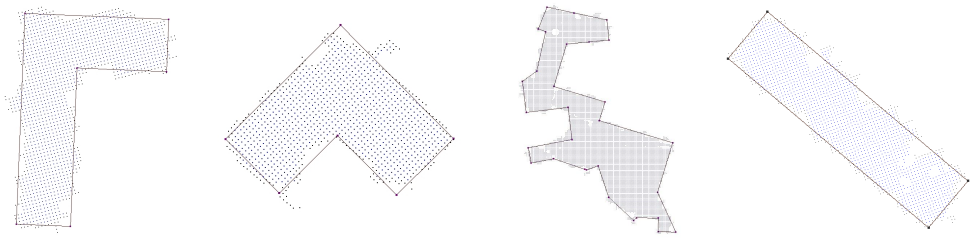
A Bayesian Approach to Building Footprint Extraction from Aerial LIDAR Data

Oliver Wang, Suresh K. Lodha, David P. Helmbold
University of California, Santa Cruz
Santa Cruz, Ca, 95064
{owang,lodha,dph}@soe.ucsc.edu



Abstract

Building footprints have been shown to be extremely useful in urban planning, infrastructure development, and roof modeling. Current methods for creating these footprints are often highly manual and rely largely on architectural blueprints or skilled modelers. In this work we will use aerial LIDAR data to generate building footprints automatically. Existing automatic methods have been mostly unsuccessful due to large amounts of noise around building edges. We present a novel Bayesian technique for automatically constructing building footprints from a pre-classified LIDAR point cloud. Our algorithm first computes a bounded-error approximate building footprint using an application of the shortest path algorithm. We then determine the most probable building footprint by maximizing the posterior probability using linear optimization and simulated annealing techniques. We have applied our algorithm to more than 300 buildings in our data set and observe that we obtain accurate building footprints compared to the ground truth. Our algorithm is automatic and can be applied to other man-made shapes such as roads and telecommunication lines with minor modifications.

1. Introduction

Building footprints are valuable tools for urban development. They have been widely used in urban planning, construction of telecommunication lines, pollution modeling, disaster planning, and many other kinds of urban simulations. In addition, building footprints not only localize buildings, but also reveal valuable information about the structure of building roofs and vertical walls that would be otherwise invisible to aerial sensors [9].

Because of their usefulness, many roof modeling techniques use building footprints to simplify the reconstruction problem. These building footprints are generally acquired by manual input or prior knowledge, often relying on architectural blueprints and skilled modelers. Blueprints are also limited in usefulness as they are generally available only for modern buildings and in large urban areas. The manual creation of building footprints requires trained personnel, which can lead to a high cost for these footprints. Any automation in this process would decrease the cost of acquisition and the reliance on outside information, thereby making urban models more accessible worldwide. This wide scale availability of building footprints could allow for more informed development decisions in many situations.

Rather than using manual input or existing models, we propose a data driven technique to generate building footprints. Aerial LIDAR technology is a good choice for the creation of 2.5D point clouds of large scale areas as it is both cheap and accurate compared to many other ranging methods. LIDAR collection involves using a time-of-flight laser ranging setup mounted on a fixed wing airplane or helicopter. Data is collected in strips as the aircraft flies overhead. As a pre-processing step, we first classify the data into buildings, trees and grass. Individual buildings are then segmented from the data. We use these segmented regions as the input to our building footprint extraction algorithm.

Techniques that use only aerial LIDAR data have the advantage that they do not rely on outside information, making data acquisition an easier task. In addition, because LIDAR

sampling density is rapidly increasing, it is reasonable to expect that the accuracy of these methods will increase directly with the density of the sampling technology. However, one common difficulty that has plagued many data-driven building footprint creation algorithms is that there is generally a lot of noise in the data points that lie around the edges of buildings. This noise can be caused by overhanging trees, sensor noise, or noise in the classification results. An example of this can be seen in Figure 1, where a noisy building boundary can be seen in the second image. Any method which works directly on the observed data points must approximate straight building edges despite the presence of noise.

