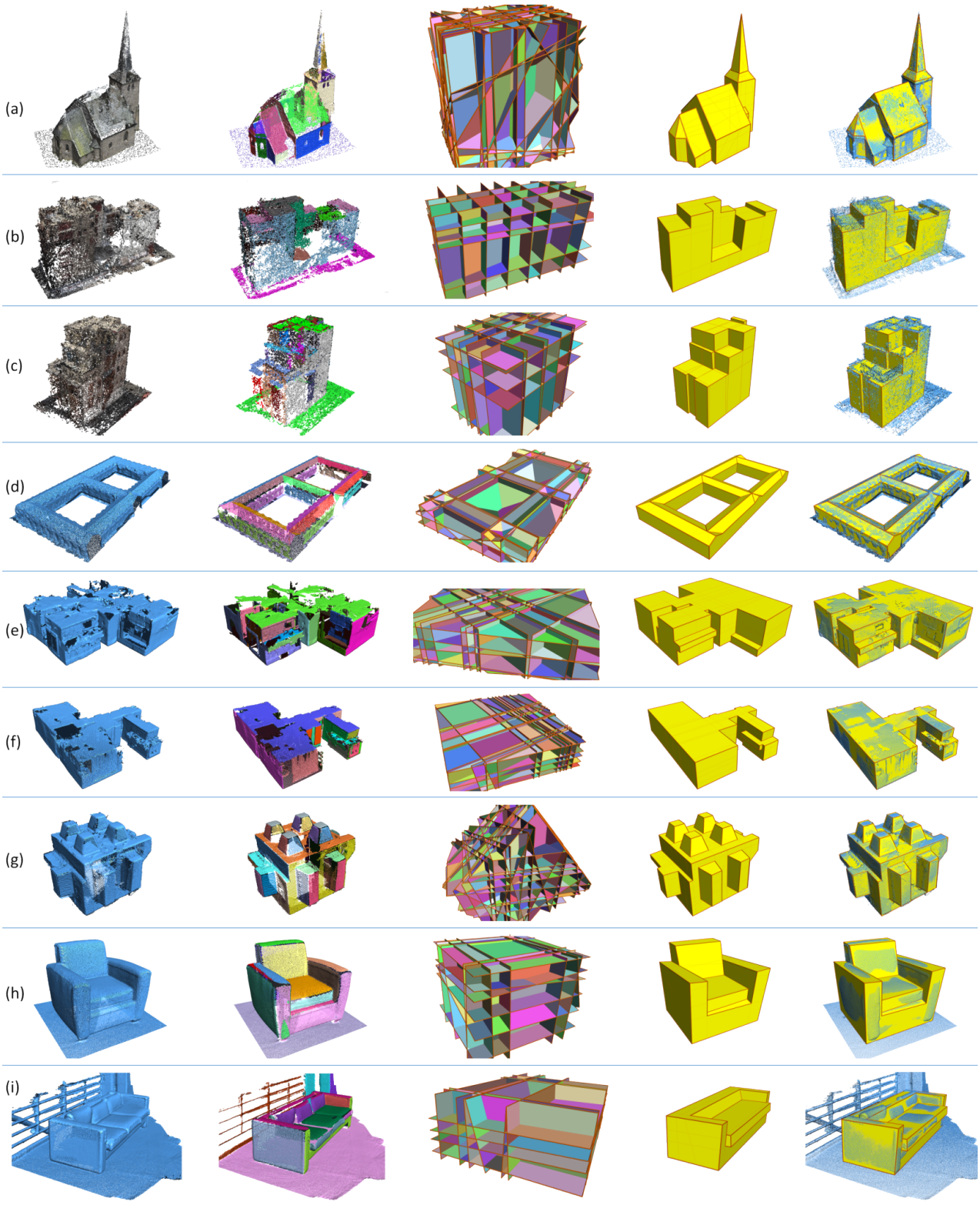

Figure 4. Reconstruction of a set of piecewise planar objects from various data sources. The input for (a) - (d) are computed from images using Multi-View Stereo; (e) and (f) are acquired by Google Tango tablets; (g) is captured by a laser scanner [17]; (h) and (i) are acquired by PrimeSense Carmine RGB-D cameras [4]. From left to right: input point clouds, extracted planar segments, candidate faces (randomly colored), reconstructed models, and models overlaid with the original point clouds (rendered with a smaller point size).

Table 1. Statistics on the examples presented in Figure 4.

