Building Segmentation for Densely Built Urban Regions Using Aerial LIDAR Data

Bogdan C. Matei Harpreet S. Sawhney Supun Samarasekera Janet Kim Rakesh Kumar
Sarnoff Corporation, 201 Washington Road, Princeton, NJ
email: {bmatei,hsawhney, ssamarasekera,jkim, rkumar}@sarnoff.comm

Abstract

We present a novel building segmentation system for densely built areas, containing thousands of buildings per square kilometer. We employ solely sparse LIDAR (Light/Laser Detection & Ranging) 3D data, captured from an aerial platform, with resolution less than one point per square meter. The goal of our work is to create segmented and delineated buildings as well as structures on top of buildings without requiring scanning for the sides of buildings. Building segmentation is a critical component in many applications such as 3D visualization, robot navigation and cartography. LIDAR has emerged in recent years as a more robust alternative to 2D imagery because it acquires 3D structure directly, without the shortcomings of stereo in untextured regions and at depth discontinuities.

Our main technical contributions in this paper are: (i) a ground segmentation algorithm which can handle both rural regions, and heavily urbanized areas, where the ground is 20% or less of the data. (ii) a building segmentation technique, which is robust to buildings in close proximity to each other, sparse measurements and nearby structured vegetation clutter, and (iii) an algorithm for estimating the orientation of a boundary contour of a building, based on minimizing the number of vertices in a rectilinear approximation to the building outline, which can cope with significant quantization noise in the outline measurements.

We have applied the proposed building segmentation system to several urban regions with areas of hundreds of square kilometers each, obtaining average segmentation speeds of less than three minutes per km^2 on a standard Pentium processor. Extensive qualitative results obtained by overlaying the 3D segmented regions onto 2D imagery indicate accurate performance of our system.

1. Introduction

Building modeling has numerous applications in a wide variety of tasks such as urban planning, cartography, 3D visualization for virtual city tours and autonomous robot navigation. In many practical situations, it is desired to obtain 3D models for areas of hundreds and thousands of square kilometers within days and with minimal manual interaction. Automating building segmentation is a critical component in any 3D modeling system because it provides 3D regions (segments) with little or no manual interaction. These regions can be refined by fitting simple parametric 3D shapes (primitives) that require significantly less storage compared to using polygonal meshes computed over the raw data and allow real-time online rendering.

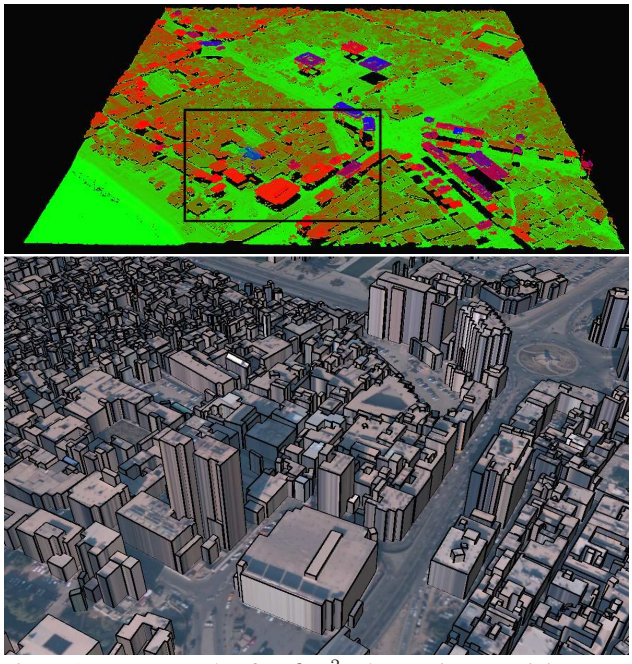

Figure 1. *Top*: Example of a 1 km^2 urban region containing more than 1.4 million points that are rendered with pseudo-color as a function of the height above ground. Red points represent higher heights than green. *Bottom*: Rendering of 3D buildings segmented and delineated by our proposed algorithm, and extruded with the height of the respective strcutures. The area enclosed by the rectangle in the top figure is depicted with aerial co-registered 2D imagery used as texture map. Note the lack of structure and texture on the sides of the buildings.

Traditionally, building modeling has employed data collected either from the ground, or from the air. Ground-based data provides high details on façades of buildings, but lacks information on tops of buildings. Aerial data can yield very accurate building footprints and can facilitate the fusion of multiple images through ortho-rectification to cover large geographical areas [10], however it lacks side information.

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

Most of the research on building modeling has employed 2D imagery, which is very inexpensive to acquire, but poses inherent difficulties in the automatic 3D modeling due to known limitations of stereo algorithms to recover the 3D structure from multiple views, especially in untextured areas and at depth discontinuities [7]. Because of these limitations, most image-based modeling systems proposed were either manual, or semi-automatic and required a long time for producing 3D models [4, 13, 18].

LIDAR (Light Detection & Ranging) has emerged in the last years as a viable and cost-effective alternative to using solely 2D imagery, because it can directly produce precise and accurate range information and can significantly alleviate the task of automatic building segmentation [14, 19, 22]. See [3] for an excellent review of various LIDAR sensors.

Aerial LIDAR collections can acquire data over entire cities very rapidly, however due to constraints imposed by the operating conditions on the altitude and speed of the airborne platform, the resolution of the data is typically less than one point per square meter. LIDAR measurements from multiple views are typically co-registered together in the same global coordinate system by tracking the pose of the LIDAR sensor as it acquires the data, using GPS and inertial measurement units (IMU). Errors in the GPS can lead to local registration errors which need to be addressed during building segmentation.

We present a novel method for automatic building segmentation in very crowded urban areas, using solely aerial LIDAR data. We have assumed that a building is uniquely determined by its roof, which has a distinct orientation compared to surrounding objects, hence we use roof and building interchangeably in this paper. Graph methods such as [21] can be used to group roofs into unique buildings with more complicated structures.

This paper makes the following technical contributions: (i) a ground segmentation algorithm which can handle both rural regions and heavily urbanized areas, where the ground is 20% or less of the data, (ii) a building segmentation technique, which is robust to buildings in very close proximity to each other, sparse measurements and nearby structured vegetation clutter, and (iii) a minimum description length algorithm for finding the orientation of a building segment based on minimizing the number of vertices in a rectilinear approximation to the building outline.

