Semantic Decomposition and Reconstruction of Residential Scenes from LiDAR Data

Hui Lin*[1] Jizhou Gao*[1] Yu Zhou[2] Guiliang Lu[2] Mao Ye[1] Chenxi Zhang[1] Ligang Liu[3] Ruigang Yang[1]

[1]University of Kentucky [2]Nanjing University [3]University of Science and Technology of China

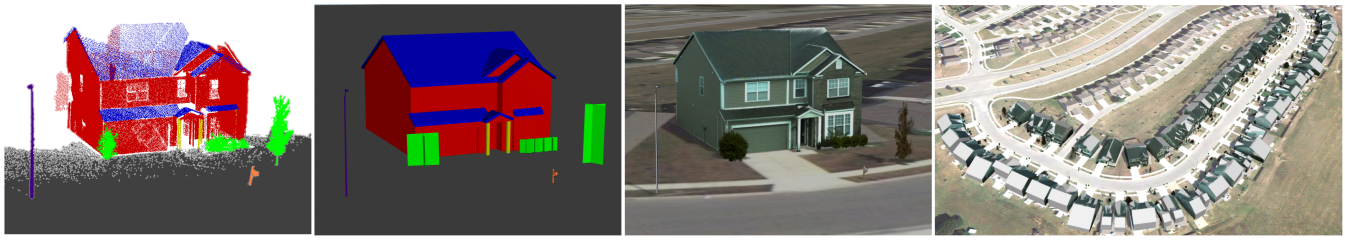

Figure 1: *Our reconstruction pipeline. From left to right: (a) semantically labeled 3D point cloud; (b) reconstructed objects using category-specific methods, including billboard trees, replaced common objects, and a building. The color-code on the building shows recognized different building parts; (c) textured 3D models on a ground plane, and (d) an overview of an automatically reconstructed large-scale scene.*

Abstract

We present a complete system to semantically decompose and reconstruct 3D models from point clouds. Different than previous urban modeling approaches, our system is designed for residential scenes, which consist of mainly low-rise buildings that do not exhibit the regularity and repetitiveness as high-rise buildings in downtown areas. Our system first automatically labels the input into distinctive categories using supervised learning techniques. Based on the semantic labels, objects in different categories are reconstructed with domain-specific knowledge. In particular, we present a novel building modeling scheme that aims to decompose and fit the building point cloud into *basic blocks* that are block-wise *symmetric* and *convex*. This building representation and its reconstruction algorithm are flexible, efficient, and robust to missing data. We demonstrate the effectiveness of our system on various datasets and compare our building modeling scheme with other state-of-the-art reconstruction algorithms to show its advantage in terms of both quality and speed.

Keywords: residential scenes, hierarchical representation, decomposition and reconstruction, symmetric blocks

Links: DL PDF WEB VIDEO

1 Introduction

3D reconstruction and modeling of urban environment, due to its vast applications in many areas, have long been an active topic in many research communities, from computer graphics, computer vision, photogrammetry, to remote sensing. Based on the type of input data (2D vs 3D), the reconstruction scale (single building vs. block/city), and the output models (facade, 2.5D, or full 3D), many algorithms and systems have been developed, enabling the production of 3D models better and faster.

Even though much progress has been made, as pointed out by a recent survey paper by Musialski et al. [2012], "automatic large-scale reconstruction remains an open problem." Existing methods either use airborne LiDAR data to generate 2.5D models that lack street-level details (e.g., [Poullis and You 2009a; Lafarge and Mallet 2011; Zhou and Neumann 2012]) or use ground-level images or LiDAR for street-side modeling only (e.g., [Xiao et al. 2009; Frahm et al. 2010]). While fusion of ground-and-airborne data can produce the most complete model (e.g., [Frueh and Zakhor 2003]), the fidelity of the model could be improved. In addition, the output models from existing approaches are usually a set of polygons (sometimes per building) with little semantic labeling. Editing models is difficult.

In this paper, we present a comprehensive system to reconstruct detailed 3D models with semantic labels. Starting with LiDAR data with registered color images, we first segment the unorganized 3D points into distinctive categories including houses, plants, street lights, etc. Then for each category we develop unique solutions to reconstruct its 3D model, taking advantage of the prior information about this particular category. For example, common objects, such as street lights, are replaced by similar 3D models found on the Internet. Plants are modeled with billboard techniques, which are known to be visually convincing. Special emphasis is put on the reconstruction of houses. The typical properties of buildings, such as piece-wise planar structures, convexity, and symmetry, are used to develop an efficient reconstruction algorithm that can deal with incomplete data. The outcome of our system is a set of visually complete 3D models consisting of common static objects in an urban scene, including not only houses, but also plants, street lights, mailboxes, etc. Each object has its own semantic labeling.

The primary target of our system is residential areas. Many of existing modeling approaches, in particular those generating models with high details and rich textures, deal almost exclusively with multiple-story or high-rise buildings (e.g. [Müller et al. 2006; Nan et al. 2010]). These buildings are typically found in downtown and highly populated urban areas. The structural details are repetitive and regular, from which user-defined grammar rules can be used to

*The first two authors contributed equally to this work.

ACM Reference Format
Lin, H., Gao, J., Zhou, Y., Lu, G., Ye, M., Zhang, C., Liu, L., Yang, R. 2013. Semantic Decomposition and Reconstruction of Residential Scenes From LiDAR Data. ACM Trans. Graph. 32, 4, Article 66 (July 2013), 10 pages. DOI = 10.1145/2461912.2461969 http://doi.acm.org/10.1145/2461912.2461969.

DOI: http://doi.acm.org/10.1145/2461912.2461969

regularize the reconstruction to generate clean output. This kind of repetition is not available in residential areas. In addition, thanks to popular modeling tools such as Google Sketchup, many of these metropolitan areas already have 3D models. Admittedly, those low-rise houses in suburban/residiental areas are less glamorous to work with from a visualization standpoint, but they are equally important from a simulation or city planning standpoint since they are literally everywhere. For these areas, automation is important because of the large scale. Prior approaches to reconstruct these areas are developed with aerial data [Dorninger and Pfeifer 2008; Lafarge et al. 2010], which produce no details below roof. The system presented in this paper fills the void of automatic decomposition and reconstruction of residential areas from ground-based data.

The major contribution of our paper lies in a novel modeling scheme for low-rise/residential houses. A house is represented as a hierarchical collection of structures at different scales and locations: planar primitives, patches and symmetric and convex blocks. The task of reconstruction is to find a possible combination of planar primitives that best explains the input data. Results have shown that this method is powerful to model a variety of houses, even in the presence of significant missing data. In addition, it runs efficiently. After segmentation, it only takes about 15 minutes to reconstruct all the houses shown in Figure 1, an average of less than 20 seconds per house.

2 Related Work

The problem of building modeling has received considerable attention in recent years, probably due to the ubiquitous digital cameras and more availability of active 3D sensing devices, such as LiDAR (Light Detection And Ranging) devices. Given the large body of work in this broad topic, it is beyond the scope of this paper to provide a comprehensive review. Interested readers are referred to two recent survey papers by Vanegas et al. [2010] and Musialski et al. [2012].

