

# Automatic Extraction of Manhattan-World Building Masses from 3D Laser Range Scans

Carlos A. Vanegas, *Member, IEEE Computer Society*,  
 Daniel G. Aliaga, *Member, IEEE Computer Society*,  
 and Bedrich Benes, *Member, IEEE Computer Society*

**Abstract**—We propose a novel approach for the reconstruction of urban structures from 3D point clouds with an assumption of Manhattan World (MW) building geometry; i.e., the predominance of three mutually orthogonal directions in the scene. Our approach works in two-steps. First, the input points are classified according to the MW assumption into four local shape types: walls, edges, corners, and edge-corners. The classified points are organized into a connected set of clusters from which a volume description is extracted. The MW assumption allows us to robustly identify the fundamental shape types, describe the volumes within the bounding box, and reconstruct visible and occluded parts of the sampled structure. We show results of our reconstruction that has been applied to several synthetic and real-world 3D point datasets of various densities and from multiple viewpoints. Our method automatically reconstructs 3D building models from up to 10 million points in 10 to 60 seconds.

**Index Terms**—3D Modeling, 3D Reconstruction, Laser Scans, Buildings, Manhattan World

## 1 INTRODUCTION

AUTOMATIC reconstruction of urban structures has attracted the attention of researchers from many different areas and it has found its way into numerous applications such as mapping, navigation, and computer-aided design. Common desires include automation in the reconstruction process of one or more buildings, robustness to noise in the measurements/observations, the ability to cope with missing data (e.g., due to occlusions), and the capacity to handle datasets of different sampling densities within a single scan or amongst various scans.

Numerous reconstruction methods in computer graphics specialize in specific types of objects (e.g., buildings and facades [1], [2], [3]) in order to make assumptions about the sampled geometry, and focus on customizing the reconstruction process of generic objects according to specific modeling and rendering objectives (e.g., [4], [5], [6]). State-of-the-art methods for reconstructing 3D geometry from range scans and photographs, including those focused on buildings and facades, focus on issues related to data quality and data completeness. On the one hand, methods based on photographs (e.g., structure from motion or dense stereo [7]) provide a relatively dense sampling of a building's exterior and, given a sufficiently large

number of images, a sufficiently complete sampling can be obtained as well. However, dense correspondences need to be robustly and accurately established to obtain range data. On the other hand, 3D laser scans omit the dependence on dense correspondence to obtain range measurements, but the sampling density and sampling completeness can vary significantly during scanning, especially when capturing large structures such as buildings. Scanning can be improved by laboriously using many laser scans (e.g., [8]), but in general it is impractical to obtain a large number of scans of a building's exterior.

The main idea of our work is to incorporate assumptions about building geometry to improve 3D laser-scanning reconstruction. While some methods pursue a local analysis in order to complete relatively small areas of missing samples (e.g., [9], [10]), these approaches cannot handle large missing areas and typically assume a dense sampling most everywhere else. Instead, we make assumptions about the possible geometrical configurations of the building and use them to limit the possible 3D shapes. This enables robustly extracting the underlying 3D building model even from sparse and incomplete 3D point clouds. One such family of geometrical configurations are buildings belonging to a Manhattan world (MW) [11] which contains structures with a predominance of three mutually orthogonal directions. The MW assumption has been used in several methods to obtain 3D structure from images capturing building interiors (e.g., [12]) and exteriors (e.g., [13], [14]).

We propose the first solution, to the best of our knowledge, which uses the MW assumption to enable the automatic and robust calculation of complete

• C. A. Vanegas and D. G. Aliaga are with the Department of Computer Science, Purdue University, 305 N. University Street, West Lafayette, IN 47907-2107. Email: [cvanegas, aliaga@cs.purdue.edu](mailto:cvanegas, aliaga@cs.purdue.edu)

• B. Benes is with the Department of Computer Graphics Technology, Purdue University, 401 N. Grant Street, West Lafayette, IN 47907-2021. Email: [bbenes@purdue.edu](mailto:bbenes@purdue.edu)

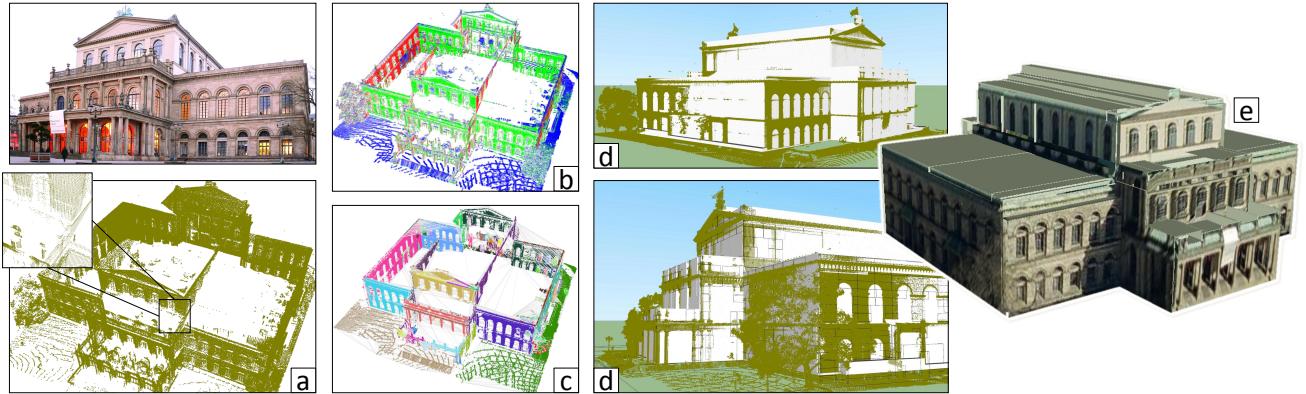


Fig. 1. System Pipeline. a) The input to our system consists of unorganized 3D point clouds. b) A local surface description type is assigned to each point. c) Typed points are organized into a connected set of clusters from which a volume description is extracted. For purposes of visual evaluation, the reconstructed volume is superimposed over the original point set, including noise and obstacles (d), and textured with photographs of the buildings (e).

building masses from unorganized 3D point clouds obtained from ground-level 3D laser scans, or LIDAR (Figure 1). Our approach uses the following three key observations:

- surface description - the local geometry represented by points sampling a MW building can be described by at most one of four fundamental types that are robustly identifiable and can be used in subsequent automatic processing ( $tWall$ ,  $tEdge$ ,  $tCorner$ ,  $tEdgeCorner$ );
- volume description - since the buildings contain only axis-aligned planes, it is possible to describe the volumes within the bounding box of the building that are inside and outside of the building mass, even for non-convex building masses; data that do not belong to the planes can be quickly discarded, and
- data description - the data described in the multiple axis-aligned planes is inherently redundant; this results in the ability to robustly reconstruct portions of a building not observed from one view direction but indirectly included in others.

Our approach consists of two main steps. The first classifies each 3D point by a surface description type. As shown in Section 3.1, in a MW building there are 47 possible local surface shapes which, after considering symmetries due to rotations and flips, can be classified into four fundamental types: walls, edges, corners, and edge-corners. To determine each point's classification, the algorithm inspects the local point neighborhood and analyses the compatibility of the classification with all neighboring points. In the second step, the building mass is reconstructed from the classified points. The classified points are organized into a connected set of clusters from which a volume description is extracted. Each point cluster describes a portion of a wall that is connected to neighboring

clusters via edges and/or corners. The connected clusters are then represented by an adaptive set of quadrilaterals obtained from recursive subdivision. A ray-casting algorithm is then used to calculate the interior and exterior volumes of the building from the quadrilaterals. The large set of small volumes that collectively represent the building mass is then coalesced and simplified to provide a polygonal description of the MW building.

