To appear in *International Journal of Digital Earth*
Vol. 00, No. 00, Month 20XX, 1–13

Fitting Boxes to Manhattan Scenes Using Linear Integer Programming

Minglei Li[a]*and Liangliang Nan[b] and Shaochuang Liu[a]

[a]*Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing, China;* [b]*King Abdullah University of Science and Technology, Thuwal, Saudi Arabic*

(*January 2016*)

We propose an approach for automatic generation of building models by assembling a set of boxes using a Manhattan-world assumption. The method first aligns the point cloud with a per-building local coordinate system, and then fits axis-aligned planes to the point cloud through an iterative regularization process. The refined planes partition the space of the data into a series of compact cubic cells (candidate boxes) spanning the entire 3D space of the input data. We then choose to approximate the target building by the assembly of a subset of these candidate boxes using a binary linear programming formulation. The objective function is designed to maximize the point cloud coverage and the compactness of the final model. Finally, all selected boxes are merged into a lightweight polygonal mesh model, which is suitable for interactive visualization of large scale urban scenes. Experimental results and a comparison with state-of-the-art methods demonstrate the effectiveness of the proposed framework.

Keywords: Urban building models; Aerial point cloud; Manhattan scenes; Linear integer programming

1. Introduction

3D architectural models have significant value for various geo-referenced applications, such as urban planning, navigation, simulation, and virtual reality. However, the automatic generation of 3D urban models is still a challenging problem (Gruen 2008; Musialski et al. 2013; Rottensteinera et al. 2014).

In the last decade, the development of various data acquisition technologies, such as Light Detection and Ranging (LiDAR), RGB-D cameras, Structure from Motion (SfM), and Multi-view Stereo (MVS), enables users to effectively obtain a 3D sampling of an urban scene (i.e., a 3D point cloud). This technology opened up many interesting research problems concerned with processing such data (Xiao et al. 2015; Li et al. 2016; Vanegas, Aliaga, and Benes 2012; Lin et al. 2013; Arikan et al. 2013; Nan et al. 2010; Zhou and Neumann 2010). Despite these recent efforts, the fully automatic generation of building mass models from noisy, incomplete point clouds still remains an open problem. In practice, the reconstruction process from such noisy data often requires tedious manual work. This hinders the generation of 3D urban models for large scale environments. Besides, a recent trend is that the

*Corresponding author. Email: minglei_li@126.com

interest of urban modeling has shifted from LiDAR-based methods to MVS-based methods (Verdie, Lafarge, and Alliez 2015; Nan et al. 2015; Wang et al. 2015; Li et al. 2016; Furukawa et al. 2009). Although LiDAR has better accuracy, its coverage is limited to building roofs thus the details of the facade is not accessible; on contrast, MVS has a better accessibility for the information of a building structure. Because of this trend, there is a great need to deal with MVS data that suffers from high level of noise, distorted structures and incomplete geometry due to imperfect camera geometry and object occlusion. Thus primitive-based methods (boxes for example) tend to be superior to data-driven methods.

In this work, we address the problem of automatic reconstruction of Manhattan scenes, which contain structures with a predominance of three mutually orthogonal directions, from imperfect point clouds. This method outputs lightweight 3D polygonal mesh models of the buildings, which are especially suitable for web-based visualization of large urban scenes in a framework of digital earth (Zhang et al. 2014). Our observation is that urban buildings usually exhibit large amount of planar geometries and regularities (i.e., orthogonality and parallelism). Thus, the underlining geometry of these buildings can be represented by an assembly of boxes. This motivates us to fit boxes to the point cloud to approximate the structure of each individual building.

Our strategy relies on choosing a minimum number of boxes from a large number of candidates using a binary integer programming optimization. First, the input point cloud is aligned with a local per-building coordinate system, and a large amount of plane hypothesis are detected from the point cloud using a random sample consensus (RANSAC) method (Schnabel, Wahl, and Klein 2007). Then, these planes are iteratively refined to best fit the input point cloud. Thus, they partition the space of the input data into a grid. In this paper, we call each cell in the grid a candidate box. Finally, we choose a subset of these candidate boxes based on a linear integer programming optimization, and assemble them into a lightweight polygonal mesh model. During the entire process, structural regularities of architectural models, such as orthogonality and parallelism, are taken into consideration for both candidate box generation and the later optimization steps.

Our method offers the following advantages for Manhattan scene reconstruction: 1) it is remarkably robust to outliers and missing data, and 2) it automatically produces clean, lightweight, and watertight models. Thus, it is quite suitable for large scale urban reconstruction.

The key contributions of our work include:

- a novel framework for the automatic reconstruction of Manhattan scenes by directly fitting boxes to imperfect point clouds.
- an iterative regularization process for candidate box generation from imperfect point clouds.
- a linear integer programming formulation for selecting a subset of candidate boxes so as to obtain a compact polygonal model that fits to the point cloud.