Image-Based One topic of emphasis is to reconstruct 3D buildings from images or video sequences(e.g., [Werner and Zisserman 2002; Akbarzadeh et al. 2006; Xiao et al. 2008]). One of these pioneering papers in this area is the Facade system [Debevec et al. 1996], in which an operator interactively selects corresponding lines between images and building primitives (blocks etc.), and the system automatically estimates the camera pose and refines the primitives to fit the images. Nowadays, the typical approach is to use structure from motion (SfM) techniques to estimate the camera motion as well as a set of sparse 3D points, followed by either user interaction(e.g., [Sinha et al. 2008]) or dense stereo matching to generate the final model(e.g., [Frahm et al. 2010]).

LiDAR-based Alternatively the input can be 3D point cloud from LiDAR devices; the focus is usually to refine the scanned data and create a more usable mesh or parametric model(e.g., [Frueh et al. 2005; Poullis and You 2009b]). This is also the main focus of our paper. Recent papers in this category often take advantage of the repetition and symmetric structures in buildings. Müller et al. developed an automatic approach for facade reconstruction with CGA shapes using ortho-rectified photos [Müller et al. 2006]. It works well with highly regular, repetitive facades. Later the system was extended to allow regular photos that have perspective distortions [Van Gool et al. 2007]. Using 3D scan as input, the *Smart-Boxes* technique focuses on high-rise buildings with many repetitive structural elements [Nan et al. 2010]. From a user-selected structural element, the system automatically finds similar copies in the input 3D point cloud. These repetitive elements are merged and refined, leading to a better final model. Automatic detection of symmetric or repetitive patterns has also been developed, in either 2D [Schaffalitzky and Zisserman 1999; Wu et al. 2010], 3D [Pauly et al. 2008; Bokeloh et al. 2009], or the combination of both [Li et al. 2011b]. They typically make a strong assumption about the layout of the symmetric or repetitive patterns, usually on a rectilinear grid, to optimize for facade processing. Similar assumptions are also applied to interior reconstruction with great success [Xiao and Furukawa 2012]. Given our emphasis on residential houses, we do not make such an assumption. Li et al. [2011a] detects global regularities among primitives to better fit to scanned data but they cannot handle completely missing surface patches like the roof area in our case. More recently, Vanegas et al. [2012] presents a system to reconstruct Manhanttan-World Building masses from 3D range scans. Assuming the building is made of axis-aligned boxes, it is able to produce water-tight building models in the presence of significant missing data. The main limitation is that it is unable to handle slanted surfaces.

Large-Scale In the area of automatic reconstruction at a large scale, approaches using ground-based data, either images (e.g., [Xiao et al. 2009; Frahm et al. 2010; Irschara et al. 2012]) or 3D point clouds [Frueh et al. 2005], usually focus only on the facade, or whatever can be captured. On the other hand, approaches with airborne data generate 2.5D models since only the roof is captured (e.g., [Dorninger and Pfeifer 2008; Poullis and You 2009a; Lafarge et al. 2010; Lafarge and Mallet 2011; Zhou and Neumann 2011]. In both scenarios, few assumptions are made about the scene, therefore, they are usually more robust about different building/scene types. The downside is that the model is only good from the viewpoint it is originally captured. In our targeted setup – houses of a few of stories, we can capture parts of both, but neither is complete. In order to complete the model, we make a few common assumptions. These assumptions also allow us to provide semantic labeling to the final model. As we will demonstrate later, our approach is able to *handle significant missing data*, in particular in the roof area. There are approaches that combine both ground and aerial data sources (e.g. [Frueh and Zakhor 2003; Karantzalos and Paragios 2010]) for large scale reconstruction, our approach can also benefit from the more data coverage.

As pointed out by other researchers [Musialski et al. 2012], it is difficult to directly compare these different reconstruction methods since they are all developed under different context with different emphasis. Nevertheless we have compared our results with two state-of-the-art algorithms. The first uses piecewise-planar assumption about the scene geometry [Chauve et al. 2010], which is also the foundation for our house reconstruction scheme. The second focuses on building 2.5D models for large scale reconstruction from aerial scans [Zhou and Neumann 2010]. Both methods are automatic. The comparisons demonstrate the advantage of our approach for the task of reconstruction of residential houses from ground-based 3D point data.

3 Algorithm Insight and Overview

As we have discussed before we focus on automatic modeling of stand-alone buildings, such as single-family houses in suburban areas. As shown in Figure 2, there are many different building styles. Compared to high-rises in downtown, the symmetry and repetitiveness are less dominant. This is particularly true for recently built houses with the "neo" style that combine different historic styles with new features. Nevertheless, we can see that even though there are many variations in building styles, the fundamental structures are the same: *a combination of convex blocks with tilted roofs*. The variations in styles are usually manifested in terms of construction materials, level of decorations, layout of windows, and slopes of roofs, etc. None of them changes the underlying structural semantics of a building.

Figure 2: *Pictures of various building styles in chronical orders. From left to right: (a) American Colonial styles between the 17 and 19th century; (b) Neoclassical style (early 19th century) that reflects classic ideas of order and symmetry; (c) Victorian house styles (late 19th century) with elaborated decorations; (d) Bungalow Styles in the early 20th century, compact, economical and informal; (e) "Neo" house styles (recently built homes) that borrow details with historic styles and combine them with modern features.*

Given the large scale point clouds acquired in the outdoor scan, our first step is to automatically segment and classify the input point clouds into several classes (e.g., houses, street lights, mailboxes, etc.) as described in Section 4. As our main focus is the residential buildings, the key to our modeling pipeline is to decompose the input point cloud of a house scan into basic *blocks* using a few pre-defined reconstruction constrains, namely, *planarity*, *symmetry*, and *convexity*. This is described in detail in Section 5. With these basic blocks in place, details such as columns, and eaves are further extracted and processed. Next, the textures are added to the reconstructed model (Section 6.1). The output of house reconstruction is a complete textured model composed of a hierarchical tree of building structures with six semantic labels: walls, roofs, columns, eaves, garage doors and chimneys. Finally, other objects (e.g., trees, mailboxes, etc.) in the landscape can also be modeled (Section 6.2).

4 Semantic Segmentation

The first stage in our system is to segment the input point cloud into semantically distinctive groups. This is done in two steps. The first is to label the input into *categories*, such as houses or plants. Then points labeled as houses are further refined into more detailed structural *classes*, such as roofs or walls. We use supervised machine learning techniques to perform these tasks.

4.1 Categorization

We define the following categories in our current implementation: *houses*, *plants*, *mailboxes*, *street lights*, *waste bins*, *cars* and *ground*, which are common objects seen in a residential area. From our scanned data, we manually label a section as the training data set and adopt the semantic segmentation approach similar to Zhang et al. [2010]. It should be mentioned that other alternatives such as [Golovinskiy et al. 2009] and [Xiao and Quan 2009] can also be applied to perform this task.

In order to deal with our point cloud data, we have made a few changes to the original method. First, we group nearby points into *superpoints*, analogues to the *superpixel* concept in image segmentation. Second, we use the Adaboost classifiers. Third, we introduce a new concept called *super-region* to differentiate objects with similar superpoint features but at different scales, such as houses and waste bins. For each super-region, we compute the following features: height, volume, area, planarity difference, and length. These additional features are added to its superpoints. The augmented feature vectors are used for training. After the classification stage, the result is further grouped into connected components. More technical details can be found in the supplementary material.