Our approach has been applied to several synthetic and real-world 3D point datasets of various densities, from multiple viewpoints, and with up to 10 million points, that resulted from capturing a single building. Scans sampling more than one building need to be previously segmented. Point sample density in reconstructed areas ranged from under ten to a few hundred points per square meter. All processing is automatic and operates on a standard desktop PC. Our reconstructions are processed from unorganized 3D point clouds in 10 to 60 seconds per building running on a single core of the CPU.

## 2 PREVIOUS AND RELATED WORK

As previous work we focus on methods that use unorganized 3D point clouds, obtained from laser scans or LIDAR devices. A first group of methods focus on reconstructing buildings geometry from airborne LIDAR data. In general, this group of approaches concentrates on reconstructing roofs, producing building footprints, and/or computing 2.5D building models. The body of work in this area is large, thus we highlight some recent papers. Zhou and Neumann [15] proposed a streaming framework for reconstructing buildings from large aerial LIDAR datasets and Poullis and You [16] an automatic 2.5D reconstructions from aerial data. Matei et al. [17] describe how to obtain buildings in dense urban areas. Verma et

al. [3] focus on segmenting points belonging to the roof surface and to the ground surface, and then fit a collection of roof models chosen from an a priori defined set. While a recent work by Kelly and Wonka [18] could be used to create additional geometry for the 2.5D reconstructions generated by this type of methods, their work infers facade geometry by procedurally extruding the building footprint, and will not use sampled data to guide the modeling process. Numerous approaches have been proposed in the areas of photogrammetry and remote sensing and we refer the reader to the survey by Tarsha-Kurdi et al. [19] for additional works.

Some reconstruction methods use assumptions about the underlying geometry in order to improve robustness (e.g., [20], [21]). In building reconstruction, the assumption of planarity is one of the most common but it alone does not always result in complete and closed 3D models. The library of assumed primitive shapes can be expanded to include more complex shape priors. Lafarge et al. [22] uses a library of 3D parametric blocks for reconstructing simple building models, Nan et al. [1] uses boxes to reconstruct building models from LIDAR data and relies on user input to guide the reconstruction process.

A second group of methods focus on reconstructing the facades of buildings, often using ground-based scans. For example, Liu and Stamos [23] register 2D images to 3D ground-level scans. Zheng et al. [2] use global consolidation and manual input to complete partial scans of building facades, and Früh and Zakhor [13] generated textured facade meshes from laser scans obtained while driving on public roads. It is worth noting that many image-based methods also focus on facades and produce good results but typically require user intervention.

Several hybrid and more specialized approaches have also been proposed that combine the previous two groups. Früh and Zakhor [24] merge scan data with images to obtain urban models. Grammar based methods join procedural modeling with terrestrial and aerial range data to detect and parse 3D point clouds of buildings [3], [14], and to complete and reconstruct building models. However, the reconstruction efforts require significant manual input in the form of manual model editing or specifying the grammar [25]. Progress has been made in automating the discovery of the procedural rules that form the underlying building grammar (e.g., [26]), but to our knowledge such has not been applied to noisy unorganized 3D point clouds resulting from laser scans. Our approach builds upon the MW assumption first proposed by Coughlan and Yuille [11]. Starting with 2D images, the MW assumption has been used to reconstruct building interiors [12] and exteriors [27], [14]. Related to our approach is the work of Schindler and Bauer [28] who propose shape priors but applied only to facade-level details. Haala et al. [29] fit a piecewise

planar boundary representation to LIDAR data, but assume the 3D points are triangulated and a building floor plan is provided beforehand.

### 3 SURFACE DESCRIPTION

Our approach for building reconstruction proceeds in two main steps (Figure 2). First, input 3D points are classified by calculating the local surface shape classification. The classification eliminates unwanted points from the dataset, provides robustness to noise, and later helps to complete missing data. In particular, since within a point’s neighborhood there might be points from surfaces at different orientations, explicitly classifying points as belonging to either segments of walls, corners, or edges is useful to recover local shape. In the second step, the building volume is described by merging the classified points and filling-in the volume. The classification provided by the first step implicitly indicates the connectivity between different faces of the model.

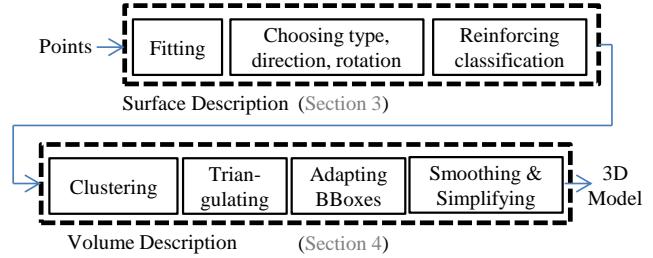


Fig. 2. System overview.

It is worth noting that our method does not assume sampled points to be exactly located at walls, corners, or edges - instead, we classify the local shape around each point and find a potentially different, but nearby, pivot point. A pivot point corresponds to the corner point or to a point on the edge axis. Thus, a building corner can lie “in between” point samples.

While existing approaches were considered for classifying the sampled points (e.g., [30], [31]), we opted to create a new method that specifically exploits the MW assumption in order to achieve classification in linear time on the number of points (i.e., constant time per point) and to support classifying points based not only on the normal directions of their containing plane, but also on the local surface shape. A related work is that by Toldo and Fusillo [30], which proposes J-Linkage to fit arbitrary planes to data points corrupted by noise and outliers, and then classifies the points based on these planes. Their classification does not detect points that belong to two or more planes, and has superlinear complexity, thus not well-suited for scaling efficiently to millions of points.

For the description of our approach, we assume without loss of generality, that the point cloud has been rotated so that the dominant triple of directions

in the MW world are aligned with the  $x$ -,  $y$ -, and  $z$ -axis.

### 3.1 Manhattan World Local Shapes

Three predominant surface directions in MW geometry give rise to three possible surface normal directions and four fundamental local shape types. The local shape surrounding a point of a MW surface manifold must belong to one of the categories denoted as  $tWall$ ,  $tEdge$ ,  $tCorner$ ,  $tEdgeCorner$  (Figure 3). Shape types  $tWall$ ,  $tEdge$  and  $tCorner$  have respectively one, two and three predominant surface directions. Shape type  $tEdgeCorner$  is used to denote a region where a plane and an edge intersect and form a shape that resembles a concave corner. Each shape type has several possible directions and rotations. The signed direction indicates the dominant orientation of the shape (i.e., along the positive or negative  $x$ ,  $y$ , or  $z$  axis) – an unsigned direction implies the same shape type can be oriented in either direction. Each shape type can also be rotated around the axis of the given direction (i.e., a rotation by  $0, \pi/2, \pi, 3\pi/2$ ). Using the aforementioned shape types,  $tWall$  has 3 unsigned directions, and 1 unique rotation per direction, for a total of 3  $tWall$  types. Shape type  $tEdge$  has 3 unsigned directions and 4 unique rotations per direction, for a total of 12  $tEdge$  types. Shape type  $tCorner$  has no orientation and 8 unique rotations. Shape type  $tEdgeCorner$  has 3 signed directions and 4 unique rotations per direction, for a total of 24  $tEdgeCorner$  types. Altogether, there are 47 possible local shapes in a MW building, but each can be described by one of four fundamental shape types. Figure 4 shows 35 of the 47 shapes.