2. Related work

Given the large volume of related work in the literature, in this section we only review the work that are most related to our proposed method. According to the reconstruction strategies, we discuss related work with respect to model-driven and

data-driven methods.

Model-driven methods. Urban buildings usually exhibit strong structural regularities, such as piecewise planar facades, orthogonality, and parallelism, etc. This prior knowledge about the structure of buildings has been exploited extensively for the urban reconstruction problem. To reconstruct detailed but lightweight architectural models, Nan et al.(Nan et al. 2015) propose to use image information to assemble a set of predefined detailed facade elements onto coarse building models. By making a Manhattan-world assumption, Matei et al. (Matei et al. 2008) and Venegas et al. (Vanegas, Aliaga, and Benes 2010) extract regular grammars from LiDAR point clouds via different approaches. Then a volume description of the building is extracted from the clusters of the classified points. Using a similar assumption, Furukawa et al. (Furukawa et al. 2009) reconstruct indoor scenes by arranging the samples of a scene to axis-aligned planes. With such structural properties as guidance or as high-level constraints, the reconstruction results from these methods usually outperform those from other methods in terms of controllability over both geometric and semantic complexity of the final models. However, these methods require uniform distribution of the point cloud, so they were basically designed for close-range LiDAR data. The airborne MVS points with uneven density will hinder the performance of these mentioned methods.

Another group of model-driven methods, namely contour-based methods, first perform a segmentation step to extract the contours of the buildings (usually followed by a refinement step), and then assemble the 2.5D building model by exploiting the structural properties of urban buildings. Poullis and You (Poullis and You 2009) create large scale city models from airborne LiDAR data by simplifying and refining 2D boundaries of buildings, from which 3D models are extruded fitting the segmented regions. Zhou and Neumann (Zhou and Neumann 2010) learn a set of principal directions that align with roof boundaries of the buildings. These roof boundaries are then used as footprint for extruding 2.5D models. In their follow up work (Zhou and Neumann 2013), they optimize the 2D boundaries of roof layers, which enables the reconstruction of buildings with arbitrarily shaped roofs. This method work well for dealing with clean and accurate LiDAR data. Larfage et al. (Lafarge et al. 2010) approximate the urban buildings by assembling 3D blocks on a Digital Surface Model (DSM). They use a Bayesian decision to find the optimal configuration of the 3D-blocks. However, these contour-based reconstruction approaches mainly exploit the roof information of the buildings, while the walls and facades (though sparse and incomplete) are ignored during the processing.

Data-driven method. Delaunay-based methods and implicit surface reconstruction are quite common in this area. The basic idea behind the Delaunay-based methods is that the reconstructed triangulated surface is formed by a subcomplex of the Delaunay triangulation. These methods place rather strong requirements on the point cloud and are impractical for MVS data containing significant imperfections. Poisson reconstruction approach (Kazhdan and Hoppe 2013) is a widely used implicit surface reconstruction method. Depending on an indicator function, Poisson reconstruction estimates a labeling to discriminate the interior from the exterior of a solid shape and approaches a surface for the solid. However, Poisson reconstruction requires the availability of oriented normals, which sometimes have poor accuracy when existing high-level of noises and outliers.

Graph cut-based methods are widely used in many related works. Garcia et al. (Garcia-Dorado, Demir, and Aliaga 2013) proposed a surface graph cuts approach for architectural modeling based on a volumetric representation. Hiep et al. (Hiep

et al. 2009) reconstructed the mesh models of different scales by extracting a visibility consistent mesh from the dense point cloud using a minimum s-t cut based global optimization. Then the mesh models are further refined relying on image information. Verdie et al. (Verdie, Lafarge, and Alliez 2015) conduct an abstraction operation on the dense meshes to obtain a level-of-detail representation of urban scenes, and surface models are extracted by a min-cut formulation. However, the optimization processes of these methods are computationally expensive since the problems are defined in 3D space. This is especially true when the scene exhibits complex structures. In addition, since these data-driven methods generally target dense mesh models, the complexity of the reconstructed models limits the application scope of these approaches.

In this work, we tackle the reconstruction problem using another strategy, i.e., transforming the reconstruction problem as assembling a set of boxes directly into the point clouds.

3. Overview

The goal of this work is to directly fit boxes to 3D point clouds for urban reconstruction. Our method takes as input a 3D point cloud of a scene (either from laser scanner or extracted from images using MVS), and outputs a 3D polygonal mesh model of the scene. In this work, we are particularly interested in fitting a set of boxes directly into the point cloud. Our method consists of two core steps.

Candidate box generation. We first extract a large number of planar segments from the input point cloud using RANSAC (Schnabel, Wahl, and Klein 2007). Since the point cloud may have noises, outliers, and missing data, the detected planar segments unavoidably contain undesired elements. Thus, we refine these planar segments by iteratively merging plane pairs and fitting new planes. After these planes are adjusted to better fit to the point cloud, we use these planes to partition the space of the input point cloud into regular cells. These cells can be considered as the input candidate boxes for the later binary optimization.