Our building segmentation system can handle regions, containing more than several thousand buildings per km^2, using sparse LIDAR data with resolution less than one point per square meter. It can process, on average 1.5 million points per km^2 in less than three minutes on a Pentium machine. A quantitative analysis of the 3D models obtained from range data alone is very difficult in practice and the performance of the 3D models produced is usually assessed using 2D data which provides textures for visualization [14]. We will use a similar approach for our performance evaluation.

A block diagram of the building segmentation system is presented in Figure 2. An example of the building segments produced using aerial LIDAR data is given in Figure 1. Note the complexity of the area handled and the quality of the building models obtained. The three main computational blocks in our system are described in the technical sections. In Section 2 we present related work in computer vision and photogrammetry for building segmentation and modeling. Ground segmentation and terrain modeling is discussed in Section 3. In Section 4 we introduce the building segmentation algorithm. The Minimum Description Length based building orientation algorithm is described in Section 5, while Section 6 presents the experimental results.

2. Related Work

In several LIDAR based building segmentation algorithms, the 3D points are classified in three classes: terrain, clutter and building. First, the 3D measurements are classified as ground and non-ground, and subsequently the non-ground points are divided into clutter and building regions [14, 19, 22].

Kraus and Pfeifer [11] presented an iterative method for terrain modeling based on removing at each step the 3D measurements with residuals to the current terrain surface larger than a threshold and re-estimating the terrain surface using the remaining data. Because the initialization of the terrain used all the data, the method may not converge for densely built regions.

Morphological opening operators are used to create a digital terrain model (DTM) which is subtracted from the input data. These filters, inspired from image processing may fail to produce good ground segmentation, especially for nonflat terrain [14]. In [21], the authors segmented the ground and the buildings in one step by computing local planar patches (surfels) using Total Least Squares (TLS) and connecting the consistent neighboring surfels into regions using bottom-up region growing. The largest region was selected as ground, while the rest of the regions were classified as individual buildings.

Rottensteiner et al. [15] present a method for the automatic delineation of roof planes using local plane estimation and grouping the local planes into larger regions starting from local seed locations. Over-segmented regions are merged together using co-planarity constraints. Points that do not belong to any surface are labeled as clutter.

Belief propagation (BP) is applied for segmentation tasks with a similar formulation as ours. For example, Sun et al. [17] used a max-product BP algorithm for stereo matching to enforce constraints that neighboring pixels with the same intensity values are assigned the same depth.

Guo et al. [8] employed a rectilinear approximation to

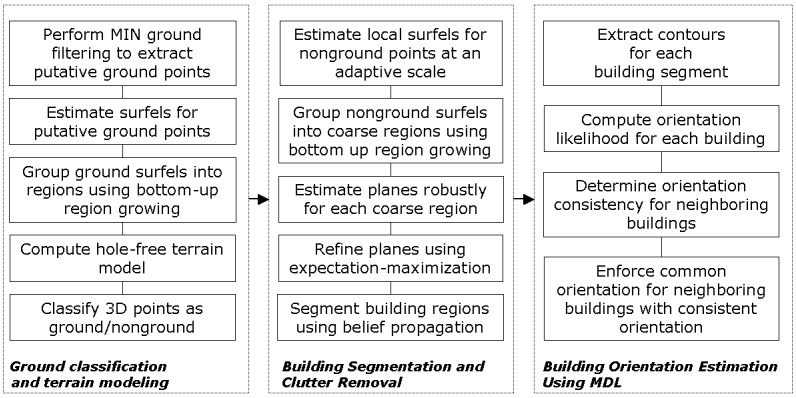

Figure 2. Diagram of the building segmentation method proposed.

outlines of buildings in their image based building detection system. The orientation of each rectilinear fit was determined by finding the maximum in a histogram of local image gradients approximating tangent directions to the contour. Their method assumes that errors of the contour points are relatively small.

3. Ground Classification and Terrain Modeling

In the first stage of our system, each 3D point is classified into ground $\mathcal{G}$ and non-ground $\mathcal{N}$. Let x_m, x_M, y_m, y_M, z_m and z_M be the minimum and maximum bounds of the 3D points along the $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{Z}$ axes. Denote by $(i,j,k) = \Upsilon(\boldsymbol{p}, v_g)$ the voxelization transformation which maps a 3D point $\boldsymbol{p}$ into a voxel of a 3D grid, with v_g being the length of the voxel side.

LIDAR sensors do not penetrate solid surfaces, hence ground points must have the minimum height in the vicinity of buildings. We employ this constraint to remove 3D measurements that cannot be part of the ground. We calculate an image $\mathsf{A} \in \mathbb{R}^{N_x \times N_y}$ containing at each pixel the minimum z value of all the points that project into it, where $N_x = \lceil (x_M - x_m)/v_g \rceil$, $N_y = \lceil (y_M - y_m)/v_g \rceil$ and $\lceil x \rceil$ is ceiling function representing the smallest integer larger than x. We run a 2D minimum filter with radius $\lceil L_{\max}/v_g \rceil$ over A to produce an image B containing at a pixel $B_{i,j}$ the ground minimum within a local neighborhood. The constant $L_{\max}$ is chosen function of the largest distance between the projections onto the XY plane of any point $\boldsymbol{p}_a \in \mathcal{N}$ and its closest ground point $\boldsymbol{p}_b \in \mathcal{G}$ (the Hausdorff distance). An upper bound on the height difference between $\boldsymbol{p}_a$ and $\boldsymbol{p}_b$ is $H_{\max} = L_{\max} |\tan(\phi_{\max}|$ where $\phi_{\max}$ is the maximum slope of the ground and we have used the monotonicity of the $\tan()$ function.