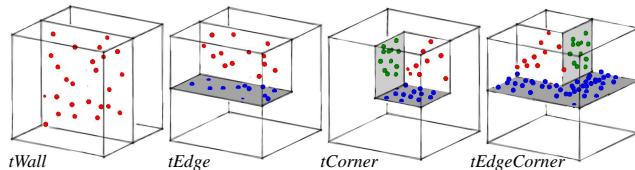


Fig. 3. Shape classification. Each of the 4 shape types are shown.

In unorganized 3D point clouds, the surface normal or local point connectivity information needed for estimating the local shape is typically not provided but can be estimated from the local neighborhood of each point. The optimal neighborhood size to estimate the fundamental shape type depends on the local density of the sample points and on the amount of the noise in the input samples. In practice, the point sample density varies and point samples have error. Thus, in order to determine a plane, the neighborhood must contain at least three points but having more points is beneficial.

### 3.2 Shape Classification

The classification of a MW point sample  $p_i = (x_i, y_i, z_i)$ , for  $i \in [1, N]$ , is achieved by defining a neighborhood ball around the sample point, determining the value of a fitting function to the MW planes, computing the pivot points of such planes, and determining the direction and rotation of the corresponding shape type. The pivot point is the center of the shape type about which the local surface produces the edges of the shape type. The exact pivot point will in most cases not coincide with any sampled point but it is expected to be nearby, and it is assumed that at least one of the points within the neighborhood ball is a reasonable approximation. Further, the initial shape types are refined using a reinforcement scheme (Section 3.3) and the direction/rotation of the shape type is later used for the volume description process (Section 4). Not all input points are classified - points with a high fitting-error or points insufficiently reinforced are ignored.

#### 3.2.1 Neighborhood Ball

For each sample point  $p_i$ , a neighborhood ball centered on  $p_i$  and of radius  $r$  is defined as

$$B(p_i, r) := \{p_k : |p_k - p_i| < r, k \in [1, N]\}$$

and contains all the sample points  $p_k$  that lay inside the ball. We denote the elements of  $B(p_i, r)$  by  $\{b_1, \dots, b_m\}$ . In our implementation, the radius  $r$  is either automatically computed to be large enough to ensure a user-specified minimum number of points or is provided by the user. For brevity, we define  $B_i = B(p_i, r)$ .

#### 3.2.2 Fitting Function

To fit to the MW planes and to compute the pivot point, we use a function that measures the deviation of every point within  $B_i$  from each of the possible MW planes. For every neighborhood  $B_i$  there is a pivot point  $b_i^*$  such that the fitting error is minimized when the planes are pivoted at  $b_i^*$ . In order to find a good pivot point, we use every point  $b_i \in B_i$  and evaluate the fitting error using

$$\varepsilon_{ij} = \sum_{b_j \in B_i} (\min(|b_{ix} - b_{jx}|, |b_{iy} - b_{jy}|, |b_{iz} - b_{jz}|))$$

where  $b_j \in B_i$  refers to all points  $b_j$  inside ball  $B_i$  and  $i \neq j$ . We choose the  $b_j$  that minimizes  $\varepsilon_{ij}$  and denote it by  $b_i^*$ . The associated fitting function is called  $\varepsilon_i^*$ . Because of the noise in the data, there is in general no more than one point that minimizes  $\varepsilon_{ij}$ .

#### 3.2.3 Shape Types and Directions

For each point  $p_i$ , we inspect the number of times the value of  $\varepsilon_{ij}^*$  was obtained from each of the  $x$ ,  $y$ , or  $z$

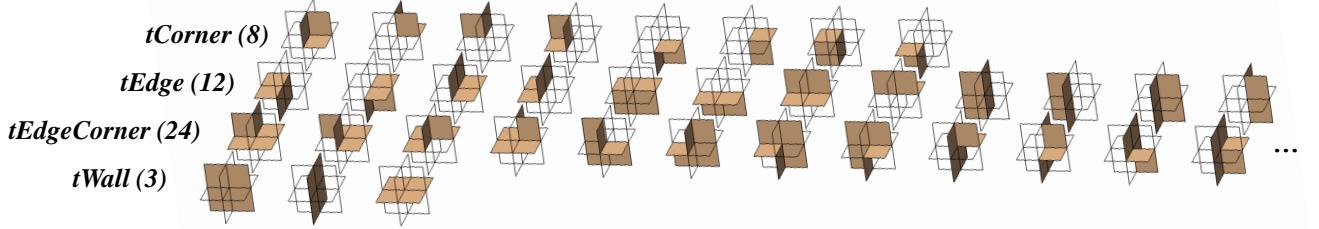


Fig. 4. Local shapes in a MW building. All of the shapes for types *tWall*, *tEdge* and *tCorner* are shown, together with 12 of the 24 shapes for type *tEdgeCorner*.

components and choose the best fitting shape type. In particular, we define

$$C_{i_{\text{dim}}} = \sum_{b_j \in B_i} (\arg \min \varepsilon_{ij}^* == \text{dim})$$

where *dim* denotes the axis *x*, *y*, and *z*. The counters  $C_{i_x}$ ,  $C_{i_y}$ , and  $C_{i_z}$  are used to determine the direction of the shape type of  $p_i$  using the following reasoning. The dominance (or lack of it) of a direction is an indicator of the point's shape type. For instance, if  $C_{i_x}/m_i \approx 1$ , then for most points in  $B_i$  the component along which the distance was minimum is *x*. This implies the points in  $B_i$  likely lie on a single plane and the normal vector of the plane is the *x* axis. The shape type then becomes *tWall* on the *YZ* plane. Analogously  $(C_{i_x} + C_{i_y})/m_i \approx 1$  indicates most points in  $B_i$  are likely to lie on two planes forming an L-shaped edge along *z* axis; the shape type in this case is a *tEdge* spanning the *XZ* and *YZ* planes. If all counts have approximately the same values, then there are no dominant directions and the points in  $B_i$  are likely to lie on three planes forming a corner.

The shape type classification computes for each point the aforementioned ratios for all possible shape types. The shape type with the largest ratio, yet above a minimum threshold (set experimentally to 0.8), is chosen. Outliers will be detected, and removed, during the later reinforcement step.

### 3.2.4 Shape Rotations

While points of type *tWall* are invariant to rotation, the rotations of points of type *tEdge*, *tCorner*, and *tEdgeCorner* do need to be computed. We define signed distance values

$$S_{i_{\text{dim}}} = \sum_{b_j \in B_i} (b_{j_{\text{dim}}} - b_{\text{dim}}^*)$$

where each term evaluates for an axis  $\text{dim} = x, y, z$ , as the sum of the signed differences between all the points in  $B_i$  and the pivot  $b^*$ . The signs of  $S_{i_x}$ ,  $S_{i_y}$ , and  $S_{i_z}$  are used to determine the rotation of the shape type. For instance, if the shape type has been determined to be a *tEdge* along the *z* axis, then there are four possible rotations about *z*. Each rotation will place the shape in a different quadrant of the *XY*

plane. If  $S_{i_x} > 0$  and  $S_{i_y} > 0$ , then the shape is in the positive *X* and positive *Y* quadrant. If  $S_{i_x} < 0$  and  $S_{i_y} > 0$ , then the shape is in the negative *X* and positive *Y* quadrant, and so forth.