Box selection. In this step, we optimally choose a subset of the candidate boxes to build a valid 3D model of the scene. To do so, we formulate the boxes selection as a linear integer programming problem. Our objective function is designed to encourage the final model to cover more of the points and meanwhile be compact (i.e., minimum volume). For efficiency, we run a step of candidate box pruning before the optimization so as to filter out large amount of invalidate candidate boxes.

An overview of the proposed approach is shown in Figure 1.

4. Candidate Box Generation

With the Manhattan-world assumption, the walls and roofs of the buildings in a scene can be abstracted as axis-aligned planes. Thus, we first identify the three dominant orientations of the scene, as well as a set of plane hypothesis on which most of the geometry lies in. Then we iteratively refined these planar segments and generate candidate boxes from the refined planar segments.

Dominant orientations. To determine the three dominant directions of the scene, we identify the three strongest peaks from the histogram of the normal distribution of the point cloud (Furukawa et al. 2009). Then the corresponding normal

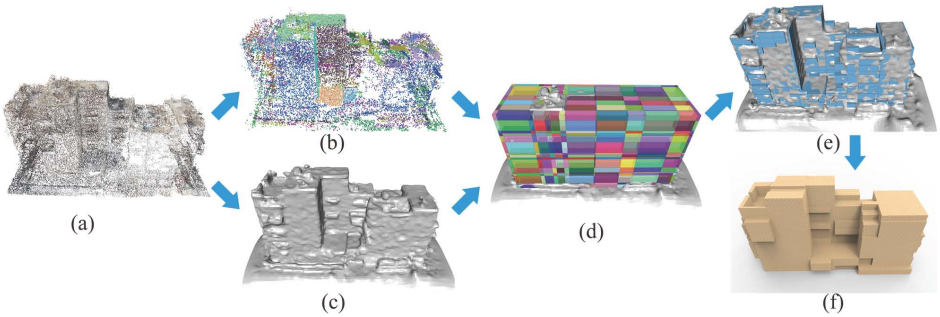


Figure 1. An overview of the proposed approach. Starting from an imperfect point cloud (a) of a building, we first extract and refine planar segments (b) from the point cloud, and build a dense mesh model using existing techniques. Then, we use the extracted planar segments to partition the space of the input point cloud into axis-aligned cells (i.e., candidate boxes). (d) shows the overlay of the candidate boxes on the dense mesh model. After that, appropriate boxes (e) are selected based on binary linear programming optimization. Finally, a lightweight 3D model (f) is assembled from the chosen boxes.

direction for each peak can be regarded as one of the dominant directions. With these dominant directions, it is trivial to transform the point cloud to be axis-aligned with the dominant directions.

Plane detection. As demonstrated in previous work, the RANSAC-based primitive detection method proposed by Schnabel et al. (Schnabel, Wahl, and Klein 2007) has proven to be effective and efficient for extracting several types of geometric primitives from noisy point clouds. We use it to detect a set of initial planar segments from the point cloud. However, due to the high-level of noise and the significant amount of outliers, the orientations of detected planar primitives do not always coincide with the three dominant directions. To tackle this problem, we propose an algorithm that iteratively refines the initial planar primitives.

4.1. Plane refinement

Considering the RANSAC primitive detection algorithm is designed based on investigating the number of points within a distance threshold to the primitives. We first run the RANSAC algorithm multiple times to generate a large number of initial plane hypothesis. This is to make sure appropriate plane segments that describe the structure of the scene exist among them.

We discard planar segments if either their orientations are far away from the three dominant directions, or they have a small number (20 minimum) of supporting points. In the next iterative refinement stage, we score each planar segment according to the number of its supporting points. Then starting from the pair of planar segments with lowest average score, we merge them if the following two conditions are satisfied: 1) the angle between the two planes is less than a threshold θ_t, and 2) the distance from the mass center of the set of points associated with one primitive to the other is less than a threshold d_t. After that, a new planar primitive is suggested by performing a least-squares fitting of the merged points. We repeat this process until no more pairs of planar segments can be merged. As a result, the planar segments are refined such that they are more coinciding with the dominant orientations, and meanwhile the number of planar segments is significantly reduced.

Figure 2 shows an example of the plane refinement process. Empirically, we set θ_t to 10° and d_t to $0.1m$. In our experiments, we observe that stable planar primitives can be obtained after a few iterations of the merging operation. A visual comparison

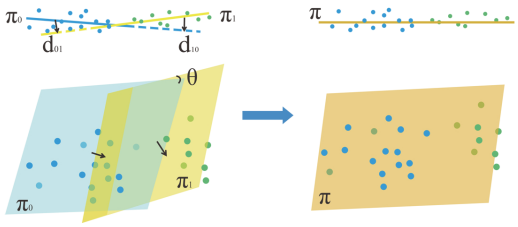


Figure 2. Plane refinement. Two planes π_0 and π_1 are merged if the angle between them is smaller than a threshold (i.e., $\theta < \theta_t$), and the distance from the mass center of the set of points associated with one plane to the other is less than another threshold (i.e., $d_{01} < d_t$ and $d_{10} < d_t$). Then a new plane π is proposed using a least-squares fitting of the union of the points.