Denote by Γ the set of voxels containing at least one point $\boldsymbol{p}_u$ obeying $p_u^z \leq B_{i,j} + H_{\max} + 6\sigma_\nu$, $(i,j,\cdot) = \Upsilon(\boldsymbol{p}_u, v_g)$, where σ_ν is the standard deviation of the noise, assumed known from the operating conditions. We have employed $\sigma_\nu = 0.15\,\text{m}$. At each voxel $(i,j,k) \in \Gamma$, we fit a local planar model at several scales $\rho_1 < \ldots < \rho_S$ using the Total Least Squares (TLS) algorithm. For each scale ρ_s we compute the centroid $\tilde{\boldsymbol{c}}(\rho_s)$ and the normalized scatter matrix $\mathsf{C}(\rho_s)$, of the points within radius ρ_s from the voxel center $\boldsymbol{q}_{i,j,k}$. A plane (surfel) estimate $\pi = (\boldsymbol{n}, \boldsymbol{o})$ is represented by the normal $\boldsymbol{n}$, $\|\boldsymbol{n}\| = 1$, and the origin $\boldsymbol{o}$. The TLS plane estimate is obtained by computing the singular value decomposition (SVD) of the matrix $\mathsf{C}(\rho_s)$ and selecting $\boldsymbol{n} = \boldsymbol{\theta}_3$, the smallest singular vector of $\mathsf{C}(\rho_s)$ and $\boldsymbol{o} = \tilde{\boldsymbol{c}}(\rho_s)$. The plane estimate is kept only if the singular values $\sigma_3^s \leq \sigma_2^s \leq \sigma_1^s$ satisfy the planarity constraints $\sigma_3^s/\sigma_2^s < \tau_p$, $\sigma_2^s/\sigma_1^s > \tau_\kappa$. We have used $\rho_S = 3\,\text{m}$, $\tau_p = 0.1$ and $\tau_\kappa = 0.05$.

The surfel defined at the scale ρ_{s_0} which yields the minimum normalized estimation error $\sigma_3^{s_0}$ is retained and assigned to voxel $\Upsilon(\boldsymbol{o}, v_g)$. Note that due to noise it is possible to have $(i,j,k) \neq \Upsilon(\boldsymbol{o}, v_g)$. If a voxel is assigned more than one surfel, the one with the smallest error σ_3 is retained. We remove surfels with normals orthogonal to the vertical direction, because aerial data has very few points sampled from vertical surfaces (e.g., walls, fences).

We did not employ a robust surfel estimation using RANSAC because of the prohibitive extra computation required for the amount of data handled. For example, for only $1\,\text{km}^2$ area we need to estimate millions of surfels and we would have to employ at least few hundred trials for each surfel estimate, given the ratio of outliers caused by nearby structures. A RANSAC trial requires an SVD decomposition and likelihood evaluation, which would have to be carried out hundreds of millions of times for $1\,\text{km}^2$, rendering the ground extraction impractical.

Let $\Gamma_1 \subseteq \Gamma$ be the subset of putative ground voxels containing surfels, and let $\Gamma_2 \subseteq \Gamma_1$ be the subset of Γ_1 that contains voxels having at least one point $\boldsymbol{p}_u$ satisfying $p_u^z \leq B_{i,j} + h_0$, $h_0 < H_{\max} + 6\sigma_\nu$. We have used $h_0 = 4\,\text{m}$. For each voxel $(i,j,k) \in \Gamma_1$ we compute a histogram $\mathcal{H}_{i,j,k}$ of normal coherence measures $|\boldsymbol{n}_{i',j',k'}^\top \boldsymbol{n}_{i,j,k}|$ for all voxels $(i',j',k') \in \Gamma_1$, $\|\boldsymbol{q}_{i',j',k'} - \boldsymbol{q}_{i,j,k}\| \leq R$, with $R = 10\,\text{m}$. The entropy of the histogram $\mathcal{H}_{i,j,k}$ yields a planarity mea-

sure at a 3D location.

We select a set of seed voxels $\Gamma_3 \subseteq \Gamma_2$ by sorting the voxels in Γ_2 in the increasing value of the entropy measures and utilize a greedy strategy to select voxels, followed by non maxima suppression to ensure spatial separation. Starting at each seed voxel from Γ_3 we perform a 26-connected-component region growing by recursively adding new voxels from Γ_1 that have consistent normals with their immediate neighbors. A voxel can be assigned to at most one ground region Γ_{g_i}. The resulting ground voxel set is $\Gamma_g = \{\Gamma_{g_i} \,|\, |\Gamma_{g_i}| > A_{\min}/v_g^2\}$, with $|\Gamma_{g_i}|$ denoting the number of voxels in Γ_{g_i}. We have used $A_{\min} = 500\,\mathrm{m}^2$ based on experimentation. Γ_g provides only a coarse ground estimate because: (i) the voxel size v_g used may be larger than the noise σ_ν; (ii) there may be ground points which belong to voxels without surfels (e.g., near buildings); (iii) Some ground regions (courtyards), may have areas less than $A_{\min}$.

We compute a digital terrain model (DTM), which is represented as an an image T containing at each pixel $T_{i,j}$ the sampled ground height. The size of the pixel, w, is chosen small enough to ensure that constant ground height is a valid approximation. We initialize the DTM using height values sampled from Γ_g and apply kernel smoothing [9, p.174] to smooth out the surfel estimates and to interpolate ground at missing locations.

Some entries in the matrix T may still be undefined, for example underneath buildings. To estimate the missing ground locations, we group them into holes based on eight-connected-component neighboring constraints. For each hole region we determine the surrounding pixels from T containing valid height estimates and use them to extrapolate the height within the hole.

A point $\boldsymbol{p}_u$ is classified as nonground, $\boldsymbol{p}_u \in \mathcal{N}$, if

$$p_u^z \geq T_{i,j} + h_b\,, \quad (i,j) = \Upsilon([p_u^x, p_u^y]^\top, w)\,, \qquad (1)$$

where h_b is the minimum height of a building, specified by the user. In our system we have used $h_b = 2\,\mathrm{m}$.