4.2 Segmentation of House Point Cloud

For each point set labeled as a house, it will be further segmented into different *classes*, including *columns*, *roofs* and *walls*. The same segmentation method in Section 4.1 is adopted with one additional superpoint feature: surrounding emptiness, which measures the number of points within a neighboring bounding box around each superpoint center. This feature is mainly used to identify columns.

5 House Modeling

In this stage each labeled house (or building) from Section 4 is reconstructed to a polygonal model based on a novel algorithm that consists of several main procedures: a top-down decomposition, a bottom-up grouping and reconstruction, and a refinement process. For each building, the algorithm takes its segmented point cloud as input, where each point is represented by (x, y, z, c) with the first three components being its spatial coordinates and the last one representing its class. The algorithm generates a hierarchical representation of the building (described in Section 5.1) with refined semantic labels (whose definition can be found in Section 5.3).

5.1 Building Representation

In our modeling algorithm, each potentially complicated building structure is represented by a combination of simple *block*s that consist of various parameterized primitives. Even though there are many variations in building styles, the basic blocks are quite similar: each block is a spatially symmetric box that consists of two or more spatially symmetric pairs of *patches*, where each patch is a conjunction of a roof and its aligned vertical wall. Therefore, our algorithm introduces a general, hierarchical-tree based representation for suburban buildings:

$$Building = \bigcup_i Block_i$$
$$Block = \bigcup_j Pair_j \, of \, patches$$
$$Patch = \{Roof; Wall\} \text{ or } \{\emptyset; Column\}.$$

Note that a patch allows at most one of its members (i.e., roof or wall) to be empty set ($\emptyset$). For instance, a box can be represented as two pairs of patches with its roof being $\emptyset$; a gable can be represented as two pairs of patches where one of the roofs is $\emptyset$; a column block can be represented as two pairs of patches with only column primitives. Our block modeling algorithm aims at finding a plausible decomposition of a single building. See Figure 3 for illustration.

5.2 Reconstruction Constraints

In order to find a plausible decomposition automatically from unorganized input points, we define three constraints based on common building structures.

Planarity Constraint This is straightforward: each basic block is essentially a set of planar primitives in which each plane (wall or roof) P_i is determined by its orientation vector (normalized) $n_i \in \mathbb{R}^3$, position in $\mathbb{R}^3$ $p_i \in \mathbb{R}^3$, and its boundary set $B_i \subset \mathbb{R}^3$, namely, $P_i = \langle n_i, p_i, B_i \rangle$, where the boundary set B_i of P_i is the

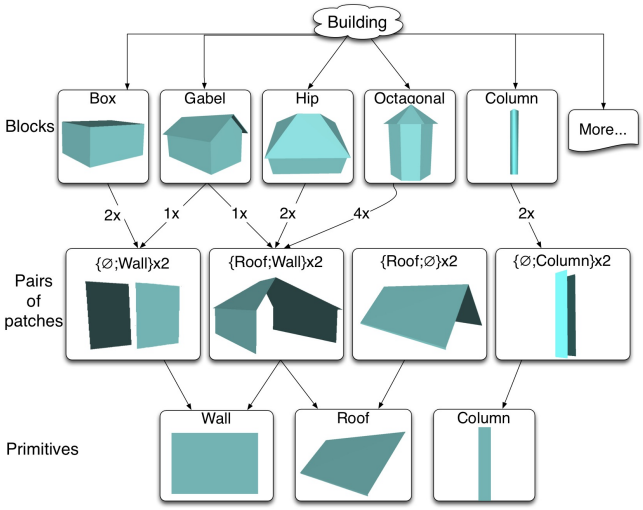

Figure 3: *Hierarchy-tree representation of buildings.*

intersection of P_i and the bounding box of all data points contained in P_i. The planar primitives are extracted using a robust plane fitting algorithm based on RANSAC [Schnabel et al. 2007].

Block-Level Symmetry Constraint As we discussed before, the regularity of a single building is not as obvious as high-rise buildings. However, the symmetry of its basic blocks still holds. Thus, any asymmetric structure can be further decomposed into multiple symmetric blocks. In the decomposition procedure, therefore, our approach requires each basic block of a building structure be symmetric with respect to x- or y-*principal direction* of the building structure. Here we define the three mutually orthogonal principal directions of each building structure as follows: z-direction as the direction perpendicular to the ground (i.e., $(0, 0, 1)$); x- and y-principal directions as primary directions of the two largest clusters of planar orientations. For LiDAR data acquired by ground-base devices, x-principal direction is chosen to be the orientation of the front wall which can be determined by the driving path of the scanner. For LiDAR data captured from air, x-principal direction is chosen to be the direction of the largest cluster of planar orientation (except for z-direction).

According to the preceding requirement, two types of block symmetry are defined:

(1) *parallel-symmetry*: planes $P_i = \langle n_i, p_i, B_i \rangle$ and $P_j = \langle n_j, p_j, B_j \rangle$ are parallel-symmetric if $n_i \cdot n_j = \pm 1$ and the projection of each planar on the other overlaps the other planar.

(2) *intersecting-symmetry*: planes $P_i = \langle n_i, p_i, B_i \rangle$ and $P_j = \langle n_j, p_j, B_j \rangle$ are intersecting-symmetric if the following conditions are satisfied: (1) $|n_i \cdot n_j| < 1$; (2) define $n_l = n_i \times n_j$, then either $x \cdot n_l = 0$, or $y \cdot n_l = 0$, where x and y are the two principal directions; (3) the projections of both planes onto the plane determined by vectors n_l and $z = (0, 0, 1)$ overlap.

To ensure the symmetry of basic blocks, we further require that for each pair of patches $\langle \{Roof_1, Wall_1\}, \{Roof_2, Wall_2\} \rangle$, $Roof_1$ and $Roof_2$ be *intersecting-symmetric*, and $Wall_1$ and $Wall_2$ be *parallel-symmetric*.

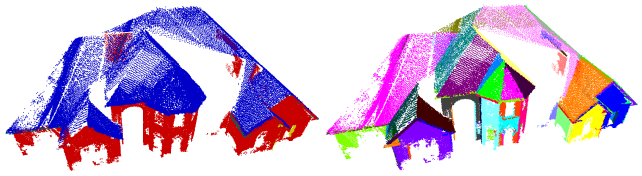

Figure 4: *Planar primitives extracted by RANSAC. (left) Input labeled point cloud (red as walls; blue as roofs); (right) plane extraction result, colorized planes represent different planar primitives. The grey points are labeled as outliers.*

Block-Level Convexity Constraint Our modeling algorithm requires convexity constraint of basic blocks. Specifically, We require that all basic blocks of the building structure be *sub-convex*. Let K be a subset of $\mathbb{R}^3$ that is the union of a finite number of planar primitives. For example, K can be a basic block. K is sub-convex if it is a simply-connected subset of the boundary of a simply-connected *convex* subset of $\mathbb{R}^3$. Here the concept of being convex is standard, namely, a subset G of $\mathbb{R}^3$ is convex if for all $u, v \in G$, the line segment $\overline{uv} \subset G$. Intuitively, a basic block is sub-convex if its projection onto a carefully chosen plane can have the shape of letter "U", but not "Z".