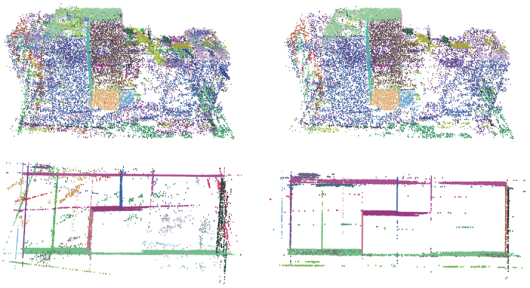


Figure 3. The arrangement of the planar primitives before (left) and after (right) the refinement step.

of the planar primitives for a building before and after the iterative refinement is shown in Figure 3. As can be seen from Figure 3 (right), the arrangement of the planar primitives has been significantly regularized and the number of primitives is reduced.

4.2. Candidate boxes generation

According to the orientations, the refined planar segments from the previous step can be separated into three groups, i.e. $\mathbf{G}_x$, $\mathbf{G}_y$, and $\mathbf{G}_z$, which are aligned with the three dominant directions. The supporting planes of these planar segments partition the space of the data into a set of axis-aligned cuboid cells. Assuming N_x, N_y, and N_z are the numbers of the planes along the three dominant directions (i.e., $|\mathbf{G}_x| = N_x$, $|\mathbf{G}_y| = N_y$, and $|\mathbf{G}_z| = N_z$), the total number of candidate boxes is given by

$$N = (N_x - 1) \cdot (N_y - 1) \cdot (N_z - 1). \tag{1}$$

In the next step, we will optimally choose a subset of these candidate boxes so as to approximate the building geometry and to obtain a compact polygonal mesh model.

5. Box Selection

In order to choose a subset of the candidate boxes that best describe the underlining geometry of the buildings, we propose a binary optimization approach based on a linear integer programming formulation. For efficiency reason, we conduct a pruning step that filters out a significant number of candidate boxes that are not likely to contribute to the building geometry.

5.1. Candidate box pruning

The number of the candidate boxes generated from the arrangement of the planar segments is usually large (see Equation 1). For example, a scene consists of 50 candidate parallel walls in each dominant direction will result in $(50-1)^3$ candidate boxes, and a linear integer programming optimization problem with the same number (117, 649, to be specific) of variables. Obviously, solving this optimization problem is computationally inefficient. Thus, it is necessary to filter out those candidates that clearly do not contribute to the final reconstruction.

One possible way to discard redundant candidate boxes is by performing a valid/invalid check for each candidate box. We observe that a large portion of the candidate boxes resides either inside or outside the surface of the building, and so these boxes do not contribute to the building's surface representation. This observation motivates us to identify and remove these candidate boxes. To this end, we first obtain an approximate reconstruction of the surface model of the scene using the Poisson reconstruction algorithm (Kazhdan and Hoppe 2013). Although the surface model reconstructed from (Kazhdan and Hoppe 2013) is not precise in terms of both geometry and topology due to noise and missing data, it provides sufficient information for identification of the unwanted candidate boxes.

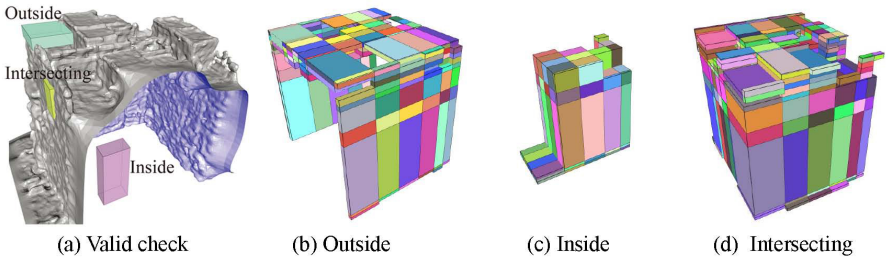


Figure 4. Categorize candidate boxes into three different status.

The status of a candidate box is determined similarly to determining the location of a 3D point with respect to a polyhedron. Specifically for each box, we cast rays from the 8 corners of that box to the corresponding corners of the bounding box of the scene. Then the candidate boxes can be classified into the following three categories by counting the number of intersections of the casted rays against the surface model:

- Outside, if all rays have even numbers (including zero) of intersections against the surface model.
- Inside, if all rays intersect the surface model with odd numbers and there is no point resided on the facets of the box.
- Intersecting, if some rays have odd numbers and others have even numbers of intersections against the surface model, or the box facets contain points.

During the judgment, a point is determined to reside on one of 6 facets of a box only when it meet the following criteria: 1) its distance to the facet is less than 40 cm; and 2) the angle between the point normal vector and the facet normal vector is less than 30 degree.