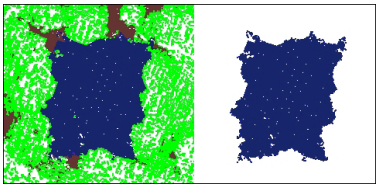


Figure 1. Aerial LIDAR data points of a rectangular building and surrounding region. The left figure contains tree and ground points. The right figure shows just building points, with high boundary noise visible.

Our approach uses a maximum a posteriori (MAP) estimation (e.g. Mitchell [16]) to determine the most likely building footprint given the noisy data and our prior knowledge of building shapes. We maximize this conditional probability by using a gradient descent optimization method. The field of optimization is mature and a variety of powerful methods exist. By formulating our problem in an optimization framework, we can take advantage of these techniques. For the initial conditions of this optimization, we first compute an approximate building footprint using a graph theoretic algorithm and greedy search. We define an objective function for a building footprint that takes into account the disparity from observed data and how well the footprint matches our prior knowledge of buildings. We solve the maximum a posteriori estimation by gradually evolving the building footprint.

While our project employs building specific priors, it would be easy to extend our algorithm to different shapes, such as roads or telecommunication lines, by simply changing the prior. We have applied our algorithm to our entire data set, which features a wide variety of building types and large amounts of noise from overhanging trees. We have visually compared our results with building footprints extracted from architectural blueprints and aerial photography in order to assess its accuracy.

This paper is organized as follows: Section 2 talks about previous work, Section 3 provides a brief introduction to LIDAR data, Section 4 explains our algorithm, Section 5 analyzes results, and Section 6 discusses our conclusion and possible future work.

2. Previous Work

The automatic and semi-automatic modeling of urban areas using a variety of sensors including aerial LIDAR data and camera images is an important area of research [11]. Most of the demand for these urban models is centered around the creation of navigable 3D virtual models of cities [18] [8] [24].

Building footprint extraction is an important step in the pipeline from data acquisition to modeling. Building footprints not only localize buildings within a point cloud, but also reduce the search space for vertical walls and step edges, which are generally believed to be one of the hardest parts of roof modeling [21]. Frequently, previous work has used pre-existing building footprints directly in their modeling techniques [10] [4]. These approaches bypass the problem that we propose as they assume previous knowledge of individual building locations and shapes.

Some previous work has used external information sources to try to find the building footprints. Brenner et al. [3] uses multi-spectral reflectance to determine image edges. In addition, standard edge detection techniques have been applied to aerial images to try to extract visual cues about the location of building boundaries [19]. Billard et al. [2] presents an approach that incorporates multi-view geometry techniques with aerial images to determine the 3D models of buildings. Kim and Nevatia [13] propose a similar method but instead they use Bayesian networks to combine edge detection information from multiple images to determine probable building walls. You et al. [24] models buildings by means of a user-driven combination of primitive roof shapes that are assembled to form complex building models. We prefer to work only from LIDAR data and with as little manual input as possible. Reducing manual input has clear cost advantages. In addition, requiring only LIDAR data reduces our dependence on the existence of outside information which may be unavailable, costly to acquire or difficult to register.

Many data driven methods of building footprint construction simplify the problem by limiting building walls to two perpendicular directions. Alharthy et al. [1] uses a histogram method to determine the two dominant directions where the walls lie. All boundary points are then connected by lines that approximate one of these two directions. Maas et al. [15] and Vosselman [23] present similar solutions for automatically extracting building footprints by analyzing a triangulated mesh of building points. They approximate the contour of the edge polygons with straight lines using the

techniques presented in Douglas-Peucker [7]. They then post-process their results in order to ensure that 80% of the points are inside the building footprint. All of these approaches have the disadvantage that they are not robust in the case of very noisy building boundaries. In addition, these algorithms assume that buildings have two sets of parallel walls, and will have difficulty on more complicated buildings. Frueh and Zakhor [8] present a data-driven method to construct an urban model from terrestrial LIDAR scans mounted on a truck. However, this approach does not explicitly model building roofs, and only provides detail visible from the ground level.