5.3 Construction of the Connection Graph

In order to generate a plausible decomposition, a global connection graph C (undirected, weighted) is constructed from all planar primitives. The algorithm optimizes the decomposition result by maximizing the connection score that is defined on the connection graph C as the weights of its edges.

Specifically, each vertex in C represents a planar primitive in the building structure and is tagged by a semantic label chosen from eight possible choices: *ground*, *wall*, *roof*, *column*, *garage door*, *chimney*, *eave* and *unknown*. The primitives are initially labeled based on the point cloud segmentation result discussed in Section 4.2: during the pre-reconstruction stage each planar primitive is initially labeled as one of ground, column, roof, and wall according to the class of its majority points, or unknown if the classes of its points are too diversified. After the house is reconstructed, the garage door, chimney and eave primitives will be detected and labeled as described in Section 5.5.

An edge between two vertices indicates that the two primitives represented by those vertices are spatially connected. Specifically, for two parallel primitives, the vertices representing them are connected by an edge if (1) they are co-planar and (2) the shortest distance between the two planar point sets is less than a predefined threshold (set to $0.5m$ for our experiments); for two non-parallel primitives, the vertices representing them are connected by an edge if the shortest distances of the two planar point sets to the intersection line are both less than a threshold (set to $0.75m$ for our experiments).

For each edge e_{uv} connecting vertices u, v, its connection score, or weight, W_{uv}, is calculated as

$$W_{uv} := \Psi(D_{uv}) + (-\infty)\chi_B(u, v)$$

where Ψ is a predefined monotone decreasing step function, D_{uv} is the spatial distance between the primitives represented by u, v, and χ_B is an indicator function whose value is 1 iff u, v satisfy *backward relation*, where two roof primitives form a "V" shape, so they do not fit in a basic block and thus are forbidden from being in the same block even if they satisfy the convexity constraint.

The connection graph C constructed according to the preceding description serves as the basic data structure for the progressive decomposition and reconstruction algorithms, which aim at finding the plausible cut of the connection graph so that the sum of the connection scores on vertices inside each block is locally maximized.

5.4 Algorithm: Progressive Decomposition and Reconstruction

A hierarchical tree (Figure 3) is generated based on the connection graph C, which is described in Section 5.3:

(1) its root as the building structure,
(2) its first level nodes as the basic blocks,
(3) its second level nodes as the pairs of patches,
(4) its third level nodes (leaves) as primitives with semantic tags.

This data structure is obtained in a progressive (iterative) manner, where for each iteration we first decompose the building structure using the greedy grouping method and then reconstruct and abstract the parameterized block models (Algorithm 1). For each step, the hierarchical tree is updated with new blocks and eventually it reaches a plausible state. The algorithm works with the decomposition and reconstruction procedures as two subroutines which will be elaborated more in the next two paragraphs. In particular, our algorithm can also effectively handle incomplete data during the reconstruction procedure. Figure 5 illustrates one iteration of the algorithm while handling missing data. Figure 6 shows the final result by the progressive decomposition and reconstruction algorithm.

Algorithm 1 Hierarchical Tree Generation

Input: a set of primitives $PS := \bigcup_i \{P_i\}$
Output: a hierarchical tree T

```
while PS ≠ ∅ do
    P_0 ←largest primitive in PS
    G ← {P_0}
    Decomposition: Update G using a greedy grouping algorithm
    Reconstruction: Update hierarchical tree T from G
    for P_j ∈ G do
        Remove P_j from PS
    end for
end while
```

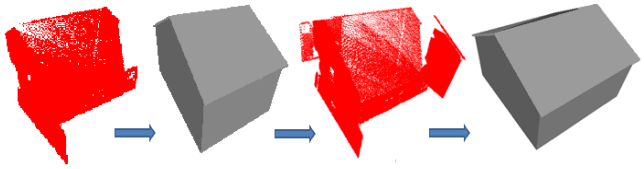

Figure 5: *.Greedy grouping and reconstruction with missing data handling. From left to right: (a) grouped primitives, (b) reconstructed mesh model, (c) extend by handling missing data and (d) final block model.*

Decomposition In order to find a plausible decomposition of the building, a greedy grouping method is used, which aims at finding the largest set of the planar primitives that are potentially part of the same basic block:

(1) start from a planar P_0 (roof planar or front wall planar), initialize group $G = \{P_0\}$;

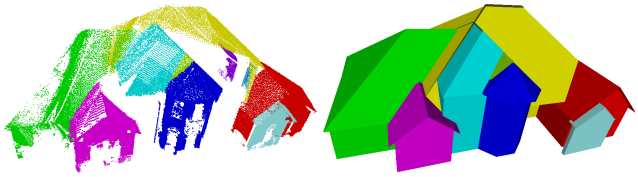

Figure 6: *Decomposition and Reconstruction result of an incomplete scan. (left) Decomposition of the building structure; (right) reconstruction result. Each color represents an individual block.*

(2) create a candidate set as the set of all planar primitives that connect to any planar element in G; If the candidate set is empty or all primitives in such set are already grouped in a block, then terminate the decomposition loop and start the reconstruction procedure.
(3) remove all primitives that do not satisfy the symmetry and convexity constraints;
(4) sort the remaining primitives with respect to their connection scores, and group the one with the highest connecting score into G; exit if there is no remaining primitives;
(5) goto step (2).

Reconstruction After a group of primitives is found using the preceding decomposition algorithm,

(1) the roof-wall patches are extracted and parameterized to form the block representation;
(2) a surface mesh is generated as the block model;
(3) such parameters are then adjusted by maximizing the number of points that "fit" into the block model, where a point is said to "fit" into a block model if its shortest distance to the surface of the block model is less than a threshold (set to 0.15m for our experiments);
(4) if one or several patches are missing from grouped primitives, the algorithm first fulfills the missing roof or wall with a virtual primitive, and then "extend" the block model to handle the incomplete data (which will be elaborated in the next paragraph);
(5) then new planar primitives are formed from the unfit points;

When block reconstruction is finished, we proceed to the next iteration with a new greedy grouping process of decomposition until no basic blocks can be reconstructed from remaining primitives.

Incomplete Data Handling Due to the limit of acquisition devices and building occlusions, the acquired LiDAR data is usually incomplete. Based on the assumption that the building structure is watertight and symmetric, our algorithm handles such incomplete or missing data during the reconstruction procedure in the following way:

(1) for aerial data, where the structures under roof are difficult to capture, our algorithm handles the missing walls by automatically fulfilling the Roof-Wall patches with aligned virtual walls;
(2) for terrestrial data, the side/back walls and roofs are likely to be missing; if the reconstruction algorithm detects such missing, it extends the block model by (a) generating a virtual primitive with respect to the symmetry constraint, (b) pushing it out along its normal direction until an actual planar primitive overlaps it, and (c) updating the block model if such a planar primitive is found, or keeping it as a node in the hierarchical tree otherwise. Figure 5 shows an example.

5.5 Refinement of Primitive Labels and Model

After the building structure is decomposed and reconstructed, the algorithm traverses all the primitives and detects garage doors, chimneys, and eaves based on the spatial relation between the candidate primitive and its neighbors within the same reconstructed block. Once one of the preceding three types has been detected the model will also be refined accordingly. See below for details.