### 3.3 Reinforcement

Given the initial shape types, the classified points within each neighborhood ball are used to reinforce *tEdge*/*tEdgeCorner* and *tCorner* classifications and to further improve the accuracy of each pivot location. For instance, a sample point  $p_i$  classified as *tEdge* spanning the *XZ* and *YZ* planes should have a large fraction of its neighbors (e.g.,  $> 0.75$ ) be *tWall*'s on the *XZ* and *YZ* planes. If this is the case, the position of the pivot point  $b_i^*$  will be modified by making the *x* coordinate equal to the mean *x* coordinates of its *YZ* plane *tWall* neighbors, and the *y* coordinate equal to the mean *y* coordinates of the *XZ* plane *tWall* neighbors. If this is not the case, the point is considered unclassified. To prevent circular dependencies, the reinforcement process is applied sequentially: points classified as *tWall* reinforce *tEdge*'s/*tEdgeCorner*'s, then *tEdge*'s/*tEdgeCorner*'s reinforce *tCorner*'s. The result is a potentially smaller, but more accurate, set of classified point samples and their corresponding pivots.

## 4 VOLUME DESCRIPTION

In the second step, our method uses the previously classified points to construct volumes that approximate the geometry sampled by the points. This second step is performed in several sub-steps. First the classified points are joined to form nearly co-planar clusters, which, in turn, are connected to their adjacent clusters. Then, a set of thin boxes is adaptively subdivided to approximate each cluster. A ray-casting algorithm is then used to connect and fill the interior volumes and the boxes are used to accelerate this process. Finally, the volume is filled-in and simplified to yield the final 3D polygonal model.

### 4.1 Classified Point Clustering

The classified-point clustering algorithm produces a set of triangulated and interconnected clusters and

exploits our MW assumption to compute a set of axis-aligned bounding boxes for approximating the cluster. The input to clustering is the set of classified points  $p_i$  whose fitting error  $\varepsilon_i^*$  is less than a given threshold.

#### 4.1.1 Cluster Creation and Triangulation

Two points  $p_i$  and  $p_k$  are in the same cluster if and only if there is a valid path of classified points between them. To succinctly describe what constitutes a valid path let's restrict ourselves to a cluster of points in the  $YZ$  plane. We use the notation  $tWallX$  to imply a  $tWall$  whose normal is along the  $x$ -axis. Similarly,  $tEdgeXY$ ,  $tEdgeXZ$ , and  $tCornerXYZ$  implies the normals of the edge or corner are along the  $x$ -axis,  $y$ -axis and/or  $z$ -axis as implied by the labeling. A valid path is composed of an ordered sequence of points  $\{q_1, q_2, \dots, q_m\}$  that satisfy:

- (i)  $q_1 = p_i$  and  $q_m = p_k$ ,
- (ii)  $\forall i : T(q_i) = (tWallX|tEdgeXY|tEdgeXZ|tCornerXYZ)$ ,
- (iii)  $\forall i : |q_i - q_{i+1}| < r$ , and
- (iv)  $|q_{i_x} - q_{(i+1)_x}| < \epsilon$ .

The criterion (i) requires  $p_i$  and  $p_k$  to be the starting point and ending point of the path. The criterion (ii) specifies that all points in the path must be of the same type/direction or of a compatible type sharing a common plane and the operator  $T(\cdot)$  returns the shape type of a given point. The criterion (iii) ensures adjacent path points are closer than a threshold distance  $r$ . The last criterion (iv) specifies that any two consecutive path points must be approximately coplanar (i.e., on the  $YZ$  plane in this case). For example, a valid path can exist between two  $tWalls$  on (nearly) the same plane, or a valid path can exist between a  $tEdge$  sharing a plane with a  $tWall$ .

The aforementioned clustering criteria produce a partitioning of the  $tWall$ 's (i.e., a  $tWall$  is assigned exactly to one cluster) and produces shared points only for  $tEdges$ ,  $tCorners$ , and  $tEdgeCorners$  spanning the expected planes implied by their directions and rotations.

The points in the same cluster are then triangulated to obtain a mesh-based representation of the cluster. Large triangles are omitted. In practice, the resulting triangulation produces a fairly accurate representation of convex and non-convex building faces (e.g., Figure 10b).

#### 4.1.2 Adaptive Bounding Boxes

For each triangulated cluster, we calculate a recursive set of axis-aligned bounding boxes (AABBs). The AABBs will speed-up the process of ray casting described in the next section. Because of the near-coplanarity criterion during cluster creation, the depth of an AABB (i.e., the length of the side perpendicular to the dominant plane) is usually small. The initial AABB is the bounding box of the entire cluster whose

width and height are recursively split until reaching a minimum box size (i.e., the box is not split along the depth dimension) (Figure 5). The minimum box size is set by default to one meter and can be modified by the user to control the detail of the reconstruction.

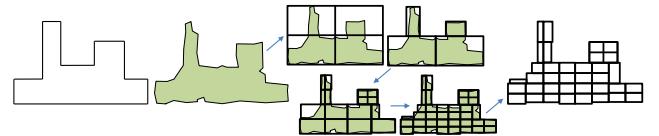


Fig. 5. Adaptive bounding boxes for cluster representation.

## 4.2 Volume Rays

Volume rays are used to compute the 3D region enclosed by the meshes created during the clustering step. For a selected MW axis, the AABBs can be used to create two structured grids representing both facades of the building along the selected axis. One possible algorithm to recover the interior volume from these grids consists in sorting the faces (i.e., planar components of the structured grid at different coordinate values along the axis) in ascending order along the axis and sweeping them in sorted order. Given an initial face, it is swept along the axis until it is obstructed by another face. Then, the obstructing face starts sweeping and the unobstructed portion of the previous face continues sweeping. The sweeping (extrusion) of these regions results in a set of adjacent and non-intersecting volumes.

An alternative simple and efficient method is to densely sample the structured grid (faces) with rays. Provided there are no missing or incomplete faces in the dataset, each ray will intersect an even number of faces. From the sorted list of faces intersected by a ray, the portions of the ray that lie inside the geometry can be easily determined. The reconstructed volume would be defined by the union of all the volume ray portions that were determined to lie inside the building (Figure 6).

In practice, reconstructed faces can be missing or incomplete, and the number of faces intersected by a ray can be odd indicating an incomplete geometry. Hence, in our method, we make a conservative decision to ignore the volume regions returned by rays with an odd number of intersections (i.e., we only keep rays with even number).

A benefit of our approach is that by combining the results of using volume rays along multiple axes, we obtain redundancy which can be exploited to account for the aforementioned missing data samples. Therefore, a volume that is not described by one of the grids of rays is often successfully captured by the others.

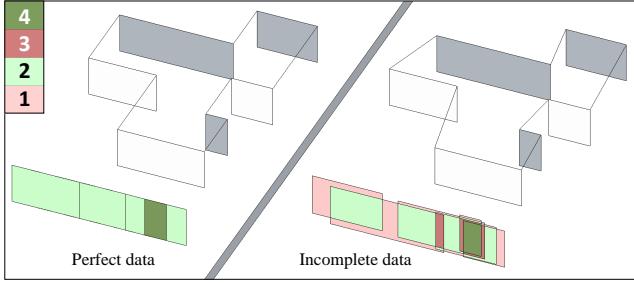


Fig. 6. Color coded number of intersections for each volume ray. An even count indicates the volume ray contains building interior.

### 4.3 Smoothing and Simplification

Several post-processing steps are followed to obtain a smoother, more accurate and more compact representation of the building volume including removing rays of low confidence, adding support rays, snapping rays to other rays of high endpoint frequency, and simplifying rays.