Using a combination of Bayesian probability with optimization techniques is not a new idea. It was presented to the graphics community by Szeliski and Terzopoulos [22]. Energy minimization has also been applied to general mesh smoothing and de-noising before, [12], and [17]. We use a method similar to Diebel et al. [6] who use energy minimization and penalty functions to achieve 3D mesh smoothing and simplification. However, to our knowledge, we are the first to use a maximum a posteriori estimation approach to the problem of urban modeling with building specific priors. In addition, we propose a polygon approximation step prior to the optimization and employ simulated annealing to assist in the convergence.

Our contribution is a new and robust footprint creation algorithm that is able to tolerate noise along building edges. Our algorithm is also automatic, steerable with a prior, and is not restricted to requiring only two directions for walls to exist.

3. LIDAR Data and Pre-Classification Overview

A typical aerial LIDAR system consists of a laser range finder, differential GPS, inertial navigation sensors, a computer and some storage media. This setup is mounted on fixed wing airplanes or helicopters. The data is usually acquired as a set of overlapping strips, each consisting of multiple scan lines corresponding to different passes over the region. A LIDAR data set consists of irregularly-spaced 2.5D points where the elevation z has a unique value as a function of x and y. This aerial LIDAR data is classified into three classes (building, tree, grass) using an Adaboost machine learning algorithm. This classification uses a small manually labeled data set to learn classification rules, which can then be applied to the rest of the data. Using four features – height, height variation, normal variation, and LIDAR return intensity, the classification performs at over 90% accuracy on test data sets. Therefore, aerial imagery data is not needed for either pre-classification, segmentation, or the algorithm presented in this work. The classified data is then clustered into individual buildings using an automatic segmentation algorithm. This process uses region growing on the individual building points to locate and group buildings. The output of this step is a set of individual building regions which are the inputs to our algorithm.

4. Automatic Building Footprint Extraction

Each building region is represented as an unstructured 2.5D point cloud. We consider a projection of these points onto the XY plane for the purpose of our algorithm. We use a Bayesian technique to represent the posterior probability of our building footprint. We determine initial conditions for our MAP estimation by first approximating the building footprint using an application of the shortest path algorithm.

Our algorithm has the three following steps:
(i) finding boundary points,
(ii) constructing building footprint approximation,
(iii) Find most probable building footprint using maximum a posteriori estimation approach.

4.1. Finding Boundary Points

In order to construct an approximation of the building footprint we first find the points that lie on the boundary of the point cloud. This is done by using a local neighborhood search. The key idea is that a point on the boundary should have a large region in some direction where no other points exist. Our algorithm tests each point P as follows:

For each point P_i, let C_i be the circle of radius R centered at point P_i. Find the largest angular region of C_i where no building points exist. If this angular region is larger than a threshold, make P_i a boundary point. We treat a point as being on the boundary when there is a gap of 70° or more.

It is possible that this algorithm could miss occasional boundary points in highly detailed boundary areas. However this is not a major problem as this step need only find a rough outline of the building. This initial approximation will be refined during optimization as described below. The results of the computation are shown in Figure 2.

4.2. Finding Building Footprint Approximation

We now construct an initial building footprint by ordering the boundary points with a greedy search using the 2D distance between points as a similarity metric. The objective of this step is to trace a path between neighboring nodes along the boundary of the building until we return to the starting point. Each point is labeled in the order that it is visited, starting with an arbitrary seed point. The depth first recursion is halted if the distance to the closest point is greater than a threshold, or there are no more points to add.

This threshold is determined on the basis of data density, as it should be set to around the average expected distance between points. If the closest point is significantly above the average density of points, it is unlikely that it is ordering correctly. This threshold will prevent far away points from being labeled in sequence. Figure 2 illustrates an example of the initial ordering.