(1) *Eave primitive*: a wall or unknown primitive is detected as an eave if (a) it is vertically thin (less than 0.3m in our experiments) and (b) adjacently beneath a roof; for each detected eave its corresponding roof primitive is stretched down vertically by the same size of the eave so that a solid roof is formed.

(2) *Garage Door*: a pair of walls is labeled as a garage-door primitive if (a) these two walls are parallel-symmetric and their representing vertices are connected in the connection graph C and (b) the point cloud of one of walls forms a "$\prod$" shape whose hollow overlaps the projection of the other wall onto this wall; for each detected such pair a solid garage door is formed by creating a box with these two walls as its front and back and removing the underneath hollow which corresponds to the region on the front wall that occupies little point cloud.

(3) *Chimney Extraction*: a wall is detected as chimney if (a) the block it belongs to is a box structure and (b) its highest position on z-direction is higher than that of the entire building structure minus $1m$; for each detected chimney primitive, it is lowest bound on z-direction will be set to the ground level.

6 Model Enhancement

We also present several techniques to enhance the visual realism of final models, including an effective texture snapping algorithm that reduces artifacts caused by inaccuracy in 3D/2D registration, automatic billboard tree/shrub reconstruction suitable for large-scale light-weight modeling of plants, and object replacement using models available on the Internet. All of these combined lead to rich and clean final models beyond just buildings. We briefly summarize in the followings, and more technical details can be found in the supplementary material.

6.1 Texture Mapping

To compute texture maps for a reconstructed building model, we first automatically find the nearby camera views that capture the model, and then back-project the views to the planar surfaces of the model. However, images and point clouds may not be perfectly aligned, and a model is only approximately reconstructed. Thus, texture maps generated from the direct back projection would produce noticeable misalignment. In order to alleviate these errors, we propose an algorithm of content-preserving warps inspired by [Liu et al. 2009] based on 2D-3D line correspondences. We further fuse the multiple overlapping views using a multi-label MRF energy minimization framework similar to [Sinha et al. 2008].

6.2 Landscape Modeling

Plants, including trees and shrubberies, are integral parts of our living environment. We develop a fully automatic method suitable for large-scale reconstruction by using the light-weight billboard representation for visually-plausible plant models. Our plant model consists of two orthogonal planes and one billboard image. Other frequently occurring static objects in residential landscape such as mailboxes and street lights are often hard to directly reconstruct using simplified geometric models from the cluttered, incomplete and noisy data. Inspired by the recent success of model replacement applied to indoor scene modeling (e.g., [Shao et al. 2012]), we download the similar models from Google 3D Warehouse, and use PCA to estimate global scaling and an initial pose between the models, and the recognized raw points and further align them using iterative closest point (ICP). In this way, points from a categorized object (e.g., a mailbox) are replaced by the corresponding model.

7 Experimental Results

We have implemented our algorithms and mainly tested on ground-based LiDAR scans. Our ground-based LiDAR data are acquired by a mobile LiDAR scanning unit and texture/color information is captured with a high-resolution panoramic video camera. The scanning trajectory is recorded with multi-channel GPS and high-precision inertial measurement unit (IMU). The 2D and 3D data are already registered (though the accuracy is not consistent) and geo-referenced. Figure 7 shows two datasets from our scanning platform. The top (*Div-A*) is a upscale residential area (containing over 150 million points) while the bottom (*Div-B*) is an average residential area (containing over 183 million points) that was newly built.

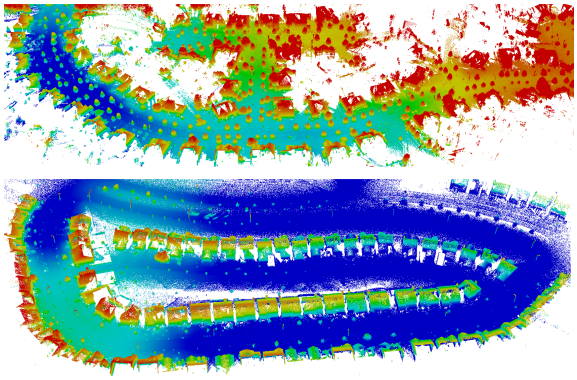

Figure 7: *Ground-based LiDAR Datasets: (top) a high-end subdivision (Div-A); (bottom) an average subdivision that is newly built (Div-B).*

7.1 Results on Semantic Segmentation

In order for training and evaluation, we have labeled both *Div-A* and *Div-B*. We used *Div-A* for training and *Div-B* for evaluation. One segmented scene is shown in Figure 1(a). The precision and recall of automatic segmentation is shown in Table 1. Compared to previous semantic segmentation methods (e.g., [Zhang et al. 2010]), our accuracy is noticeably higher, partially due to our high-quality input. We also report the precision (P) and recall (R) of the house classes: *walls* ($P = 94\%$, $R = 98\%$), *roofs* ($P = 94\%$, $R = 81\%$) and *columns* ($P = 72\%$, $R = 87\%$). *It should be emphasized that all of our subsequent modeling results use only semantic labels from the classifier's output, not the ground-truth*. In addition, our house modeling is not sensitive to mislabeling since it is highly regularized by various constraints.

7.2 Reconstruction Results

We show modeling results on *Div-A* and *Div-B* datasets. Since the symmetry and convexity of the basic blocks are the only assumptions we make, our algorithm is capable of modeling a variety of

Table 1: *Precision and recall of semantic segmentation.*

real \ predict (precision)	car	ground	house	mailbox	plant	road sign	streetlight	waste bin
car	0.68	0.00	0.00	0.00	0.00	0.01	0.01	0.10
ground	0.02	0.89	0.00	0.00	0.00	0.00	0.00	0.00
house	0.02	0.01	0.88	0.00	0.09	0.00	0.00	0.00
mailbox	0.00	0.00	0.00	0.91	0.00	0.00	0.00	0.04
plant	0.24	0.10	0.12	0.05	0.91	0.04	0.02	0.14
road sign	0.00	0.00	0.00	0.00	0.00	0.91	0.05	0.01
streetlight	0.00	0.00	0.00	0.03	0.00	0.04	0.92	0.00
waste bin	0.04	0.00	0.00	0.01	0.00	0.00	0.00	0.71

real \ predict (recall)	car	ground	house	mailbox	plant	road sign	streetlight	waste bin
car	0.70	0.00	0.04	0.00	0.24	0.00	0.00	0.02
ground	0.02	0.85	0.08	0.00	0.05	0.00	0.00	0.00
house	0.00	0.00	0.83	0.00	0.17	0.00	0.00	0.00
mailbox	0.01	0.01	0.01	0.85	0.10	0.00	0.00	0.02
plant	0.00	0.00	0.06	0.00	0.94	0.00	0.00	0.00
road sign	0.04	0.00	0.11	0.00	0.12	0.57	0.09	0.07
streetlight	0.00	0.00	0.01	0.06	0.50	0.01	0.42	0.00
waste bin	0.17	0.00	0.00	0.01	0.02	0.00	0.00	0.80