#### 4.3.1 Estimating Ray Confidence

Similar to the reinforcement process for classifying points using contextual information (i.e., the shape types of neighboring points), the confidence of a ray is evaluated based on the coherence with its neighboring rays. Ray confidence is useful to remove thin groups of one or more rays that may result from regions where the intersection count was falsely even. To evaluate the confidence of a ray  $r$ , we first compute the overlap between  $r$  and each of its neighbors. The neighborhood of a ray is a  $d \times d$  sub-grid of the structured ray grid, centered in  $r$  ( $d$  was experimentally set to 5). Then, for each neighbor  $s$  of  $r$ , we compute the ratio between the length of the projection of  $s$  onto  $r$ , and the length of  $r$ . The confidence of  $r$  is a normalized sum of these ratios, weighted by the grid-distance from  $s$  to  $r$ . The length of a ray is the sum of the lengths of the portions of the ray that lie inside the geometry. Intuitively, the confidence of a ray is raised by the presence of other rays with similar extents in the immediate neighborhood.

#### 4.3.2 Adding Support Rays

An optional criterion that is useful for buildings whose XY cross-sections are monotonically decreasing upwards (i.e., the building cross-section of floor  $f+1$  is always contained in the building cross-section of floor  $f$ ) is to add support rays in all the vertical positions under the ray, all the way to the base of the building. Often in street-level LIDAR datasets only the uppermost portion of a building is not occluded by other faces. Downwards completion makes a basic inference of the presence of rays that are not observed in the data but that should nevertheless be in the building given MW assumptions.

#### 4.3.3 Snapping and Simplification

We perform additional snapping and simplification operations. Small deviations in the ray endpoints can be compensated for by snapping such endpoints to nearby planes where a significantly higher number of ray endpoints already lie. This step uses information of the grid in one of the directions (e.g., grid on  $Y$ ) to modify the rays in the other direction (e.g., grid on  $X$ ).

Further, to avoid the number of recovered rays from growing too quickly, simplifications are made after any of the above post-processing operations, including closing small gaps in consecutive rays, and merging overlapping rays on a same ray grid position. Interior rays (i.e., rays that are surrounded by other rays in all neighbors) are also removed after post-processing.

The end result is a sparse collection of rays that collectively describe the building volume. The volume occupied by each ray can be coalesced and exported for subsequent geometrical modeling tasks.

## 5 IMPLEMENTATION DETAILS

Since the datasets we use contain up to 10 million points, we use a hash-based spatial partitioning data structure to efficiently store and access the 3D points. For instance, our breadth-first clustering algorithm (Section 4.1) is reduced to nearly  $O(N)$  complexity on the number of classified points. This is achieved by performing comparisons only with points in nearby neighbors that are extracted in constant time from the hash structure, instead of running costly comparisons between all pairs of points. Our hash function uses the  $x$ ,  $y$ , and  $z$  coordinates of each point to compute the index of the partition element in which the point belongs.

To efficiently access the rays, we use an additional ray grid data structure which allows for constant-time access to any ray when given its starting position. Since neighbors of rays are very frequently queried, providing such a constant lookup time is highly beneficial as it avoids a potential bottleneck of the system.

## 6 RESULTS AND DISCUSSION

We have applied our method for reconstruction of several real-world buildings using publicly available 3D point datasets [32]. Results are shown in Figures 1, 12, 10 and 11. All reconstructions are fully automated and take from 10 to 60 seconds. For each building, the system automatically computes a neighborhood ball radius to ensure that at least half of the balls have 20 or more points inside them.

## 6.1 Classification and Clustering

Computing the type of each point in the dataset and partitioning them into clusters are the key components of the pipeline. Figure 7 shows how the reinforcement process (Section 3.3) further improves the quality of the classification. Figure 8 focuses on some points of type  $tWall$  (a) and  $tEdge$  (b). The colors indicate the direction of each type. Points of type  $tWallX$  are shown in red, and points of type  $tWallY$  are shown in green. Points of type  $tEdge$  are shown as an extruded L rendered in two colors, corresponding to the estimated directions of the faces that meet at them. Figure 8c shows a close-up of the two clusters of points of type  $tWallX$  and  $tWallY$  that connect at several points of type  $tEdgeXY$ . Notice how the use of typed points allows for a correct clustering of the points in regions where the classification uncertainty is high (gap between green wall and red wall in part (a)).

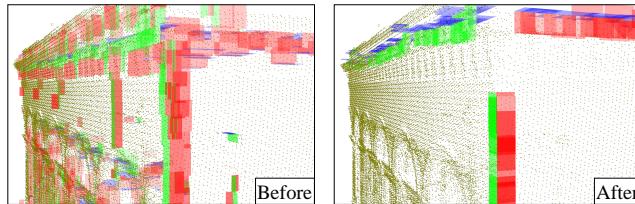


Fig. 7. Type reinforcement based on neighboring classified points.

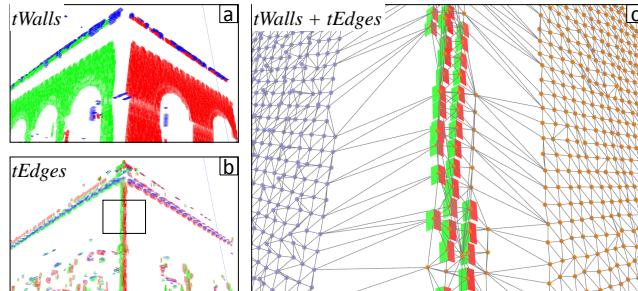


Fig. 8. Clusters of points (c) connecting  $tWalls$  (a) and  $tEdges$  (b).

## 6.2 Robustness to Point Sample Noise and Density

We have evaluated the sensitivity of our approach to varying levels of noise and density in the point sample. For this purpose, the fitting errors between a ground-truth geometry and its reconstructions are computed and visualized (Figure 9). The ground-truth geometry is a synthetic model of an artificial building that was devised to exhibit the three  $tWall$  types and at least five different types of each one of the other shape categories ( $tEdge$ ,  $tCorner$ ,  $tEdgeCorner$ ). We use a synthetic model instead of a particular real-world

building for the evaluation because (i) a model of an existing building will necessarily have non-zero error, and (ii) the mass model of any real-world MW building can be constructed by assembling one or more of these shape types.

Figure 9 shows in the top row the synthetic sample model, one of the point samples that were computed on it, and the reconstructed 3D model for that particular sample. The matrix at the bottom of the figure shows the computed reconstructions for varying levels of point density (horizontal axis) and noise (vertical axis). The local error of the reconstruction is visualized on the surface of the model using the color coding defined by the color scale on the right side of the figure. The error of a reconstructed face is computed by measuring its distance to the closest parallel face in the ground-truth model. As it is expected, the reconstruction error is minimal for a point sample with high density and low noise (bottom right model), and becomes larger as the density decreases or as the noise increases. In general, the amount of noise in the input data most directly affects the reconstruction error and also the amount of reconstruction error is usually on par with the noise level.