Model index in Figure 4	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(i)
# points	129 K	101 K	73 K	577 K	186 K	176 K	382 K	756 K	911 K
# planar segments	36	21	22	22	28	33	52	17	19
# candidate faces	6239	472	946	959	1778	2770	7540	580	523
# faces of the final model	38	18	25	29	26	35	52	16	14
Primitive extraction (sec)	0.21	0.14	0.09	1.17	0.45	0.13	0.93	1.04	1.54
Candidate generation (sec)	0.05	0.01	0.01	0.02	0.01	0.02	0.06	0.02	0.03
Face selection (optimization) (sec)	2.73	1.46	1.17	8.26	2.90	3.05	7.39	13.26	16.01

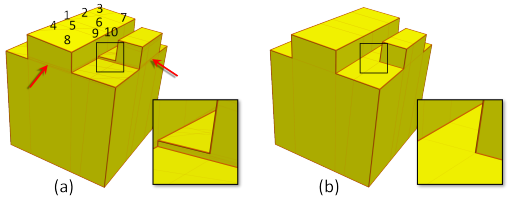

Figure 6. Result without (a) and with (b) plane refinement step. In (a), the top comprises ten faces originating from three different planes. The red arrows indicate long but very thin polygonal faces.

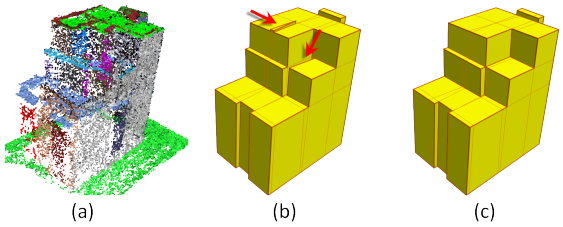

Figure 7. A building (a) reconstructed without (b) and with (c) the complexity term. The red arrows indicate bumps and gaps.

As can be seen from the number of faces in the final models, our method can produce lightweight 3D models. We consider this as the *third* main advantage of our approach.

Timings. Table 1 shows the running times of each step for the examples presented in Figure 4. We can see that the face selection step is slower compared to primitive extraction and candidate face generation. It should be noted that the construction of the α-shape meshes dominated this step. Down-sampling of input point clouds may speed up this process.

Effect of plane refinement. We tested our algorithm on a few data sets by omitting the refinement step. One such example is shown in Figure 6. We can see that even without the plane refinement step, our optimization still recovers the main structure of this building. However, we observe that the two models have some differences in the details. First, the top-most face of the reconstructed polyhedron in (a) is composed of ten candidate faces lying on three different planes, while in (b) it comprises fewer faces originating from the same plane. Second, some undesired bumpy structures can be avoided with the plane refinement step. This is because some initial faces are quite close to each other, making the energy terms less distinguishable. Another benefit of the plane refinement step is the gain in efficiency. With plane refinement, the total number of candidate faces is reduced from 1185 to 348. Accordingly, the run-time of the optimization decreased from 1.25 seconds to 0.31 seconds, i.e., more than four times faster.

Effect of the energy terms. The core of our reconstruction method lies in the optimization-based face selection that is designed to favor different aspects necessary for high-quality reconstructions. Both data fitting and coverage are essential requirements. We observed that having only these two terms works perfectly for reasonably complete datasets, but it is still not sufficient for point clouds with significant imperfections (i.e., noise, outliers, and missing data). Figure 7 (b) shows such an example. We can see that the reconstructed model demonstrates some desired bumps and gaps. In contrast, with our model complexity term, the reconstructed model is cleaner and compact (c). To further evaluate the behavior of the model complexity term, we ran our face selection on a dataset by gradually increasing the weight (see Figure 9). Not surprisingly, increasing the influence of the model complexity term resulted in less detailed 3D models. Thus, the model complexity term also provides control over the model details.

Parameters. The parameters for most examples are as follows: $\lambda_f = 0.46$, $\lambda_c = 0.27$, and $\lambda_m = 0.27$ (Note that weights in a wide range can produce the same results). Slightly different weights ($\lambda_f = 0.3$, $\lambda_c = 0.4$, and $\lambda_m = 0.3$) are used for the sofa example in Figure 4 (i), where the background (ground plane) has a much higher density than the object (sofa), thus the smaller data fitting weight.