4. Building Segmentation and Clutter Removal

To initialize the building segmentation, we estimate surfels at voxel locations containing nonground points $\boldsymbol{p}_u \in \mathcal{N}$ using a similar adaptive estimation as shown in Section 3. Building segmentation requires a smaller voxel size $v_b < v_g$ compared to ground classification. We use $v_b = 1$ m, since the data handled has typically a distance between samples of 1 m or more. Formally, let Φ denote the set of nonground voxels and $\Phi_1 \subseteq \Phi$ denote the subset of voxels from Φ which contains surfels. Similar to [15] we employ a bottom-up grouping of neighboring voxels that contain consistent surfels into larger, nonparametric regions $\mathcal{R}_i$. Two voxels $(i_0, j_0, k_0), (i_1, j_1, k_1) \in \Phi_1$ are considered consistent if

$$\|\boldsymbol{q}_{i_0,j_0,k_0} - \boldsymbol{q}_{i_1,j_1,k_1}\| \leq \delta,\, |\boldsymbol{n}_{i_0,j_0,k_0}^\top \boldsymbol{n}_{i_1,j_1,k_1}| \geq \kappa_1\,, \qquad (2)$$

$$|\boldsymbol{n}_{i_0,j_0,k_0}^\top \boldsymbol{d}_{0,1}| \leq \kappa_2,\, |\boldsymbol{n}_{i_1,j_1,k_1}^\top \boldsymbol{d}_{0,1}| \leq \kappa_2\,, \qquad (3)$$

with $\boldsymbol{d}_{0,1} \stackrel{\text{def}}{=} (\boldsymbol{o}_{i_0,j_0,k_0} - \boldsymbol{o}_{i_1,j_1,k_1})/\|\boldsymbol{o}_{i_0,j_0,k_0} - \boldsymbol{o}_{i_1,j_1,k_1}\|$ and κ_1, κ_2 are two thresholds and δ is the maximum distance between the closest two 3D points that belong to the same surface. We choose δ to be at least the sampling size. We have used $\delta = 2$ m, $\kappa_1 = \cos(10^\circ)$ and $\kappa_2 = \cos(85^\circ)$.

The regions $\mathcal{R}_i$ with areas larger than a threshold are classified as buildings. The measurements which are not assigned to any valid region $\mathcal{R}_i$ are classified as clutter.

4.1. Parametric Surface Refinement with EM

Bottom-up region growing offers a versatile way of grouping surfels into larger, nonparametric regions, however it suffers from several shortcomings: (i) surfel estimates have a relatively high covariance because of the small scale used for their estimation, requiring the thresholds κ_1 and κ_2 to be loose enough to guarantee that those surfels belonging to the same surface are joined. However, this may lead to region under-segmentation; (ii) buildings have flat or quadric surfaces with much less degrees of freedom compared to a connected-component region. Enforcing a parametric model globally is desirable because it improves the stability of the surface parameter estimates and can mitigate the under-segmentation occurring.

We will assume in the following that buildings have flat (planar) roofs. A second-order model (quadric) can be used in the same framework to model vaults. Let $\mathcal{R}_i$, $i = 1, \ldots, N$ be the coarse regions produced in Section 4. For clarity of presentation we will drop in the following the region index i. In each region $\mathcal{R}$ we will fit planar models robustly using MLESAC [20]. The number of planes within $\mathcal{R}$ is unknown and has to be also estimated. Let π_j, $j = 1, \ldots, P$ denote the surfels from region $\mathcal{R}$. We sample without replacement a number $B < P$ surfels from the set $\{\pi_j, j = 1, \ldots, P\}$ and obtain a set $\Delta = \{\pi_b^*, b = 1, \ldots, B\}$. We apply the algorithm from Table 1 to segment out a number of planar regions $\Pi = \{\Pi_1, \ldots, \Pi_F\}$. The set Π is refined using the expectation-maximization algorithm from Table 2.

4.2. Building Segmentation Using BP

The points assigned to any of the planes from the set Π are not guaranteed to form a contiguous region because local neighborhood constraints were not taken into account during the EM estimation. To enforce neighborhood constraints and ensure sharp boundaries between regions, we will use a loopy belief propagation (BP) framework. BP was used for solving computer vision problems with a formulation similar to ours [17]. In stereo we need to assign to each pixel in an image a label from a *known* set, corresponding to discrete depth levels, such that neighboring pixels with same color are assigned same labels.

We project all coarse regions onto the ground plane XY and extract neighborhood relationships between them. For each region $\mathcal{R}_i$, $i = 1, \ldots, N$ we determine all the regions $\mathcal{R}_j$ that are within a distance $D = 5$ m from $\mathcal{R}_i$. The neigh-

Table 1. Planar segmentation algorithm using MLESAC.

1. Initialize the set $\Pi = \{\}$ and the current set of measurements $\mathcal{U} = \{\boldsymbol{p}_u \,|\, \boldsymbol{p}_u \in \mathcal{R}\}$. Let $F = 1$.
2. Calculate the MLESAC cost

$$\mathcal{L}_b^* = \sum_{p_u \in \mathcal{U}} d^2(\boldsymbol{p}_u, \pi_b^*),\ \pi_b^* \in \Delta, \quad \text{where} \quad (4)$$

$$d^2(\boldsymbol{p}_u, \pi_b^*) = \min\left(D_{\max}^2, |\boldsymbol{n}_b^{*\top}(\boldsymbol{p}_u - \boldsymbol{o}_b^*)|^2\right)$$

and $D_{\max}$ is the inlier distance. We have used $D_{\max} = 3\sigma_\nu$.

3. Select the hypothesis $\Pi_F \stackrel{\text{def}}{=} \pi_{b_F}^* \in \Delta$ yielding the smallest cost (4). Assign the points $\boldsymbol{p}_u \in \mathcal{U}$ to the plane $\Pi_F = (\boldsymbol{n}_F, \boldsymbol{o}_F)$ if $|\boldsymbol{n}_F^\top(\boldsymbol{p}_u - \boldsymbol{o}_F)| \leq D_{\max}$. Remove the points assigned to plane Π_F from $\mathcal{U}$.
4. Refine the plane hypothesis Π_F using the measurements $\boldsymbol{p}_u$ assigned to it. Insert Π_F to the set Π. Remove from the hypothesis list Δ all surfels which are consistent with Π_F. Two planes (surfels) $\pi_1 = (\boldsymbol{n}_1, \boldsymbol{o}_1)$ and $\pi_2 = (\boldsymbol{n}_2, \boldsymbol{o}_2)$ are considered consistent when $|\boldsymbol{n}_i^\top(\boldsymbol{o}_1 - \boldsymbol{o}_2)| \leq D_{\max}$, $i = 1, 2$.
5. If the number of measurements in $\mathcal{U}$, $|\mathcal{U}|$, satisfies $|\mathcal{U}| \leq 3$, or the number of hypotheses $|\Delta| = 0$, STOP. Otherwise, set $F = F + 1$ and go to Step 2.

borhood relationships between regions form a graph. We assume that the neighborhood relationships are transitive, i.e. if $\mathcal{R}_i \leftrightarrow \mathcal{R}_j$ and $\mathcal{R}_j \leftrightarrow \mathcal{R}_k$, we have $\mathcal{R}_i \leftrightarrow \mathcal{R}_k$. We extract cliques of regions forming disjoint sets. Clique extraction from graphs is an NP-complete problem so we use a sub-optimal approach based on greedy assignment.