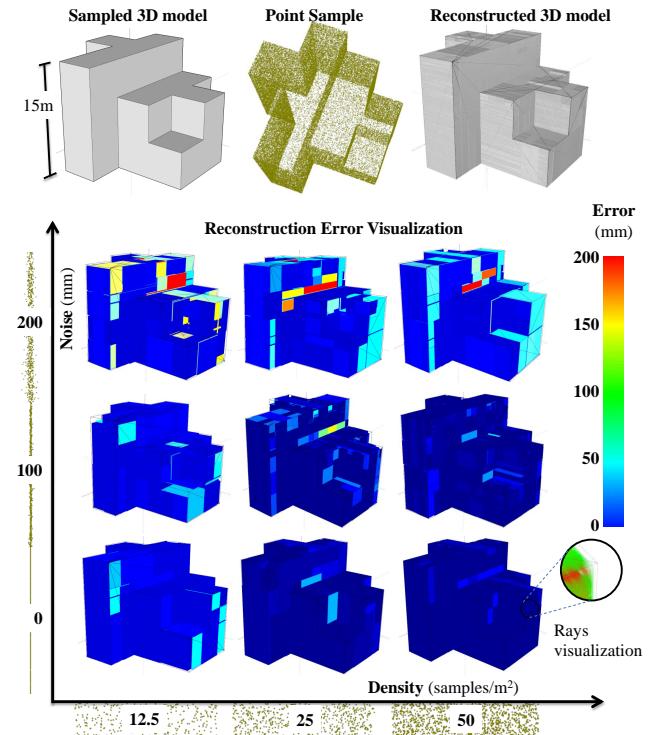


Fig. 9. Reconstruction error for varying point sample density and noise. Starting from point samples generated on the surface of a synthetic model (top), several reconstructions are computed using our method (bottom). The error of several reconstructions generated from point sets of varying levels of density and noise is computed and visualized using a color coding (right).

### 6.3 Reconstructed Buildings

Figure 10 shows the reconstruction of the mass of a castle-like building with two towers. Figure 10a shows the raw input data that is used for reconstruction, which includes points sampled on nearby trees and other objects that are not part of the building structure. Figure 10b shows the triangulations of the clusters of typed points that are further used to estimate the volume (Figures 10d-e). To highlight the importance of counting the number of cluster boxes intersected by volume rays, Figure 10c shows the result of computing volume rays without counting. This criterion is essential to accurately reconstructing the mass of buildings with non-convex volumes.

Figure 11 pictorially shows the pipeline of our method. The images in the left column show the input points (top) and their classification into *tWall* types (middle) and *tEdge* types (bottom) for surface extraction. The middle column shows the clustering of points (top), the triangulation of each cluster (middle) and the subdivision bounding boxes used to represent each triangulation. Notice how non-convex facades (e.g., the front ones) are accurately represented by these boxes. The right column shows the raw volume rays before post-processing (top), the extracted volume after ray cleaning (middle), and a manually texture-mapped model (bottom). The good fit of the reconstructed geometry to the 3D points is evidenced by the overlap of the points with the volume (middle) and by the good match of the texture (taken from a front photograph of the facade) to the building mass.

Figures 1 and 12 show two more reconstructed buildings, together with the 3D points, and for Figure 1, the typed points and cluster triangulations. All contained points sampled on trees, benches, lamps, and/or nearby parked vehicles, that were automatically discarded by our approach, omitting any data clean-up preprocessing.

### 6.4 Non-Manhattan-World Regions

Our results show that non-MW regions of the buildings are typically not reconstructed. This is an expected consequence of the fact that points sampled on non-MW regions (e.g., hip and slanted roofs) are not classified and thereafter ignored in the point clustering, triangulation and adaptive boxes fitting steps. Ignoring these points is a design choice that limits the reconstruction to MW geometry and that avoids that sample points originating in non-MW regions interfere with the correct reconstruction of MW regions. This choice enables our method to reconstruct the regions in a building that are MW-compliant without imposing the harder requirement that the entire geometry be MW-compliant. In some cases, non-MW regions are approximated by the volume rays reconstructed from connecting meshes that are MW-compliant. Gable roofs, in which only two of the

four sides slope downwards to the walls (e.g., the roof in Figure 1), are an example of these cases. In other cases, the absence of MW-compliant meshes at a given height results in no rays approximating the non-MW regions. Hip roofs, where all four sides slope downwards to the walls (e.g., the roof in Figure 11), constitute an example of these cases.

### 6.5 Visual Comparison

Figure 13 contains a table that succinctly compares our approach to several related previous works. The table lists several representative methods for building modeling (top box) and facade modeling (bottom box), together with images of their input data and resulting reconstructions. While some of the other works produce less restrictive building geometry including facades of buildings with arbitrary polygonal (non-MW) footprints and piecewise planar roofs, the main difference of our approach is the ability to create complete buildings using only ground-level scans (i.e., no roof sample points provided). Nonetheless, the geometry produced by our approach is MW compliant (i.e., MW footprint and flat roofs). Automatically reconstructing the complete geometry of a building with arbitrary facade and roof orientations is clearly a more difficult problem that is yet to be solved robustly.

The methods in [22], [16] and [33] attempt to fit polygons to the sample points and then compute the vertical extrusion of each polygon up to an estimated height. A 2.5D reconstruction is obtained as a result. While these approaches can automatically reconstruct entire cities, they completely rely on the assumption that the points were sampled on the roofs of the buildings, in order to produce complete models. This effectively implies the use of airborne acquisition for reconstructing any building. While the method by Toshev *et al.* [34] does not explicitly require aerial range data, its focus is on extracting individual planar surfaces from the points and structuring them into parse trees that represent a semantic decomposition of the building. Furthermore, their paper does not present any results showing complete building reconstructions or reconstructions where only facade points were used. In contrast, our method can automatically generate reconstructions of complete buildings even when only data from ground-level scans is available (e.g., points sampled on facades as in Figure 1a), and can provide a coarse approximation of the roof geometry based on facade data (e.g., Figure 1e).

The methods for facade reconstruction ([13], [2]) focus on creating very detailed models of facades from ground-level range scans. While these methods exploit symmetry, repetition, regularity, and interactive user guidance to compensate for unsampled regions of the facade, the resulting reconstruction is often incomplete and only represents the facade of the building that was sampled. In contrast, our method

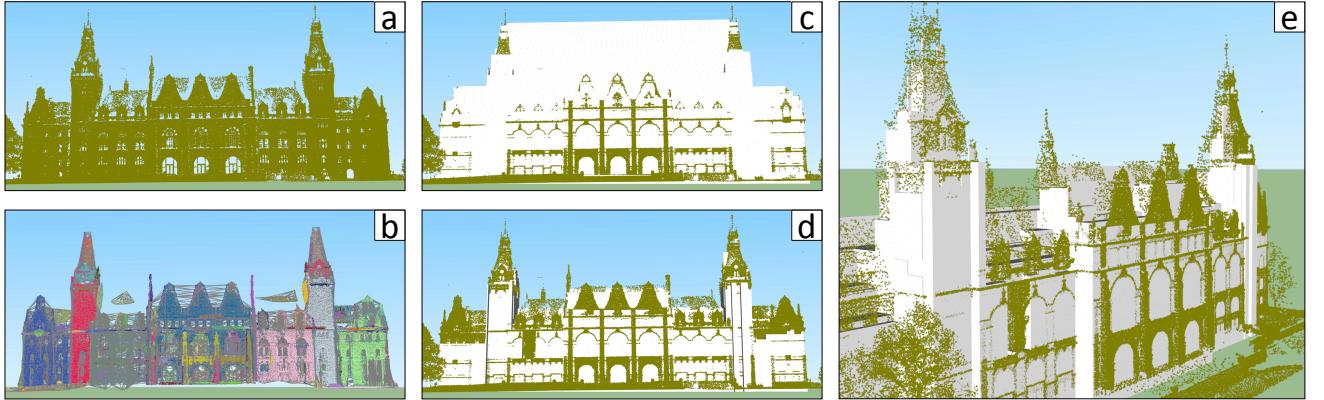


Fig. 10. Reconstructed castle-like building with non-convex volumes. a) Input 3D points. b) Clusters of classified points. c) Naïve volume rays reconstruction. d) Our complete volume ray reconstruction using intersection count criteria. e) Side view of reconstructed building showing the adapted volume rays.