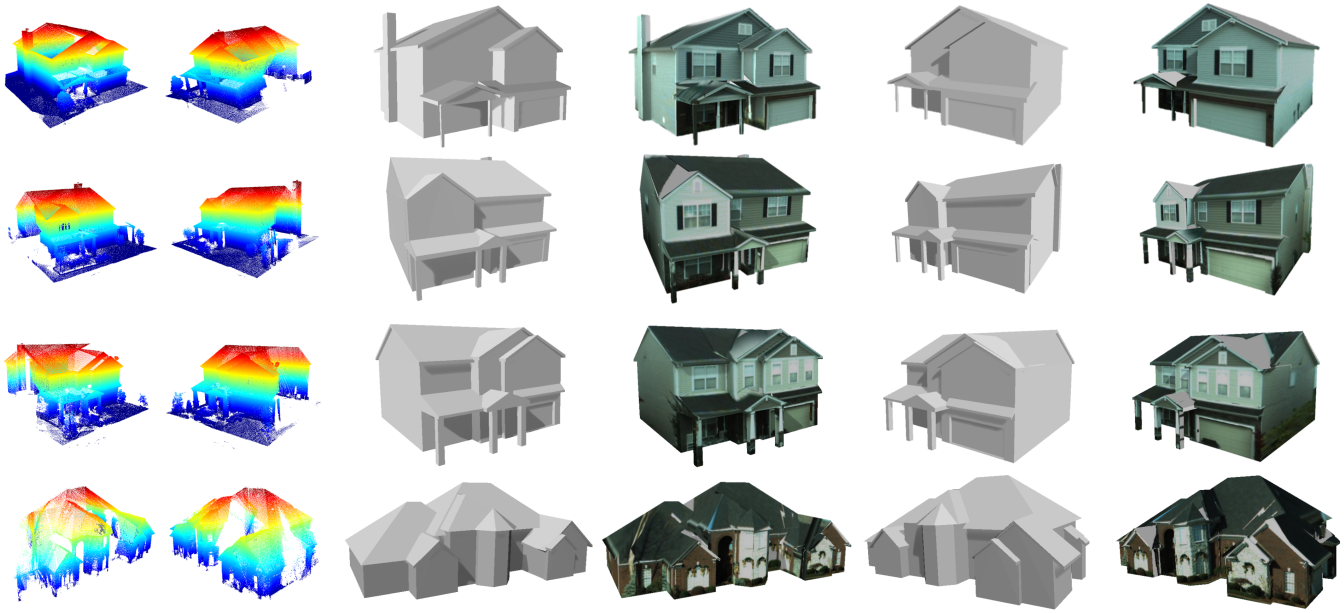

Figure 8: *Results on ground-based LiDAR datasets. Each row shows the result of one house. 1st and 2nd columns: point clouds color coding based on height in two perspectives, 3rd and 5th columns: reconstructed models, and 4th and 6th columns: reconstructed models with textures.*

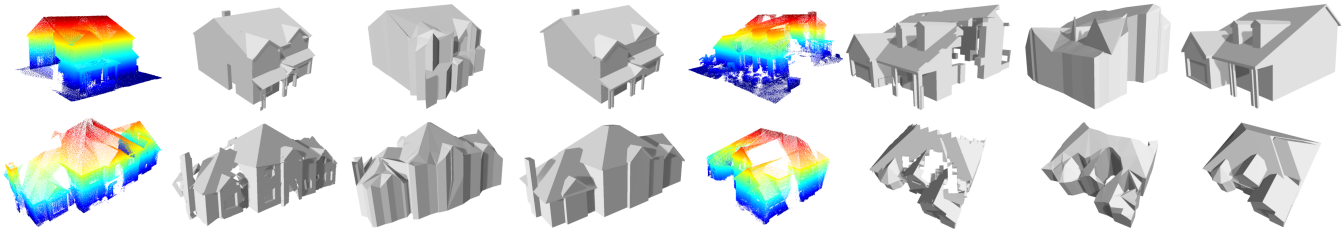

Figure 9: *Qualitative evaluation on three building reconstruction algorithms. Four groups of comparison results are shown. In each group, from left to right: point clouds, results by Piecewise Planar Reconstruction, results by 2.5D Dual Contouring, and results by our approach.*

houses and dealing with significant missing data. As shown in Figure 8 (1st, 2nd and 3rd rows), Figure 9 (1st rows) and Figure 11, we correctly model the houses composed of several gables and box structures at different scales, location and nested structures, and with missing point scan at various levels.

All houses in Div-B are automatically reconstructed together with other categorized and modeled objects such as trees and mailboxes in the landscape can be seen in the accompanying video. Figure 1(d) shows the overview of the large-scale reconstruction results overlaid on the geo-registered satellite image.

More complex structures and complicated composition of building blocks are often found in *Div-A*. As shown in Figure 8 (4th row) and Figure 9 (2nd row), the houses consist of multiple gables, hipped structures and even octagonal shapes. Our algorithm can successfully identify and reconstruct those blocks in a complex configuration from severe missing data. For example one tilted roof of the hipped structure is completely missing in Figure 9 (2nd row), but benefited from the symmetric block constraint, we can still reconstruct the complete structure from the partial scan. Figure 10 shows an overall view of the reconstructed *Div-A* dataset without textures.

Figure 10: *An overall view of reconstructed houses in Div-A.*

7.3 Comparisons

We have compared our house reconstruction approach with two automatic state-of-the-art algorithms: Piecewise Planar Surface Reconstruction (PPSR) [Chauve et al. 2010] and 2.5D Dual Contouring (2.5D DC) [Zhou and Neumann 2010]. We have implemented the PPSR algorithm ourselves and use Zhou and Neumann's provided implementation of 2.5D DC to perform the evaluation. Figure 9 shows the qualitative evaluation on four houses varying at surface complexity and scan completeness. The 2.5D DC method, which successfully works on airborne LiDAR data, can only reconstruct rough geometric shapes but lacks ground-level details when data are more or less complete, especially at roofs. However, it creates holes when the roof scan is incomplete. The PPSR method is a general surface reconstruction method, which only regularizes the input assuming piece-wise smoothness. Therefore, it can reconstruct more surface details, but cannot well handle missing data or low-density scan well. Our algorithm is high-regularized and therefore robust to both cases. *The comparison results in a full* 360° *view can be seen in the accompanying video*. For quantitative evaluation, we use the point to surface distance as the error metric and visualize in Figure 11. The overall reconstruction error is 0.0290 ± 0.0865 meters (PPSR), 0.3591 ± 0.2972 meters (2.5D DC) and 0.0598 ± 0.1174 meters (Ours). Thus, our method can not only get quantitatively small fitting error but also achieve the most visual realism of reconstruction results.

Our house modeling method can also be applied to airborne LiDAR data. We manually cut two buildings from the *Wright-State-100* [1] airborne LiDAR scans. Our learned classifiers on roofs and walls can still generate good results on the cut-out building data. The airborne LiDAR data often lack details under roofs or even the entire walls, however our method is still capable of reconstructing the complete models as shown in Figure 12. We are aware that a recent method that uses global regularities [Zhou and Neumann 2012] can probably generate similar results in this data set where the roof is completely scanned. But that method, unlike ours, is designed to work with only aerial scans.