Denote by $\mathcal{B}$, one of the cliques resulted. Because each clique is independent of each other, the segmentation task can be distributed to be processed in parallel. Let $x_m^{\mathcal{B}}$, $x_M^{\mathcal{B}}$, $y_m^{\mathcal{B}}$, $y_M^{\mathcal{B}}$ be the bounding box of the regions in $\mathcal{B}$ projected onto horizontal plane. We assign to $\mathcal{B}$ all the points $\boldsymbol{p}_u$ having the x and y components withing the bounding box of $\mathcal{B}$. Let f_{p_u} be the label of $\boldsymbol{p}_u$, with $f_{p_u} \neq 0$, if it is assigned to a plane region $\Pi_{f_{p_u}}$ assigned to a region from $\mathcal{B}$ and $f_{p_u} = 0$ otherwise.

We initialize a set $\Psi = \{\Psi_1, \ldots, \Psi_J\}$ and $\Psi_m = \{\boldsymbol{p}_u \in \mathcal{B} \,|\, \forall \boldsymbol{p}_v \in \Psi_m,\ \|\boldsymbol{p}_u - \boldsymbol{p}_v\| \leq \delta,\ f_{p_u} = f_{p_v}\}, m = 1, \ldots, J$, using greedy recursive grouping. This initial partitioning provides an estimate on the number of labels (regions) within $\mathcal{B}$. The choice of δ was discussed in (2). A point $\boldsymbol{p}_u \in \mathcal{B}$ is assigned to at most one region $\Psi_{m_{p_u}}$.

Define $\mathsf{V} \in \mathbb{R}^{n_x \times n_y}$, $n_x = \lceil (x_M^{\mathcal{B}} - x_m^{\mathcal{B}})/v_b \rceil$, $n_y = \lceil (y_M^{\mathcal{B}} - y_m^{\mathcal{B}}))/v_b \rceil$ and assign all $\boldsymbol{p}_u \in \mathcal{B}$ to V. To simplify notations we represent a cell $V_{i,j}$ using one index, V_q. Due to visibility constraints, all the points belonging to V_q will share the same label with V_q.

Table 2. Top-down region refinement using the EM algorithm.

1. Initialize $t = 0$ and $\Pi_f^0 = \Pi_f$, $f = 1, \ldots, F$.
2. **Expectation**: For all $\boldsymbol{p}_u \in \mathcal{R}$ compute the probability (weight) of assigning $\boldsymbol{p}_u$ to $\Pi_f^t = (\boldsymbol{n}_f^t, \boldsymbol{o}_f^t)$

$$\xi(\boldsymbol{p}_u, \Pi_f^t) = \left[1 - \left(\frac{(\boldsymbol{p}_u - \boldsymbol{o}_f^t)^\top \boldsymbol{n}_f^t}{D_{\max}}\right)^2\right]^2,$$

if $|(\boldsymbol{p}_u - \boldsymbol{o}_f^t)^\top \boldsymbol{n}_f^t| \leq D_{\max}$, and $\xi(\boldsymbol{p}_u, \Pi_f^t) = 0$, otherwise. As in Table 1, $D_{\max} = 3\sigma_\nu$.
3. **Maximization**: Reestimate the planes Π_f^{t+1}, using a close-form variant of the HEIV algorithm [12], in which each constraint is weighted by $\xi(\boldsymbol{p}_u, \Pi_f^t)$. If changes occur, set $t = t + 1$ and go to Step 2.
4. Assign $\boldsymbol{p}_u \in \mathcal{R}$ to the plane $\Pi_{f_0}^t$

$$f_0 = \arg\min_{f=1,\ldots,F} |(\boldsymbol{p}_u - \boldsymbol{o}_f^t)^\top \boldsymbol{n}_f^t|, \quad (5)$$

if $|(\boldsymbol{p}_u - \boldsymbol{o}_{f_0}^t)^\top \boldsymbol{n}_{f_0}^t| \leq D_{\max}$.
5. Repeat: merge any two planes $\Pi_{f_1}^t$, $\Pi_{f_2}^t$ if

$$(\boldsymbol{\beta}_{f_1} - \boldsymbol{\beta}_{f_2})^\top \mathsf{C}_{\Pi_{f_{1,2}}}^\dagger (\boldsymbol{\beta}_{f_1} - \boldsymbol{\beta}_{f_2}) \leq \chi_{5,0.95}^2,$$

where $\boldsymbol{\beta}_{f_i}^\top = [\boldsymbol{n}_{f_i}^\top, \boldsymbol{o}_{f_i}^\top]$, $\mathsf{C}_{\Pi_{f_i}} \in \mathbb{R}^{6\times 6}$,is the rank five covariance matrix of the plane estimates $\Pi_{f_i}^t$, $i = 1, 2$, estimated as in [12], $\mathsf{C}^\dagger$ is the pseudo-inverse of C and $\chi_{p,1-\alpha}^2$ is a quantile of a χ^2 distribution with p degrees of freedom and coverage $1 - \alpha$. Reestimate the merged plane and update $F = F - 1$.

At V_q, we associate the *belief vector* $\boldsymbol{\lambda}_q \in \mathbb{R}^J$ containing the probability that V_q belongs to each of the J regions. We calculate the set of labels $\zeta_q = \{m_{p_{u'}} \,|\, m_{p_{u'}} = 1, \ldots, J\}$, $|\zeta_q| \leq J$ of all the points $\boldsymbol{p}_{u'}$ within a small neighborhood of $\boldsymbol{p}_u$ having $m_{p_{u'}} \neq 0$.