Among the above three categories (see Figure 4), candidate boxes with outside status are obviously unwanted and are first discarded. The inside boxes are temporarily set aside, as they will be used at the end to merge for the final complete

model. Hence, only these boxes with status intersecting will be taken as input in the later optimization step.

5.2. Optimization

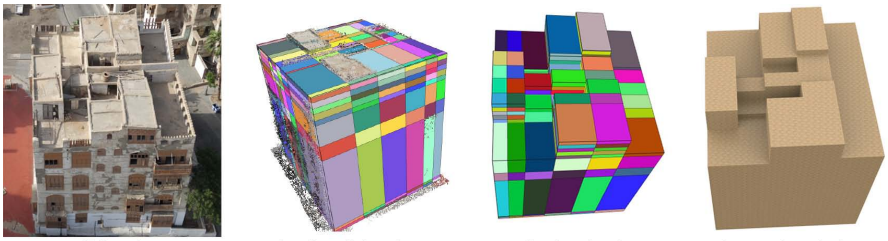


Figure 5. The optimization result of a quasi Manhattan building.

The goal of the optimization step is to choose an optimal subset of the candidate boxes to assemble a compact 3D polygonal model for the scene described by the point cloud. We formulate the candidate box selection as a zero-one (binary) linear programming problem.

Given N valid candidate boxes $b_i (1 \leq i \leq N)$, let $\mathbf{X}$ denote the binary labels for all the candidate boxes and let $\mathbf{x_i}$ correspond to the i th box's binary option, the solution to box selection problem is a subset of the candidate boxes that minimizes an energy balancing between two terms: point coverage and compactness.

- **Point coverage**. Since we are reconstructing 3D models from point clouds, we prefer that the final model covers more of the sampled points. Thus, a score function $S(b_i)$ is defined to measure how much a candidate box b_i is supported by the point cloud. Specifically, the score function $S(b_i)$ is defined as below

$$S(b_i) = \frac{\sum_{j=1}^{6} num(f_j)}{\sum_{j=1}^{6} \rho \cdot A(f_i) \cdot \mathbb{1}(f_j)}, \tag{2}$$

  where $num(f_j)$ denotes the number of points that reside on the j_{th} face f_j of box b_i; ρ has a uniform value that is set to the sampling density of the face with highest confidence (i.e., highest point density); $A(f_i)$ is the area of the face f_j; $\mathbb{1}(f_j)$ is an indicator function that has value 1 if there exists points lying on face f_j, otherwise it has value 0. In other words, we only count faces that have supporting points into the scoring function. Thus, the point coverage term is defined as

$$E_c(\mathbf{X}) = 1 - \sum_{i=1}^{N} x_i \cdot S(b_i)/N. \tag{3}$$

- **Compactness**. This term encourages representing the final model by a compact assembly of the boxes. Let $V(b_i)$ denote the volume of box b_i, then the

compactness term is defined as

$$E_v(\mathbf{X}) = \sum_{i=1}^{N} x_i \cdot V(b_i)/V_{bbox}, \tag{4}$$

where V_{bbox} denotes the volume of the bounding box of the scene.

Intuitively, the point cloud coverage term E_c encourages to choose boxes that are supported by dense points, and the compactness term favors smaller boxes. By putting these two terms together, our objective function is given as below

$$E(\mathbf{X}) = E_c(\mathbf{X}) + \lambda \cdot E_v(\mathbf{X}), \tag{5}$$

where λ is a weight parameter that balances between the point coverage term and the compactness term. In our experiments, λ is set to 0.1 for all the examples shown in this paper.

Minimizing the above energy (see Equation 5) results in a zero-one (binary) linear programming problem. We solve it using the conventional Gurobi solver (Gurobi 2015). After the energy being minimized, the variables with value 1 suggest the subset of candidate boxes that approximate the underlining structure of the scene.

We observe walls of a building can be completely missing from the point cloud due to occlusions. In such case, it results in some holes in the final 3D models. As a compensation, the inside boxes are employed to fill the holes and to present a complete solid entity. Specifically, we merge together both the inside boxes and those suggested by the optimization, and then extract their boundary faces as the final polygonal model.

To better depict the proposed framework, an illustration of the optimization procedure is shown in Figure 5, where (a) indicates an image of a quasi Manhattan building, (b) and (c) are the candidate boxes before and after optimization, and (d) presents a rendered view.

6. Results and Discussion

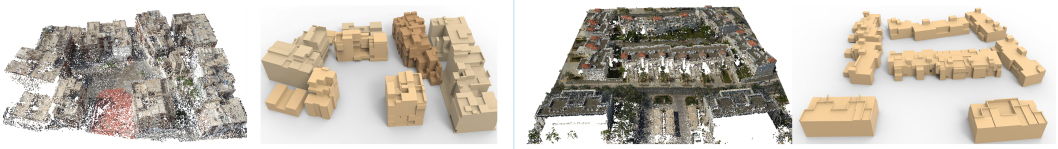


Figure 6. Two example scenes reconstructed using our method. Our approach can automatically reconstruct a Manhattan scene by fitting boxes into the noisy point cloud (left) of the scene. The final 3D model is shown on the right.