Robustness and accuracy. We evaluated the robustness of our approach by reconstructing from synthetic data with an increasing amount of noise (see Figure 10). In (c), the standard deviation of the noise distribution is 0.6 meters high, our method still obtained topologically accurate reconstruction. However, a much higher level

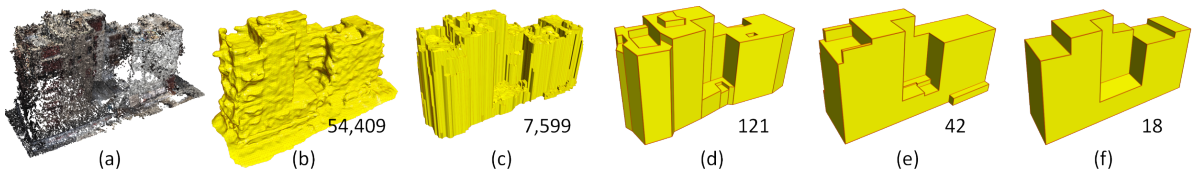

Figure 8. Comparison with four state-of-the-art methods on a building dataset. (a) Input point cloud. (b) Model reconstructed by the Poisson surface reconstruction algorithm [11]. (c) The result of the 2.5D Dual Contouring approach [27]. (d) The result of [15]. (e) The result of [16]. (f) Our result. The number under each sub-figure indicates the total number of faces in the corresponding model.

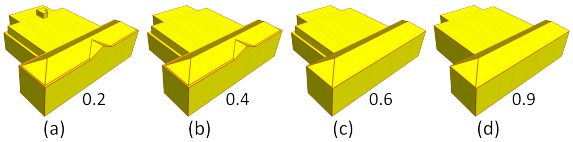

Figure 9. Reconstruction of the building shown in Figure 5 by gradually increasing the influence of the model complexity term. The values are the weights used in the corresponding optimization.

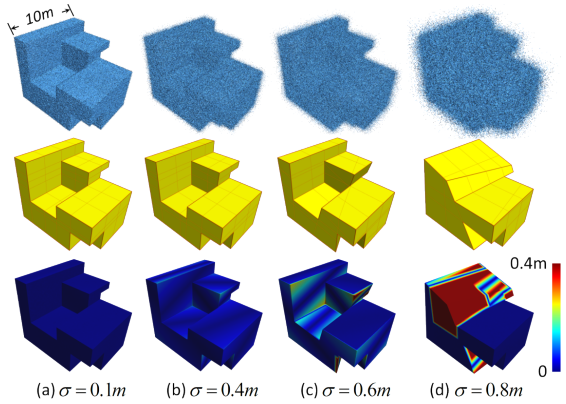

Figure 10. Reconstruction from synthetic data with increasing noise. Top: input. Middle: reconstruction. Bottom: reconstruction error. σ indicates the standard deviation of the Gaussian noise.

of noise may further pollute the structures of the object, resulting in large errors in the final model.

Comparison. Figure 8 shows the comparison of our approach with four other methods on a building dataset. We can see that both Poisson [11] and 2.5D Dual Contouring [27] generate dense surfaces with large numbers of bumps. The method in [15] uses only the roof information and also produce a 2.5D reconstruction. By making the Manhattan-World assumption, the result of [16] is more regularized. However, these two methods both produce undesired structures. In contrast, our approach generates the most compact and clean model. Besides, we can also observe that our result is less regular than that of [16]. However, this can be improved by a post-processing step using the coarse model optimization technique proposed in [20].

Limitations. Our *hypothesizing and selection* based reconstruction strategy is intended for reconstructing simple polygonal surfaces. It may encounter computation bottlenecks for large complex objects (e.g., urban scenes of many buildings). Running on such objects results in a huge number of candidate faces and the computation may not be affordable. In our experiments, we did not encounter such situations because we only tested our method on problems of manageable scales.

7. Conclusions and Future Work

We introduced a novel framework that casts polygonal surface reconstruction from point clouds as a binary labeling problem. Our framework is based on a *hypothesizing and selection* strategy, and we proposed a novel optimization formulation to obtain lightweight, watertight polygonal surface models. We demonstrated the effectiveness of our method on datasets captured by a range of devices. Since our approach seeks to find an optimal combination of the intersected planes under manifold and watertight constraints, it is guaranteed to produce lightweight, manifold models without boundary.

Future direction. Our current implementation exploits planar primitives and it is suitable for reconstructing piecewise planar objects. However, the *hypothesizing and selection* strategy is general and has the potential to handle various types of basic primitives. In future work, we would like to extend our method to incorporate more types of geometric primitives, such as cylinders and spheres. To handle data with completely missing planes, we plan to infer the missing planes by exploiting structural priors (e.g., symmetry, parallelism, and orthogonality) to obtain complete reconstructions.