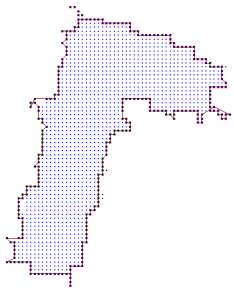


Figure 2. Boundary points obtained by step 1 are shown as bolder points. The initial ordering of the boundary points discussed in section 4.2 is shown as a red line.

This step of the algorithm may skip points in highly detailed areas, or cut off regions that are connected by only one point. We notice that this occurs mostly in tree regions that have been misclassified as buildings. Because of the half-meter resolution of our data points, it is unlikely that a building would have a segment which is only one point wide. Therefore, these cases tend not to affect building regions.

We observe in Figure 3 that noise along the boundary is preserved in the initial ordering. We attempt to reduce this by computing an approximation to the initial building footprint ordering by using a shortest path algorithm. This step is extremely important because the MAP estimation which we describe in the next section defines prior error terms on a local neighborhood, so noisy details can increase the chances of converging to a poor local minimum. For example, we can see in Figure 3 that if the original data is sufficiently noisy, the initial building footprint ordering may contain line segments belonging to the same wall, but with 90° angles between them. Because our prior has a preference to preserve right angles, these errors can keep the optimization step from converging to a good solution.

The purpose of the next step of the algorithm is to approximate this set of line segments so as to prevent local level noise from affecting the convergence of the minimization problem. We will approximate the set of linear segments of the initial ordering using a min-ϵ error criteria presented by Stone [20]. We use a graph search algorithm proposed in Dahl and Realfsen [5] to solve for the best approximation. When connecting points a and b, we define our error term as the average Euclidian distance from all the points between a and b to the line segment formed by ab. This is illustrated in Figure 4.

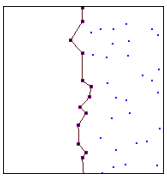


Figure 3. An initial ordering along a straight building wall is shown as a series of line segments. Noise in the boundary points leads to right angles which can hinder convergence in the optimization process.

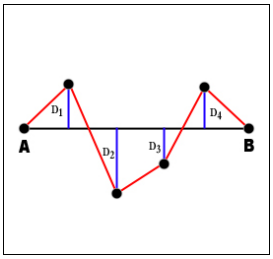


Figure 4. The error of approximating five line segments with a line segment from a to b is defined as the sum of the distance from the subsumed points to the final line segment, shown in this figure as the four blue lines d_1,d_2,d_3,d_4.

To solve the min-ϵ problem, we create a graph where each node is a point in our initial building footprint. An edge exists for each line segment in the initial building ordering obtained in the previous step. We now create new edges for all nodes which when connected have some error less than ϵ. Computing the shortest path on this graph will give us a polygon that has the minimum number of sides where each edge has error less than ϵ for a given start and end node. However, because we do not know the optimal starting node beforehand, we modify the algorithm to compute the best approximation for any given starting point.

We first wrap the initial polygon twice around itself, creating a graph with $2N$ nodes and then run the algorithm described above. We then look for the shortest path from nodes P_i to P_{i+m} over all nodes i, where m is the number of nodes in our initial graph. This essentially tests all nodes as starting nodes, and finds the one with the minimum number of sides. We use the Floyd-Warshall algorithm to compute the shortest path, so each time it is computed, we have a $O(N^3)$ solution. If we looked for the shortest path given

every starting node by recomputing the shortest path each time, we would have a $O(N^4)$ algorithm. Using this method that we present, we only need solve the shortest path once on an input size of $2N$, which gives us a quick $O(2N)^3$ algorithm. Figure 5 shows an example of the output of this step of the algorithm.

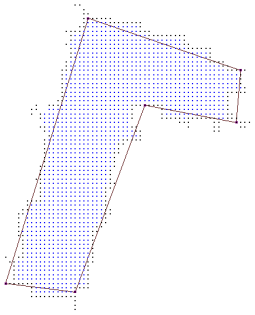


Figure 5. Approximation of the boundary of a building with a polygon obtained after step 2 of the algorithm.