We have applied our approach on several datasets of real-world buildings and conducted both qualitative and quantitative evaluations of the proposed method.

Datasets. The point clouds used to test our algorithms are generated using MVS method from a series of images captured by a Sony QX100 camera ($20M$ pixels) and $24mm$ (equivalent lens) mounted on a unmanned aerial vehicle (UAV). The combination of the technologies of the MVS and UAV provides a flexible avenue for downtown modeling with low cost and medium accuracy. Unfortunately, since MVS

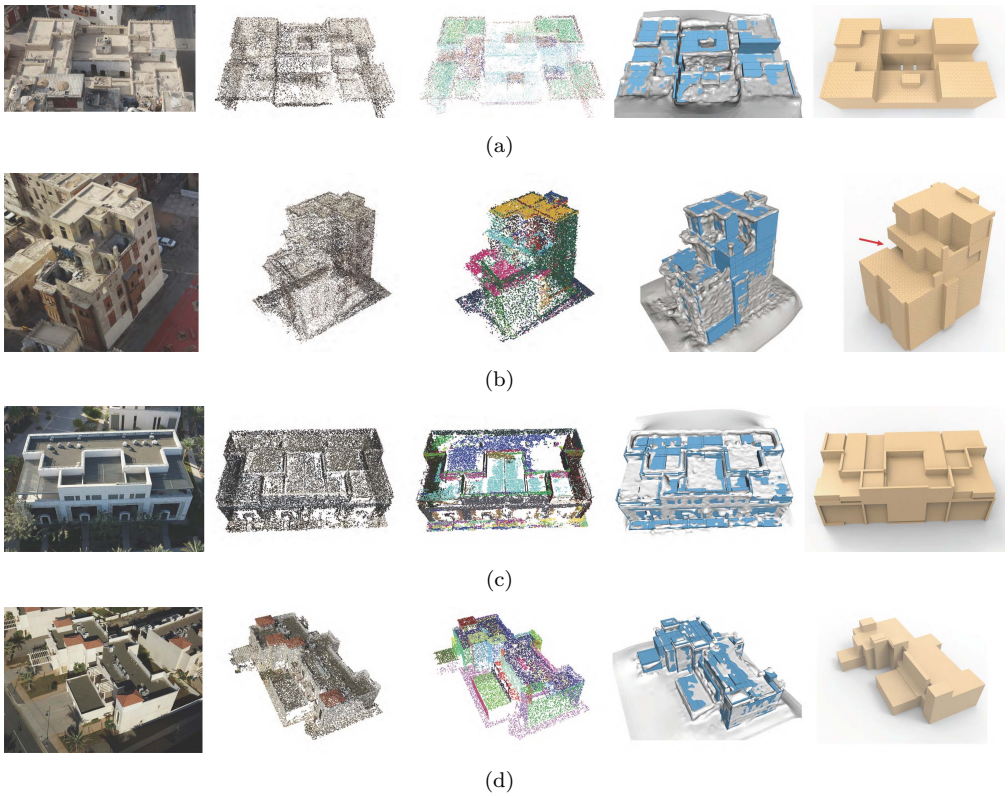


Figure 7. Reconstruction results. Each row (from left to right) shows the representative photograph captured by the UAV camera, initial point cloud extracted from MVS, detected planar segments, Poisson surface overlaid on the selected boxes, and the final 3D model, respectively.

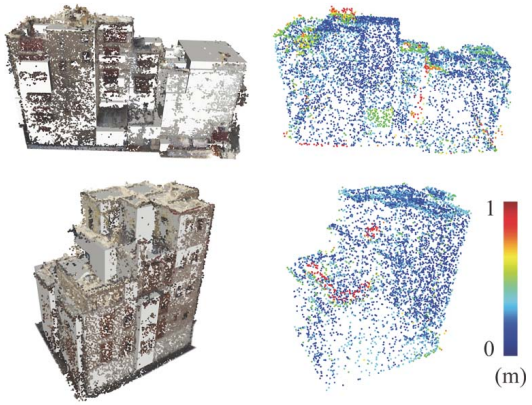


Figure 8. Reconstruction errors of two buildings. The left column shows the point clouds overlaid on the final 3D models, and the right column shows the reconstruction errors.

is based on local image features, the extracted point clouds are noisy, incomplete, and with uneven densities.

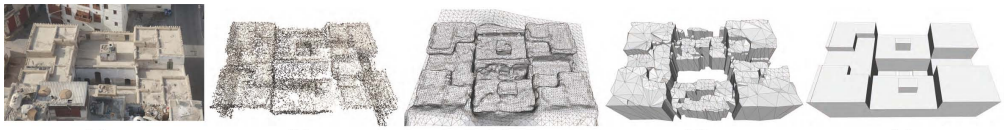
Reconstruction results. Results show that our method is able to generate faithful and compact polygonal models from the point clouds of complex scenes (see Figure 6) and individual buildings (see Figure 7).