Acknowledgements

We would like to thank Florent Lafarge, Michael Wimmer, and Pablo Speciale for providing us the data used in Figures 4 (d), (a) , and (e)-(f), respectively. We also thank Tina Smith for recording the voice-over for the video. This research was supported by the KAUST Office of Sponsored Research (award No. OCRF-2014-CGR3-62140401) and the Visual Computing Center (VCC) at KAUST.

References

[1] M. Arikan, M. Schwärzler, S. Flöry, M. Wimmer, and S. Maierhofer. O-snap: Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics*, 32:6:1–6:15, 2013.

[2] A. Boulch, M. de La Gorce, and R. Marlet. Piecewise-planar 3d reconstruction with edge and corner regularization. In *Computer Graphics Forum*, volume 33, pages 55–64, 2014.

[3] 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*, pages 1261–1268.

[4] S. Choi, Q.-Y. Zhou, S. Miller, and V. Koltun. A large dataset of object scans. *arXiv preprint:1602.02481*, 2016.

[5] T. K. F. Da. 2D alpha shapes. In *CGAL User and Reference Manual*. CGAL Editorial Board, 4.9 edition, 2016.

[6] M. A. Fischler and R. C. Bolles. Readings in computer vision: Issues, problems, principles, and paradigms. chapter Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography, pages 726–740. 1987.

[7] Gurobi. Gurobi optimization. `http://www.gurobi.com/`.

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

[9] H. Jiang and J. Xiao. A linear approach to matching cuboids in rgbd images. In *CVPR 2013*.

[10] I. Jolliffe. Principal component analysis. *Encyclopedia of Statistics in Behavioral Science*.

[11] M. Kazhdan, M. Bolitho, and H. Hoppe. Poisson surface reconstruction. Symposium on Geometry Processing 2006, pages 61–70.

[12] F. Lafarge and P. Alliez. Surface reconstruction through point set structuring. In *Computer Graphics Forum*, volume 32, pages 225–234, 2013.

[13] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk. The digital michelangelo project: 3d scanning of large statues. In *SIGGRAPH 2000*, pages 131–144.

[14] B. Li, R. Schnabel, J. Shiyao, and R. Klein. Variational surface approximation and model selection. *Computer Graphics Forum*, 28(7), 2009.

[15] M. Li, L. Nan, N. Smith, and P. Wonka. Reconstructing building mass models from uav images. *Computers & Graphics*, 54(C):84–93, 2016.

[16] M. Li, P. Wonka, and L. Nan. Manhattan-world urban reconstruction from point clouds. In *ECCV 2016*, pages 54–69.

[17] 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*, volume 30, page 52.

[18] 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. *SIGGRAPH 2013*, 32(4):66.

[19] A. Monszpart, N. Mellado, G. J. Brostow, and N. J. Mitra. Rapter: Rebuilding man-made scenes with regular arrangements of planes. *SIGGRAPH 2015*, 34(4):103:1–103:12.

[20] L. Nan, C. Jiang, B. Ghanem, and P. Wonka. Template assembly for detailed urban reconstruction. In *Computer Graphics Forum*, volume 34, pages 217–228, 2015.

[21] L. Nan, A. Sharf, H. Zhang, D. Cohen-Or, and B. Chen. Smartboxes for interactive urban reconstruction. In *SIGGRAPH 2010*, volume 29, page 93.

[22] Y. Ohtake, A. Belyaev, M. Alexa, G. Turk, and H.-P. Seidel. Multi-level partition of unity implicits. In *SIGGRAPH 2003*, pages 463–470.

[23] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas. Example-based 3d scan completion. In *Symposium on Geometry Processing 2005*, pages 23–32.

[24] R. Schnabel, R. Wahl, and R. Klein. Efficient ransac for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226, 2007.

[25] C. A. Vanegas, D. G. Aliaga, and B. Benes. Automatic extraction of manhattan-world building masses from 3d laser range scans. *IEEE transactions on visualization and computer graphics*, 18(10):1627–1637, 2012.

[26] Y. Verdie, F. Lafarge, and P. Alliez. Lod generation for urban scenes. *ACM Transactions on Graphics*, 34(3), 2015.

[27] 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*, pages 115–128.

[28] M. Zuliani, C. S. Kenney, and B. Manjunath. The multiransac algorithm and its application to detect planar homographies. In *ICIP 2005*, volume 3, pages III–153.