[1] http://voyager.makai.com/Workspace_web.php

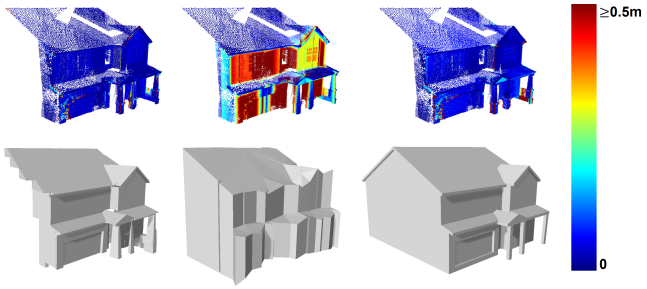

Figure 11: *Quantitative evaluation on three building reconstruction algorithms. From left to right, results by Piecewise Planar Reconstruction, results by 2.5D Dual Contouring and results by our method. Top: point clouds color-coding based on point to surface distance. Bottom: the corresponding reconstruction results.*

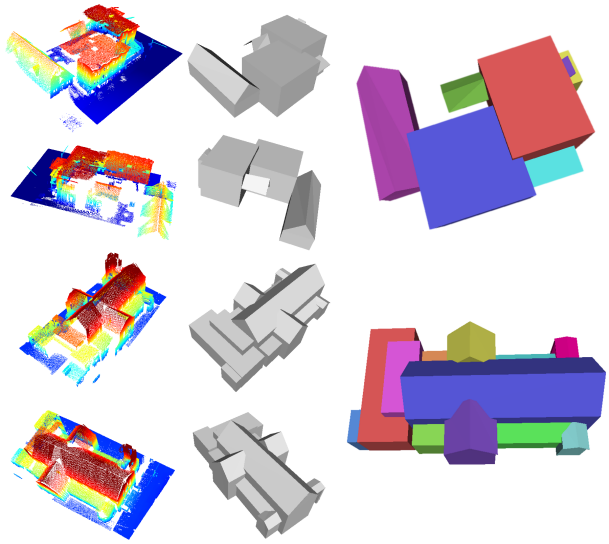

Figure 12: *Results using the Wright-State-100 airborne dataset. From left to right: point cloud, reconstructed models by our method, and the color-coded model showing the identified and reconstructed block structures of our method.*

7.4 Timing Performance

Our semantic segmentation algorithm takes about 40 minutes for labeling *Div-B* including feature extraction, classification and segmentation. This residential subdivision includes 53 houses.

Our geometry processing pipeline is very efficient. An average house in the subdivision datasets contains between 350K to 600K points. The processing time is around 15 to 22 seconds, in which RANSAC plane fitting takes half of the time. For the Wright-State-100 dataset that contains larger buildings with over 2 million points per building, the run time increases to 70 seconds per house. It should be noted that these timing numbers are based on unoptimized Matlab implementation running on a Intel i7 processor with 4GB of RAM. Compared to the PPSR algorithm that takes about

15-20 minutes per house, our average per-house reconstruction time is about two minutes (including semantic segmentation), almost one order faster. We attribute this to a highly regularized modeling scheme. Since it takes at least a few seconds to scan a building, we are optimistic that real-time and online reconstruction could be reached with optimized code.

8 Limitations

Our house modeling scheme has a few limitations, causing artifacts in the final model, which are shown in Figure 13. The left column shows the limitation of the block-level symmetry assumption. The main body of this house is in fact not symmetric; the back half is shorter than the front half. Since the backside of the house is completely missing from the scan, our grouping algorithm considers the shorter sidewall as missing data, resulting a symmetric house body which is incorrect. The second column shows the effect of severe occlusion. The concave area is not scanned at all, resulting in two separate houses instead of one. We can find these errors only by cross-examining the reconstructed model and the images. Therefore, exploiting the image information early on in the reconstruction process is a very promising direction. The last one shows the missing of columns. These columns are fairly flat, with vegetation in between. They are misclassified as walls, leading to mis-detection of column primitives and inaccurate reconstruction.

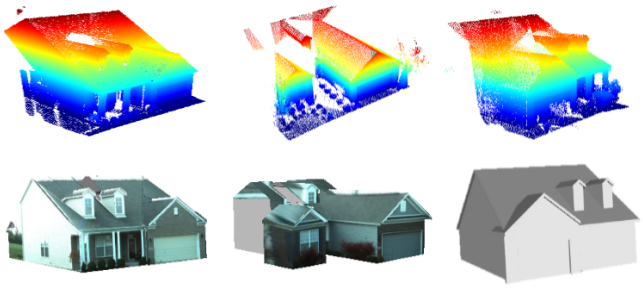

Figure 13: *Three failure cases due to erroneous symmetry assumption (left), severe occlusion (middle), and missed column extraction (right).*

Providing high-quality textures to the model turned out to be a very challenging problem. Many large scale reconstruction methods do not deal with textures at all. In the *Div-B* dataset, the registration between the color image and the 3D point cloud is not very accurate. Sometimes the error can be as big as 20 pixels. Our line-based warping algorithm cannot handle large offset without introducing distortions in the final texture. In addition, there are plenty of places that are not captured by camera. These areas now show up as gray color. These texture artifacts can be seen in the video. The problem of occlusion is even more severe in *Div-A* which has taller houses, *much denser* vegetation and flourishing trees since it is an old and more up-scale subdivision. We believe texture synthesis could provide better texturing results.

9 Conclusion

In this paper we present a complete system for residential scene modeling from 3D point cloud captured by mobile scanners. By first recognizing individual objects, we develop and apply category-specific reconstruction methods to obtain visually pleasing polygonal models in the presence of occlusions and incomplete data. A novel house modeling method is developed. The key idea is to decompose potentially complicated structures into basic blocks, including walls, roofs, columns, etc. An analysis-and-reconstruction scheme is developed to find a plausible configuration of basic blocks that best fits both the input data and typical house topology. Our system requires very little human intervention and therefore is suitable for large-scale modeling.

Looking into the future, in addition to addressing the limitations discussed, we believe that there are many exciting opportunities for further exploration. For example, we should be able to use the semantic labels inherent in our model to support editing and retargeting. In terms of modeling quality, some fine details on the house, such as hand-rails and staircases, are not reconstructed in our current approach. These details are only represented with a few sparse points. They are however visible in the images. We plan to improve our segmentation algorithm by using both depth and color so that these details can be recognized and replaced. In addition, we plan to automatically cluster points in the same category to find different objects. Overall, the combination of pattern recognition algorithms with geometry processing is expected to lead to better models that support not only high-fidelity visualization, but also editing and eventually search.

Acknowledgements

We would first like to thank the anonymous reviewers for their valuable feedback. Also, we would especially like to thank Yongwook Song for LiDAR data acquisition and processing, Qing Zhang, Aaron Camenisch, Seth Parker for video editing, Wenmeng Zhou and Yiping Yang for data labeling. This work is supported in part by US NSF grant IIS-0448185, CCF-0811647, and CNS-0923131. Yu Zhou is supported by the National Natural Science Foundation of China (61100111). Ligang Liu is supported by the National Natural Science Foundation of China (61222206) and the National Basic Research Program of China (2011CB302400).

References

AKBARZADEH, A., FRAHM, J.-M., MORDOHAI, P., CLIPP, B., ENGELS, C., GALLUP, D., MERRELL, P., PHELPS, M., SINHA, S., TALTON, B., WANG, L., YANG, Q., STEWNIUS, H., YANG, R., WELCH, G., TOWLES, H., NISTR, D., AND POLLEFEYS, M. 2006. Towards urban 3d reconstruction from video. In *3DPVT*.