As can be seen from Figure 6 and Figure 7, the final polygonal models are compact and meanwhile are faithful to the input point clouds. Note, although the point cloud is extremely noisy, sparse, and has a large amount of outliers and missing parts, our

method can still produce faithful reconstruction results.

Since ground truth of real world buildings are usually not available, we overlay the point clouds on the final 3D models and evaluate the quality of the reconstruction results by measuring the average distance from the points to their nearest faces in the polygonal model. Figure 8 shows two such examples. To be specific, our method has an average reconstruction error of $0.15m$ for all the examples shown in this paper.

Comparisons. We also conduct both qualitative and quantitative comparisons with state-of-the-art methods, namely the Screen Poisson surface reconstruction algorithm (Kazhdan and Hoppe 2013) and the 2.5D dual contouring method proposed by (Zhou and Neumann 2010).



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

Figure 9. Comparison of our method with Screened Poisson reconstruction(Kazhdan and Hoppe 2013) and 2.5D dual contouring(Zhou and Neumann 2013) methods on a single building. (a) A photograph of the buildings. (b) Input point cloud. (c) Dense mesh model reconstructed using the Screened Poisson algorithm(Kazhdan and Hoppe 2013). (d) Reconstruction result using the 2.5D dual contouring method(Zhou and Neumann 2010). (e) Our result.

From Figure 9, we can see that the Screened Poisson reconstruction method can generate isotropic dense surface models. This method, however, may fail if there exists large amounts of outliers or large portion of the facades are occluded (i.e., holes in the point clouds). In such a case, it usually produces some undesired surfaces passing through the outliers and the missing regions. Besides, the sharp features of the buildings are usually smoothed, and it's rather difficult to recover them as a post-processing step. These defects can also be observed from the fourth column in Figure 7. The 2.5D dual contouring method was initially designed to deal with aerial LiDAR point clouds with higher density and accuracy. Thus, it mainly relies on roof information and it is quite sensitive to noise and uneven point distribution. We can see from Figure 9 (d), the result from this method contains large areas of small bumps. Compared with these approaches, our method can generate more compact and visually pleasing reconstruction results.

Table 1. Comparison of our method with Screened Poisson reconstruction (SP)(Kazhdan and Hoppe 2013) and 2.5D dual contouring(Zhou and Neumann 2013) methods in terms of running times (in seconds), mesh sizes (measured as face number), and reconstruction errors (in meters).

	2.5D(Zhou and Neumann 2013)	SP(Kazhdan and Hoppe 2013)	Ours
Time	0.38	2.2	24
# Faces	2, 492	10, 806	212
Error	0.13	0.09	0.15

Table 1 shows a quantitative comparison of our approach with the aforementioned two methods on the building shown in Figure 9. We can see that the Screened Poisson reconstruction method wins in terms of accuracy, but the final surface model is more fluctuating. This can be seen from Figure 9 (c). Our method is a bit slow and has similar accuracy with the 2.5D dual contouring method, but the reconstruction results are more compact than the other two approaches. All

experiments are conducted on a laptop with an Intel i5-3210M CPU and a 4.0 GB RAM.

Limitations. In the candidate box pruning step, a surface approximation from Poisson reconstruction method is used to filter out a large number of irrelevant candidate boxes. For point clouds that have high-level of noise, outliers, and large missing regions, the Poisson reconstruction method are likely to produce undesired surfaces. So good boxes could be filtered out and thus our method will generate results with holes (see Figure 7 (b)).

Another limitation is that since our method is based on the Manhattan World assumption, it cannot handle some residential buildings with tilted planes, e.g., gable or hipped roofs. A possible avenues for the future work is to separating non-MW geometries from point cloud and expending other shape types.

7. Conclusions and future work

This paper presented a box fitting algorithm for reconstructing Manhattan scenes. Unlike previous work, we fit boxes directly into the noise and sparse point clouds. Our method is based on a generate and select strategy, i.e., we choose an optimal set of boxes from a large number of candidates to assemble a compact polygonal mesh model, and formulate the box selection as a binary linear programming problem. Our formulation favors to represent the scene with a compact assembly of boxes and meanwhile respects the input point cloud. The approach is designed to provide a tradeoff between data fitting and compactness of the final model. Thus, the results of our method are polygonal models with simplified geometric structures, which can be broadly applied in visualization, 3D mapping, geographic information system, and digital earth.

References

Arikan, Murat, Michael Schwärzler, Simon Flöry, Michael Wimmer, and Stefan Maierhofer. 2013. "O-snap: optimization-based snapping for modeling architecture." ACM Transactions on Graphics 32 (1): 6:1–6:15.

Furukawa, Yasutaka, Brian Curless, Steven M. Seitz, and Richard Szeliski. 2009. "Manhattan-world stereo." In Computer Vision and Pattern Recognition, 2009 IEEE Conference on, 1422–1429. IEEE.