At a cell V_q we compute the data term $\boldsymbol{\nu}_q \in \mathbb{R}^J$

$$\nu_q(m) = \begin{cases} \exp\left(-\frac{d^2(\boldsymbol{p}_u, \Pi_m)}{2\sigma_\nu^2}\right), & \text{if } d(\boldsymbol{p}_u, \Pi_m) \leq 3\sigma_\nu \\ \exp(-9/2), & \text{otherwise} \end{cases},$$

$$d(\boldsymbol{p}_u, \Pi_m) = \min_{p_u \in V_q} |(\boldsymbol{p}_u - \boldsymbol{o}_p^m)^\top \boldsymbol{n}_p^m|.$$

We employ a Potts model to model the interaction between 4-connected neighboring cells cells V_q and V_r

$$\omega_{q,r}(m_q, m_r) = \begin{cases} P_{potts}, & \text{if } m_q \neq m_r \\ 1, & \text{otherwise} \end{cases}, \quad (6)$$

because it reduces the complexity of BP from quadratic to linear [5]. We have used $P_{potts} = 0.01$.

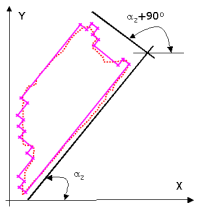

Figure 3. Example of rectilinear fit for a 2D closed contour.

We apply a *max-product* algorithm which maximizes the joint probability of label assignment over all cells [2]. The message from V_q to V_r is calculated as

$$\psi^t_{q\to r}(m) = \max\left(\mu_q(m), a \cdot P_{potts}\right), \tag{7}$$

with $\boldsymbol{\mu}_q = \boldsymbol{\nu}_q \prod_{s\in\mathfrak{N}(q)\setminus r} \boldsymbol{\psi}^{t-1}_{s\to q}$, $a \stackrel{\text{def}}{=} \max_m \mu_q(m)$ and $\mathfrak{N}(q)$ is the neighborhood of q. The belief at node V_q is

$$\boldsymbol{b}_q = \boldsymbol{\nu}_q \prod_{s\in\mathfrak{N}(q)} \boldsymbol{\psi}^T_{s\to q}, \tag{8}$$

where T is the last iteration of (7). The final label assigned to V_q is $m_q = \arg\max_{m=1,\ldots,J} b_q(m)$.

5. Building Orientation Estimation

In this Section we describe a novel method for finding the orientation of building segments which allows us to initialize the fitting of 3D polyhedral objects to the building segments extracted. These 3D objects allow the rendering of the buildings using standard graphics languages such as OpenGL and can be exported into Geographic Information Systems (GIS) for further manipulation and refinement. They can also be integrated with other mapping products, such as road networks or utility maps. The extraction of 3D models and their fitting is beyond the scope of the paper.

Most buildings have normals to their roofs very close to the vertical direction $\boldsymbol{Z}$, hence finding the orientation α_z of the building regions around the $\boldsymbol{Z}$ should be addressed first. The orientations around the $\boldsymbol{X}$ and $\boldsymbol{Y}$ directions, α_x, α_y can be easily determined from the knowledge of the normal $\boldsymbol{n}$ to the surface region, if α_z is known. We will discuss in the following the estimation of α_z for a building segment.

For each building segment $\mathcal{R}$ we compute the 3D boundary (outline) using the ball pivoting algorithm [1] and project it onto the horizontal plane. The outlines typically have a significant quantization noise due to sparseness of the data acquired from the air. Finding the orientation of the contour by computing the maximum of the histogram of local tangent directions to the curve as in [8] did not work in our case because of the contour noise which renders the local tangent estimates too noisy leading to orientation histograms without any clearly defined peaks. Attempts based on finding salient points on the contour using the adaptive method of Fischler and Wolf [6] did not yield good results either. The main shortcoming of local methods in estimating the global orientation of an outline is caused by the difficulty in finding an adaptive scale at each contour location,

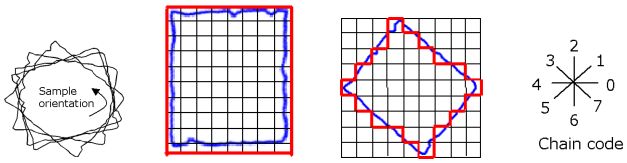

Figure 4. (a) We rotate a closed contour (outline) at orientations sampled from 0° to 90° and fit rectilinear structures to each orientation. (b), (c) Illustration of rectilinear fits depending on a specific orientation. Note that in (b) the number of vertices required to approximate the curve is much smaller than in (c). (d) Chain code utilized to represent the direction of an edge. In a rectilinear polygon, the edges can have only directions encoded by even numbers.

which yields enough support to compensate for the noise, yet is robust to big jumps in orientation.

To cope with the local contour noise we employ a rectilinear polygon representation to a building outline, which approximates well a large class of buildings encountered. A rectilinear polygon has the angle between any consecutive sides equal to $\pm 90°$ and an even number of vertices. A rectangle, for example, is a particular case of rectilinear polygon. The rectilinear polygons can approximate any closed contour. An example of such fit is shown in Figure 3. The 2D orientation of any polygon can be uniquely specified by an angle between 0° and 180°. In the case of the rectilinear polygon, due to the fact that the angles between any two sides are a multiple of 90°, the orientation of the rectilinear shape can be specified uniquely using $\alpha_z \in [0°; 90°]$. The rectilinear fit procedure is very simple and requires very little computations. The points from the outline are assumed ordered in a counter-clockwise order and can be quantized in an image having length of the pixel w_c. We select w_c based on the typical edge of a building. We have employed $w_c = 5\,\text{m}$.

Based on the order of traversal of the contour points we determine the pixel traversal order. We determine the upper-left occupied contour pixel and begin to traverse the contour, pixels however at every step we can move only along the horizontal and vertical direction. To represent the direction of a side in the rectilinear polygon we employ the chain code from Figure 4(d). Let (i_k, j_k) be the current pixel and (i_{k+1}, j_{k+1}) the next pixel on the contour. The direction γ_{k+1} is found from the offset $(i_{k+1} - i_k, j_{k+1} - j_k)$. If γ_{k+1} is an even number we introduce a corner if $\gamma_{k+1} \neq \gamma_k$. If γ_{k+1} is an odd number we need to introduce an extra corner and approximate the direction with two edges along the $\boldsymbol{X}$ and $\boldsymbol{Y}$ directions.