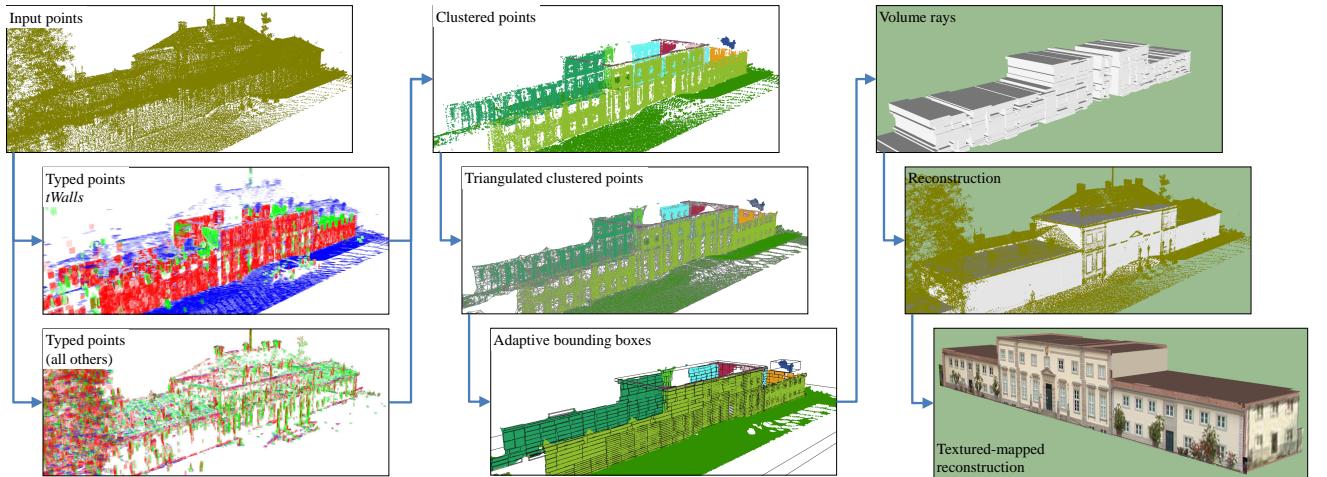


Fig. 11. System Pipeline showing: points classification (left), clustering and triangulation (middle), and volume ray reconstruction (right). The left side facade of the building is partly occluded by a tree and no cluster of triangles and adaptive boxes are created in this region. The volume rays generated by the triangulations on the front and back facades allow for a complete reconstruction in spite of the missing data.

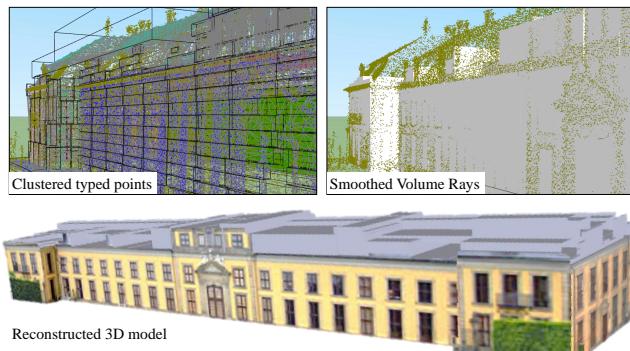


Fig. 12. Example of reconstructed building.

processes all facades simultaneously and combines their information in order to make inferences about the geometry of the unsampled parts of the building, including occluded and hidden regions of the facades,

and the building roof (e.g., Figure 10e).

Nevertheless, our method does not focus on creating detailed facade models from ground-level data, but rather on creating complete building models from incomplete ground-level samples, with no user intervention. It also does not make any assumption about the density of the point sample and it can produce reconstructions of coarsely sampled buildings (e.g., Figure 11e). However, if a dense sampling of a facade is available, the facade reconstruction methods can be used in conjunction with our approach during a postprocessing step.

## 7 CONCLUSIONS AND FUTURE WORK

We have described a novel approach for the reconstruction of urban structures from 3D laser range scans exploiting an assumption of MW building geometry. First, the input points are robustly classified

	INPUT	OUTPUT	Reconstructs if only facade points are given	Creates complete building model	Approximates roofs if not sampled	Type of Produced Geometric Models
Our system			YES	YES	YES	MW geometry + Flat roofs
Toshev <i>et al.</i> , 2010			YES	NO	NO	Polygonal footprints + Piecewise-planar roofs
Pouli and You., 2009			NO	YES	n/a	Polygonal footprints + Flat roofs
Lafarge <i>et al.</i> , 2010			NO	YES	n/a	Piecewise-planar and curved geometry
Zhou <i>et al.</i> , 2008			NO	YES	n/a	Polygonal footprints + Piecewise-planar roofs
FACADE MODELING			YES	NO	NO	Piecewise-planar geometry
			YES	NO	NO	Piecewise-planar geometry

Fig. 13. Comparison to other methods for facade and building modeling from unorganized 3D points. The results in rows 2 to 7 of this figure were originally published in [34], [16], [22], [33], [13] and [2], and are used in this paper with permission from their authors.

according to the MW assumption into one of four fundamental local shape types. In the second step, the classified points are organized into a connected set of clusters from which a volume description is extracted. The MW assumption allows us to describe the volumes within the bounding box, and to reconstruct occluded (or not captured) parts of the input data. Our method is automatic and operates at interactive computing speeds on a standard desktop PC for real-world 3D point datasets of millions of points.

Our method has several limitations. The first one comes from the MW assumption. Our algorithm reconstructs efficiently the MW parts of building exteriors, but many buildings are not pure MW buildings and have parts that are not axis-aligned. Second, our algorithm depends on the classification of the local points. Even though we have attempted for a robust solution, a large amount of noise or data incompleteness can yield to undesirable results (see Figure 9). Third, our method is not suitable for data sets with highly varying local geometry such as complex baroque facades.

There are many possible avenues for the future work. An automated roof reconstruction could be

achieved by expanding the shape types to allow for some non-MW geometries. By considering the local surface curvature, the classified points could be theoretically generalized into a class that would allow for  $C^0$  or  $C^1$  connections. By extending the range of the connecting angles and of the local context for each point, a general class of connections is possible allowing for cylindrical or pyramidal building structures as well, for example.

## ACKNOWLEDGMENTS

This work was supported by NSF IIS Grant No. 0964302, NSF CNS Grant No. 0913875, and a Google Research Gift. We are grateful to Pascal Müller for his initial motivation for this work, and to Christoph Hoffmann and Elisha Sacks for their help with our approach. We would like to thank Florent Lafarge, Ulrich Neuman, Charalambos Poullis, Andrei Sharf, Alexander Toshev and Avideh Zakhor for granting us permission to include results from their papers in Figure 13.