BOKELOH, M., BERNER, A., WAND, M., SEIDEL, H.-P., AND SCHILLING, A. 2009. Symmetry detection using line features. *Computer Graphics Forum (Proceedings of Eurographics)*.

CHAUVE, A.-L., LABATUT, P., AND PONS, J.-P. 2010. Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data. In *CVPR*.

DEBEVEC, P. E., TAYLOR, C. J., AND MALIK, J. 1996. Modeling and rendering architecture from photographs: A hybrid geometry-and image-based approach. In *SIGGRAPH*, 11–20.

DORNINGER, P., AND PFEIFER, N. 2008. A comprehensive automated 3d approach for building extraction, reconstruction, and regularization from airborne laser scanning point clouds. *Sensor 8*, 7323–7343.

FRAHM, J.-M., GEORGEL, P., GALLUP, D., JOHNSON, T., RAGURAM, R., WU, C., JEN, Y.-H., DUNN, E., CLIPP, B., LAZEBNIK, S., AND POLLEFEYS, M. 2010. Building rome on a cloudless day. In *ECCV*.

FRUEH, C., AND ZAKHOR, A. 2003. Constructing 3d city models by merging aerial and ground views. *IEEE Computer Graphics and Applications 23*, 6, 52–61.

FRUEH, C., JAIN, S., AND ZAKHOR, A. 2005. Data processing algorithms for generating textured 3d building facade meshes from laser scans and camera images. *Int. J. Comput. Vision 61*, 159–184.

GOLOVINSKIY, A., KIM, V., AND FUNKHOUSER, T. 2009. Shape-based recognition of 3d point clouds in urban environments. In *ICCV*.

IRSCHARA, A., ZACH, C., KLOPSCHITZ, M., AND BISCHOF, H. 2012. Large-scale, dense city reconstruction from user-contributed photos. *Computer Vision and Image Understanding 116*, 1, 2–15.

KARANTZALOS, K., AND PARAGIOS, N. 2010. Large-scale building reconstruction through information fusion and 3-d priors. *IEEE Transactions on Geoscienec and Remote Sensing 48*, 5, 2283–2296.

LAFARGE, F., AND MALLET, C. 2011. Bulding large urban environment from unstructed point data. In *International Conference on Computer Vision*, 1068–1075.

LAFARGE, F., DESCOMBES, X., ZERUBIA, J., AND DESEILLIGNY, M. P. 2010. Structural approach for building reconstruction from a single dsm. ieee trans. *PAMI 32*, 1, 135–147.

LI, Y., WU, X., CHRYSANTHOU, Y., SHARF, A., COHEN-OR, D., AND MITRA, N. J. 2011. Globfit: Consistently fitting primitives by discovering global relations. *ACM Transactions on Graphics 30*, 4.

LI, Y., ZHENG, Q., SHARF, A., COHEN-OR, D., CHEN, B., AND MITRA, N. J. 2011. 2d-3d fusion for layer decomposition of urban facades. In *ICCV*.

LIU, F., GLEICHER, M., JIN, H., AND AGARWALA, A. 2009. Content-preserving warps for 3d video stabilization. *ACM Trans. Graph. 28*, 3 (July), 44:1–44:9.

MÜLLER, P., WONKA, P., HAEGLER, S., ULMER, A., AND VAN GOOL, L. 2006. Procedural modeling of buildings. *ACM Trans. Graph. 25* (July), 614–623.

MUSIALSKI, P., WONKA, P., ALIAGA, D. G., WIMMER, M., VAN GOOL, L., AND PURGATHOFER, W. 2012. A survey of urban reconstruction. In *EUROGRAPHICS 2012 State of the Art Reports*, Eurographics Association,

NAN, L., SHARF, A., ZHANG, H., COHEN-OR, D., AND CHEN, B. 2010. Smartboxes for interactive urban reconstruction. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2010) 29*, 4, Article 93.

PAULY, M., MITRA, N. J., WALLNER, J., POTTMANN, H., AND GUIBAS, L. J. 2008. Discovering structural regularity in 3d geometry. *ACM Trans. Graph. 27* (August), 43:1–43:11.

POULLIS, C., AND YOU, S. 2009. Automatic creation of massive virtual cities. In *IEEE Virtual Reality Conference*, 199–202.

POULLIS, C., AND YOU, S. 2009. Photorealistic large-scale urban city model reconstruction. *IEEE Transaction on Visualization and Computer Graphics 15*, 4, 654–669.

SCHAFFALITZKY, F., AND ZISSERMAN, A. 1999. Geometric grouping of repeated elements within images. In *Shape, Contour and Grouping in Computer Vision*, 165–181.

SCHNABEL, R., WAHL, R., AND KLEIN, R. 2007. Efficient ransac for point-cloud shape detection. *Computer Graphics Forum 26*, 2 (June), 214–226.

SHAO, T., XU, W., ZHOU, K., WANG, J., LI, D., AND GUO, B. 2012. An interactive approach to semantic modeling of indoor scenes with an rgbd camera. *ACM Trans. Graph.*.

SINHA, S., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. 2008. Interactive 3d architectural modeling from unordered photo collections. In *SIGGRAPH Asia*, 159:1–159:10.

VAN GOOL, L., ZENG, G., VAN DEN BORRE, F., AND MULLER, P. 2007. Invited paper: Towards mass-produced building models. In *PIA07*, 209.

VANEGAS, C. A., ALIAGA, D. G., WONKA, P., MUELLER, P., WADDELL, P., AND WATSON, B. 2010. Modeling the appearance and behavior of urban spaces. *Computer Graphics Forum 29*, 1, 25–42. Also in Eurographics 2009 STAR (State-of-the-Art Report).

VANEGAS, C. A., ALIAGA, D. G., AND BENES, B. 2012. Automatic extraction of manhattan-world building masses from 3d laser range scans. *IEEE Transactions on Visualization and Computer Graphics 18*, 10, 1627–1637.

WERNER, T., AND ZISSERMAN, A. 2002. New techniques for automated architectural reconstruction from photographs. In *ECCV*.

WU, C., FRAHM, J.-M., AND POLLEFEYS, M. 2010. Detecting large repetitive structures with salient boundaries. In *ECCV*.

XIAO, J., AND FURUKAWA, Y. 2012. Reconstructing the world's museums. In *ECCV*.

XIAO, J., AND QUAN, L. 2009. Multiple view semantic segmentation for street view images. In *ICCV*.

XIAO, J., FANG, T., TAN, P., OFEK, P. Z. E., AND QUAN, L. 2008. Image-based façade modeling. *ACM Trans. Graph. 27*, 5, 1–10.

XIAO, J., FANG, T., ZHAO, P., LHUILLIER, M., AND QUAN, L. 2009. Image-based street-side city modeling. *ACM Transactions on Grpahics 28*, 5, 10–19.

ZHANG, C., WANG, L., AND YANG, R. 2010. Semantic segmentation of urban scenes using dense depth maps. In *ECCV*.

ZHOU, Q.-Y., AND NEUMANN, U. 2010. 2.5d dual contouring: a robust approach to creating building models from aerial lidar point clouds. In *ECCV*.

ZHOU, Q., AND NEUMANN, U. 2011. 2.5d building modeling with topology control. In *CVPR*.

ZHOU, Q.-Y., AND NEUMANN, U. 2012. 2.5d building modeling by discovering global regularities. In *CVPR*.