From Figure 4 it can be seen that the number of vertices in the rectilinear outline approximation varies significantly with the change in orientation. Similar to the Minimum Description Length (MDL) criterion we seek to minimize the number of vertices in the rectilinear representation of the curve. For each angle α_k sampled uniformly

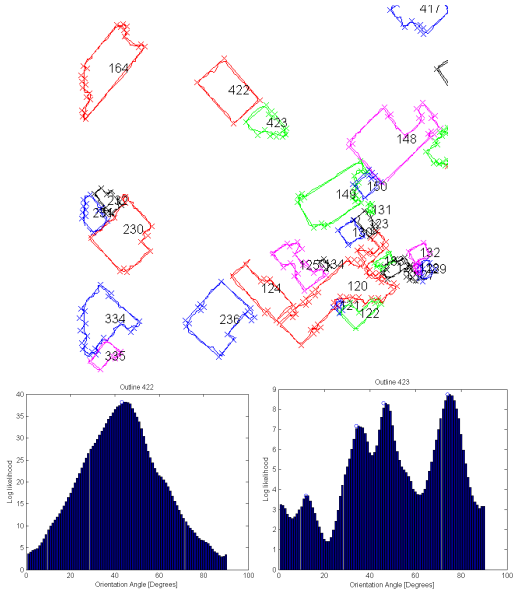

Figure 5. (Top) Example of rectilinear fits for several contours. Contours are represented by continuous lines. The rectilinear fits are represented by curves with '+' markers. (Bottom left) Proposed log-likelihood dependency on the orientation for contour 422 from (a). (Bottom right) Same as (Bottom left), but for contour 423. There is a higher degree of uncertainty resulting in multiple peaks denoted by circles caused by the smaller outline and the more complex outline.

from 0° to 90° we fit a rectilinear polygon and retain the number of vertices Λ_{α_k}. We compute the log-likelihood $\log(L(\alpha_k)) = \Lambda - \Lambda_{\alpha_k}$, where $\Lambda = \max_{\alpha_k} \Lambda_{\alpha_k}$, thus we seek to find the angle α_k which maximizes $\log(L(\alpha_k))$. Observe in Figure 5 the discriminability yielded by the likelihood method proposed.

In most urban layouts the buildings within a city block have the same orientation (facing the street). To enforce this constraint, we employ the cliques of neighboring buildings extracted in Section 4.2. For each outline we extract the modes of log-likelihood using mean-shift and retain only the modes which have a likelihood ratio higher than 0.8 compared to the best peak. We determine the neighboring outlines (regions) which have consistent orientation (i.e., they have peaks within 10° of each other). For all the neighboring outlines with consistent orientation we add the individual log-likelihoods and determine the angle corresponding to the maximum. If α is the final angle estimate for a contour, we compute the interval of the 2D projection of the contour along the direction corresponding to α and $\alpha + 90^\circ$ and retain the angle leading to the largest interval.

6. Experimental Results

We have applied the building segmentation method presented in this paper to a very large number of data sets covering urban areas of hundreds of square kilometers with various terrain and urban density. Typically, the regions were very densely built with thousands of buildings per km^2 and very little spatial separation between them. The LIDAR measurements from multiple views were registered to a common coordinate system resulting in data with resolution up to 1 point/m^2. However, in areas of little overlap between views, the resolution can be worse than 0.25 points/m^2. Together with LIDAR data we have available 2D imagery acquired at the same time with the LIDAR. For each image we have available the intrinsic and extrinsic parameters (rotation and translation) of the camera in the global coordinate of the LIDAR. The extrinsic parameters give only an initialization and have to be refined by image to image and image to LIDAR alignment. We rely on SIFT matching to align the 2D imagery with each other using an affine model. The image to image 2D poses and image to LIDAR 3D poses are fed into an optimization process similar to the local-to-global framework from [16]. After image and LIDAR registration we ortho-rectify the input 2D imagery using the available LIDAR 3D data to render them from nadir viewpoint.

We partition the 3D data into square tiles with 0.6 km sides and to eliminate boundary effects we use a padding of 200 m around each side of the tile resulting in overlapping tiles with 1 km edges, which are fed to the building segmentation system. For all the experiments carried out we have employed the same parameters, selected to ensure that no buildings are missed at building segmentation stage. For each tile, we follow the diagram from Figure 2 to first eliminate the ground, then extract building segments and remove clutter, and finally obtain the orientation for each contour.

A quantitative analysis of the building segmentation algorithm is very hard over an extended area, due to difficulties in extracting ground truth. We follow [14] to rely on 2D imagery in assessing the accuracy of the building segments proposed. We fit 3D objects which are water-tight to each building segment extracted using the initialization of the orientation obtained as in Section 5. The 3D objects are texture-mapped using the ortho-rectified 2D imagery and can be rendered in graphics hardware. The textured-mapped 3D objects are fed into a visualization system and the quality of the buildings assessed visually by comparing the consistency of the 3D shapes with the edges of buildings in an image.

We have run the proposed building segmentation on various data sets, with the largest one having 800 km^2. The algorithm was used without tuning parameters for each tile processed. On average, we processed a tile in three minutes per km^2 tile using a 3.6 GHz CPU and 2 GBytes of RAM. The texture mapping and additional model fitting (not discussed in the paper) required seven more minutes per km^2 tile. Because the processing is highly parallelizable we have employed several machines, requiring about two days to produce 3D models for the whole area.

In Figure 6 we show the output of our building segmen-

tation method proposed over a $1\,\mathrm{km}^2$ tile. 3D objects were extruded to the ground plane using the rectilinear fit corresponding to the orientation of the segment, estimated as in Section 6. There is no texture on the sides because we don't have imagery collected from the ground to cover the façades. In Figure 6, top we show the building segments, each building being shown in a distinct color, while in the middle and bottom we show the 3D texture objects created together with the outlines. More detailed results can be found in the supplemental material attached.

Acknowledgements

This research was conducted under the Digital Video Laboratory of the 46-th Range Group, Eglin A.F.B..