## REFERENCES

- [1] L. Nan, A. Sharf, H. Zhang, D. Cohen-Or, and B. Chen, "Smartboxes for interactive urban reconstruction," *ACM Trans. Graph.*, vol. 29, no. 4, 2010.
- [2] Q. Zheng, A. Sharf, G. Wan, Y. Li, N. J. Mitra, D. Cohen-Or, and B. Chen, "Non-local scan consolidation for 3d urban scenes," *ACM Trans. Graph.*, vol. 29, no. 4, 2010.
- [3] V. Verma, R. Kumar, and S. C. Hsu, "3d building detection and modeling from aerial lidar data," in *CVPR* (2), 2006, pp. 2213–2220.
- [4] Y. Yu, A. Ferencz, and J. Malik, "Extracting objects from range and radiance images," *IEEE Trans. Vis. Comput. Graph.*, vol. 7, no. 4, pp. 351–364, 2001.
- [5] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva, "Computing and rendering point set surfaces," *IEEE Trans. Vis. Comput. Graph.*, vol. 9, no. 1, pp. 3–15, 2003.
- [6] H. Huang, D. Li, H. Zhang, U. M. Ascher, and D. Cohen-Or, "Consolidation of unorganized point clouds for surface reconstruction," *ACM Trans. Graph.*, vol. 28, no. 5, 2009.
- [7] M. Pollefeys, D. Nistér, J.-M. Frahm, A. Akbarzadeh, P. Mordohai, B. Clipp, C. Engels, D. Gallup, S. J. Kim, P. Merrill, C. Salmi, S. N. Sinha, B. Talton, L. Wang, Q. Yang, H. Stewénius, R. Yang, G. Welch, and H. Towles, "Detailed real-time urban 3d reconstruction from video," *International Journal of Computer Vision*, vol. 78, no. 2-3, pp. 143–167, 2008.
- [8] M. Levoy, K. Pulli, B. Curless, S. Rusinkiewicz, D. Koller, L. Pereira, M. Ginzton, S. E. Anderson, J. Davis, J. Ginsberg, J. Shade, and D. Fulk, "The digital michelangelo project: 3d scanning of large statues," in *SIGGRAPH*, 2000, pp. 131–144.
- [9] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas, "Example-based 3d scan completion," in *Symposium on Geometry Processing*, 2005, pp. 23–32.
- [10] A. Sharf, M. Alexa, and D. Cohen-Or, "Context-based surface completion," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 878–887, 2004.
- [11] J. M. Coughlan and A. L. Yuille, "Manhattan world: Compass direction from a single image by bayesian inference," in *ICCV*, 1999, pp. 941–947.
- [12] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Reconstructing building interiors from images," in *ICCV*, 2009, pp. 80–87.
- [13] C. Früh and A. Zakhor, "Data processing algorithms for generating textured 3d building façade meshes from laser scans and camera images," in *3DPVT*, 2002, pp. 834–849.

- [14] C. A. Vanegas, D. G. Aliaga, and B. Benes, "Building reconstruction using manhattan-world grammars," in *CVPR*, 2010, pp. 358–365.
- [15] Q.-Y. Zhou and U. Neumann, "A streaming framework for seamless building reconstruction from large-scale aerial lidar data," in *CVPR*, 2009, pp. 2759–2766.
- [16] C. Poullis and S. You, "Automatic reconstruction of cities from remote sensor data," in *CVPR*, 2009, pp. 2775–2782.
- [17] B. C. Matei, H. S. Sawhney, S. Samarasekera, J. Kim, and R. Kumar, "Building segmentation for densely built urban regions using aerial lidar data," in *CVPR*, 2008.
- [18] T. Kelly and P. Wonka, "Interactive architectural modeling with procedural extrusions," *ACM Trans. Graph.*, vol. 30, no. 2, p. 14, 2011.
- [19] F. Tarsha Kurdi, T. Landes, P. Grussenmeyer, and M. Koehl, "Model-driven and data-driven approaches using lidar data: Analysis and comparison," in *PIA07*, 2007, p. 87.
- [20] S. Shalom, A. Shamir, H. Zhang, and D. Cohen-Or, "Cone carving for surface reconstruction," *ACM Trans. Graph.*, vol. 29, no. 6, p. 150, 2010.
- [21] R. Gal, A. Shamir, T. Hassner, M. Pauly, and D. Cohen-Or, "Surface reconstruction using local shape priors," in *Symposium on Geometry Processing*, 2007, pp. 253–262.
- [22] F. Lafarge, X. Descombes, J. Zerubia, and M. P. Deseilligny, "Building reconstruction from a single dem," in *CVPR*, 2008.
- [23] L. Liu and I. Stamos, "A systematic approach for 2d-image to 3d-range registration in urban environments," in *ICCV*, 2007, pp. 1–8.
- [24] C. Früh and A. Zakhor, "3d model generation for cities using aerial photographs and ground level laser scans," in *CVPR* (2), 2001, pp. 31–38.
- [25] B. Hohmann, U. Krispel, S. Havemann, and D. Fellner, "Cityfit high-quality urban reconstruction by fitting shape grammars to image and derived textured point clouds," in *In: Proceedings of the International Workshop 3D-ARCH 2009*, 2009.
- [26] M. Bokeloh, M. Wand, and H.-P. Seidel, "A connection between partial symmetry and inverse procedural modeling," *ACM Trans. Graph.*, vol. 29, no. 4, 2010.
- [27] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski, "Manhattan-world stereo," in *CVPR*, 2009, pp. 1422–1429.
- [28] K. Schindler and J. Bauer, "A model-based method for building reconstruction," in *HLK*, 2003, pp. 74–82.
- [29] N. Haala, S. Becker, and M. Kada, "Cell decomposition for the generation of building models at multiple scales," 2006.
- [30] R. Toldo and A. Fusiello, "Robust multiple structures estimation with j-linkage," in *ECCV* (1), 2008, pp. 537–547.
- [31] M. Zuliani, C. S. Kenney, and B. S. Manjunath, "The multiransac algorithm and its application to detect planar homographies," in *ICIP* (3), 2005, pp. 153–156.
- [32] I. für Kartographie und Geoinformatik. (2010, October) Terrestrische laserscans. [Online]. Available: <http://www.ikg.uni-hannover.de/index.php?id=413>
- [33] Q.-Y. Zhou and U. Neumann, "Fast and extensible building modeling from airborne lidar data," in *GIS*, 2008, p. 7.
- [34] A. Toshev, P. Mordohai, and B. Taskar, "Detecting and parsing architecture at city scale from range data," in *CVPR*, 2010, pp. 398–405.



**Carlos A. Vanegas** is a fourth-year PhD student in the Department of Computer Science at Purdue University and a research assistant at Purdue's Computer Graphics and Visualization Lab. His research is focused on concurrent behavioral and geometric methods for fast design and editing of 3D urban models. He has published 12 peer-reviewed papers, 7 of them in urban modeling. Carlos obtained his B.S. degree in Applied Mathematics from EAFIT University (Colombia) in

2007, and was an academic guest at the Faculty of Architecture at ETH Zürich in 2010.



**Daniel G. Aliaga** is a Computer Science faculty member at Purdue University since 2003. He obtained his Ph.D. and M.S. degree from UNC Chapel Hill and his B.S. degree from Brown University. Dr. Aliaga's research is primarily in the area of 3D computer graphics but overlaps with computer vision. His research area is of central importance to many critically important industries, including computer-aided design and manufacturing, telepresence, scientific simulations, and education. To date Prof. Aliaga has over 60 peer reviewed publications and has chaired and served on numerous ACM and IEEE conference and workshop committees.



**Bedrich Benes** is an associate professor in the Computer Graphics Technology department at Purdue University. He obtained his Ph.D. and M.S. degree from the Czech Technical University. His research is primarily in the area of procedural modeling, real-time rendering, and 3D computer graphics in general. To date he has published over 50 peer reviewed publications.