Garcia-Dorado, Ignacio, Ilke Demir, and Daniel G Aliaga. 2013. "Automatic urban modeling using volumetric reconstruction with surface graph cuts." Computers & Graphics 37 (7): 896–910.

Gruen, A. 2008. "Reality-based generation of virtual environments for digital earth." International Journal of Digital Earth 1 (1): 88–106.

Gurobi. 2015. "Gurobi: Gurobi Optimization." `http://www.gurobi.com/`.

Hiep, V.H., R. Keriven, P. Labatut, and J. Pons. 2009. "Towards high resolution large-scale multi-view stereo." In Computer Vision and Pattern Recognition, IEEE Conference on, 1430–1437. Miami, US.

Kazhdan, Michael, and Hugues Hoppe. 2013. "Screened Poisson surface reconstruction." ACM Transactions on Graphics 32 (3): 29:1–29:13.

Lafarge, Florent, X. Descombes, J. Zerubia, and M. Pierrot-Deseilligny. 2010. "Structural approach for building reconstruction from a single DSM." Pattern Analysis and Machine Intelligence, IEEE Transactions on 32 (1): 135–147.

Li, Minglei, Liangliang Nan, Neil Smith, and Peter Wonka. 2016. "Reconstructing building mass models from UAV images." Computers & Graphics 54: 84–93.

Lin, Hui, Jizhou Gao, Yu Zhou, Guiliang Lu, Mao Ye, Chenxi Zhang, Ligang Liu, and Ruigang Yang. 2013. "Semantic decomposition and reconstruction of residential scenes from LiDAR data." ACM Transactions on Graphics 32 (4): 66:1–66:10.

Matei, Bogdan C., Harpreet S. Sawhney, Supun Samarasekera, Janet Kim, and Rakesh Kumar. 2008. "Building segmentation for densely built urban regions using aerial LiDAR data." In IEEE Computer Society Conference on Computer Vision and Pattern Recognition, IEEE.

Musialski, Przemyslaw, Peter Wonka, Daniel G. Aliaga, Michael Wimmer, Gool Luc van, and Werner Purgathofer. 2013. "A survey of urban reconstruction." Computer Graphics Forum 32.

Nan, Liangliang, Caigui Jiang, Bernard Ghanem, and Peter Wonka. 2015. "Template assembly for detailed urban reconstruction." Comput Graph Forum 35: 217–228.

Nan, Liangliang, Andrei Sharf, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2010. "SmartBoxes for unteractive urban reconstruction." ACM Transactions on Graphics 29 (4): 93.

Poullis, Charalambos, and Suya You. 2009. "Automatic reconstruction of cities from remote sensor data." In Computer Vision and Pattern Recognition, 2009 IEEE Conference on, 2775–2782. IEEE.

Rottensteinera, Franz, G. Sohnb, M. Gerkec, J. Wegnerd, U. Breitkopfa, and J. Jungb. 2014. "Results of the ISPRS benchmark on urban object detection and 3D building reconstruction." ISPRS Journal of Photogrammetry and Remote Sensing 93.

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

Vanegas, Carlos, Daniel Aliaga, and Bedrich Benes. 2012. "Automatic extraction of Manhattan-world building masses from 3D laser range scans." IEEE Transactions on Visualization and Computer Graphics 18 (10): 1627–1637.

Vanegas, Carlos A., Daniel G. Aliaga, and B. Benes. 2010. "Building reconstruction using manhattan-world grammars." In Computer Vision and Pattern Recognition, 2010 IEEE Conference on, 358–365. June.

Verdie, Yannick, Florent Lafarge, and Pierre Alliez. 2015. "LOD generation for urban scenes." ACM Transactions on Graphics 34 (3): 15.

Wang, Jinglu, Tian Fang, Qingkun Su, Siyu Zhu, Jingbo Liu, Shengnan Cai, Chiew-Lan Tai, and Long Quan. 2015. "Image-based building regularization using structural linear features." Visualization and Computer Graphics, IEEE Transactions on .

Xiao, Yong, Cheng Wang, Jing Li, Wuming Zhang, Xiaohuan Xi, Changlin Wang, and Pinliang Dong. 2015. "Building segmentation and modeling from airborne LiDAR data." International Journal of Digital Earth 8: 694–709.

Zhang, Liqiang, Chunming Han, Liang Zhang, Xiaokun Zhang, and Jonathan Li. 2014. "Web-based visualization of large 3D urban building models." International Journal of Digital Earth 7: 53–67.

Zhou, Qian-Yi, and Ulrich Neumann. 2010. "2.5D Dual Contouring: A robust approach to creating building models from aerial LiDAR point clouds." In Proceedings of the 11th European Conference on Computer Vision Conference on Computer Vision: Part III, Heraklion, Crete, Greece. ECCV'10. 115–128.

Zhou, Qian-Yi, and Ulrich Neumann. 2013. "Complete residential urban area reconstruction from dense aerial LiDAR point clouds." Graphical Models 75 (3): 118–125.