References

[1] F. Bernardini, J. Mittleman, H. Rushmeier, C. Silva, and G. Taubin. The ball-pivoting algorithm for surface reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 5(4):349–359, 1999. 6

[2] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. 6

[3] F. Blais. Review of 20 years of range sensor development. *Journal of Electronic Imaging*, 13(1):231–240, 2004. 2

[4] M. Drauschke, H. Schuster, and W. Forstner. Detectability of buildings in aerial images over scale space. In *PCV06*, 2006. 2

[5] P. Felzenswalb and D. Huttenlocher. Efficient belief propagation for early vision. *IJCV*, 70(1):41–54, 2006. 5

[6] M. Fischler and H. Wolf. Determinining local natural scales of curves. *PAMI*, 16:113–129, 1994. 6

[7] D. Gallup, J. Frahm, P. Mordohai, Q. Yang, and M. Pollefeys. Real-time plane-sweeping stereo with multiple sweeping directions. In *CVPR*, pages 1–8, 2007. 2

[8] Y. Guo, H. Sawhney, R. Kumar, and S. Hsu. Learning-based building online detection from multiple aerial images. In *ECCV*, pages 545–552, 2001. 2, 6

[9] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning. Data mining inference and prediction*. Springer, 2001. 4

[10] J. Hu, S. You, and U. Neumann. Approaches to large-scale urban modeling. *Computer Graphics and Applications*, 23(6):62–69, 2003. 1

[11] K. Kraus and N. Pfeifer. Determination of terrain models in wooded areas with airborne laser scanner data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 53(4):193–203, 1998. 2

[12] B. Matei and P. Meer. Estimation of nonlinear errors-in-variables models for computer vision applications. *PAMI*, 28:1537–1553, 1999. 5

[13] H. Mayer. Automatic object extraction from aerial imagery: A survey focusing on buildings. *CVIU*, 74(2):138–149, May 1999. 2

[14] F. Rottensteiner and C. Briese. A new method for building extraction in urban areas from high-resolution LIDAR data. In *IAPRSIS*, volume XXXIV, pages 295 – 301, 2002. 2, 7

[15] F. Rottensteiner, J. Trinder, S. Clode, and K. Kubik. Automated delineation of roof planes from LiDAR data. In *Laser05*, pages 221–226, 2005. 2, 4

[16] H. S. Sawhney, S. C. Hsu, and R. Kumar. Robust video mosaicing through topology inference and local to global alignment. In *ECCV*, pages 103–119, 1998. 7

[17] J. Sun, H.-Y. Shum, and N.-N. Zheng. Stereo matching using belief propagation. *PAMI*, 19:787–800, 2003. 2, 4

[18] I. Suveg and G. Vosselman. Reconstruction of 3D building models from aerial images and maps. *ISPRS Journal of Photogrammetry and Remote Sensing*, 58:202–224, 2004. 2

[19] I. Suveg and G. Vosselman. The utilisation of airborne laser scanning for three-dimensional mapping. *International Journal of Applied Earth Observation and Geoinformation*, 6:177–186, 2005. 2

[20] P. Torr and A. Zisermann. MLESAC: A new robust estimator with application to estimating image geometry. *CVIU*, 78:138–156, 2000. 4

[21] V. Verma, R. Kumar, and S. Hsu. 3D building detection and modeling from aerial LIDAR data. In *CVPR*, 2006. 2

[22] S. You, J. Hu, U. Neumann, and P. Fox. Urban site modeling from LiDAR. In *Proc. 2nd Int'l Workshop on Computer Graphics and Geometric Modeling*, pages 579–588, 2003. 2

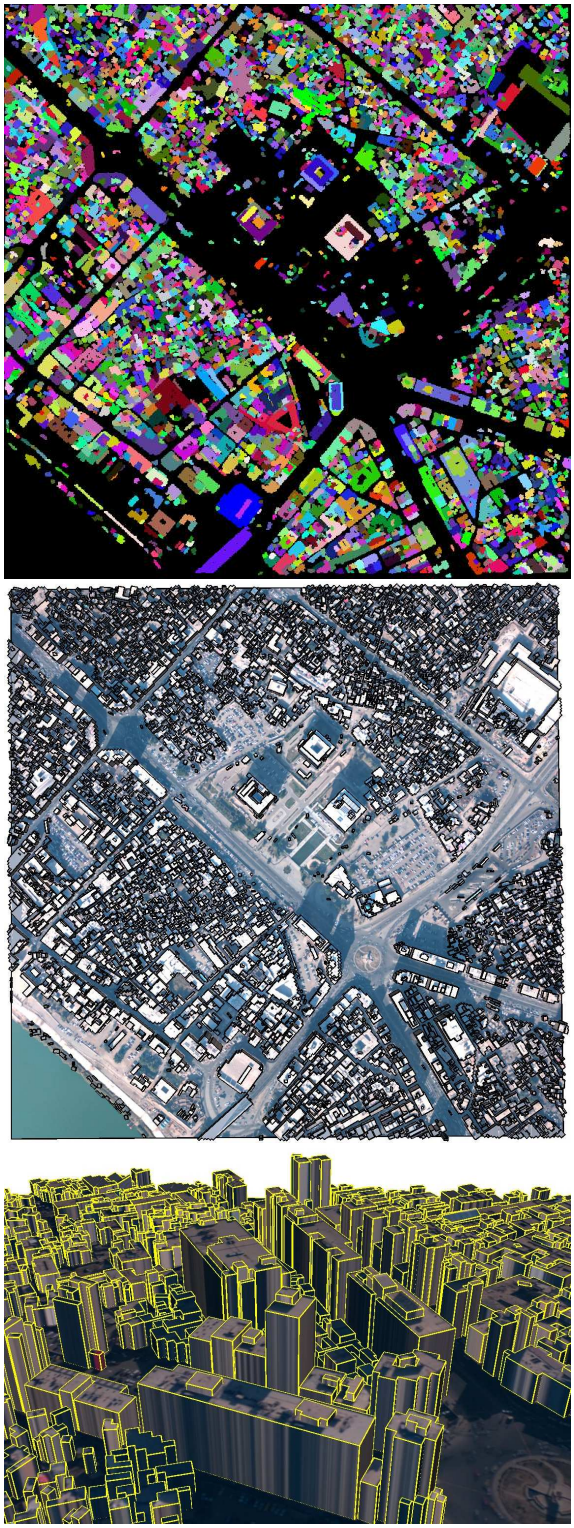

Figure 6. *Top*: Building segmentation result for a $1\,\mathrm{km}^2$ area. Each color denotes a distinct building. *Middle*: Visualization of the building segments and their orientation using 3D models obtained by extruding the rectilinear outlines corresponding to the segment orientation to the ground and texture mapping them with 2D imagery, co-registered with the LIDAR. Rectilinear contours are shown in black. *Bottom*: snapshot from a different viewpoint.