4.3. Bayesian Maximum A Posteriori Estimation

At this point we have found a preliminary polygonal footprint for the building point cloud. This section describes how a Bayesian approach can improve the footprint by combining the goodness of fit to the data with a prior on footprint shapes. The goodness of fit is measured by the (planar) distance from the boundary points to the polygon, while the prior is a function of the polygon's angles that encourages straight lines, $90°$, and (to a lesser extent) $45°$ and $135°$ angles.

The Bayesian approach makes the assumption that there is a joint distribution $P(X, Z) = P(X)P(Z \mid X)$ where X represents a building footprint and Z represents the location of the boundary points. The prior $P(X)$ encodes our belief in the likelihood of various footprints while $P(Z \mid X)$ is the probability of seeing the boundary points Z given the particular building footprint X. The maximum *a posteriori* (MAP) prediction selects the footprint $\hat{X}$ from the data Z that maximizes $P(\hat{X} \mid Z)$. By Bayes Theorem, this is equivalent to selecting

$$\hat{X} = \arg\max_X P(X)P(Z \mid X) \ . \tag{1}$$

We consider distributions $P(X)$ that depend only on the k angles, $a_1, \ldots, a_k$, between the footprint's k sides[1] and can be written as $P(X) \propto \prod_{i=1}^{k} f(a_i)$. Square root priors, where $f(a)$ depends on $\sqrt{a}$ have been suggested for natural images because they have the effect of smoothing the data while enhancing edges [14]. However, we are dealing with man-made buildings, and use a function f that encourages typical building angles as illustrated in Figure 6.

The boundary points tend to be reasonably distributed around the point cloud's perimeter, so we assume that $P(Z \mid X)$ is determined solely by the distance from the boundary points to the footprint boundary. Furthermore, we assume that these distances are independent draws from a Gaussian distribution with mean 0 and variance σ^2 (negative values indicate that the boundary point lies outside the footprint). Therefore, when the distances from the n boundary points to the footprint polygon are $d_1, d_2, \ldots d_n$,

$$P(Z \mid X) = \prod_{j=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{d_j^2}{2\sigma^2}\right) \tag{2}$$

Rather than maximizing Equation 1 directly, it is equivalent (and more convenient) to find the X minimizing the negative log-likelihood, $-\log(P(X)P(Z \mid X))$. With our $P(X)$ and $P(Z \mid X)$, the MAP estimate becomes (after dropping constants which do not affect the minimization)

$$\hat{X} = \arg\min_X -\sum_{i=1}^{k} \log f(a_i) + \frac{1}{2\sigma^2} \sum_{j=1}^{n} d_j^2 \ . \tag{3}$$

The function $-\log f(a)$ is defined as a cubic spline as shown in Figure 6. The reason for using a cubic spline is that it gives us enough flexibility to represent the information that we want in our prior. Furthermore, a cubic spline prior is simple to implement and it is easy to compute the gradient at any point [6]. Figure 6 brings out the preference for specific angles iin the prior clearly. A $180°$ angle between line segments corresponds to a straight line and is highly preferred, penalty being zero. Similraly, $90°$ between line segments is also preferred with zero penalty. Angles $45°$ and $135°$ are preferred with slightly lower penalties compared to the angles nearby. Furthermore, we limit our angles from $0°$ to $180°$ by reflecting around $180°$ so that a distinction is not made between right and left turns.

This gives an optimization problem where the $1/(2\sigma^2)$ factor functions as a trade-off parameter between the importance of matching the prior (minimizing $-\log f(a_i)$) and minimizing the distances between the the boundary points and the footprint polygon. We use a gradient descent optimization method to iteratively find a local minima for minimizing X in Equation 3. In addition, during this optimization we use simulated annealing of the trade-off parameter

[1] The number of sides (and thus angles) in the footprint have been fixed by the previous step, although we do allow adjacent sides to be co-linear (having a $180°$ angle).

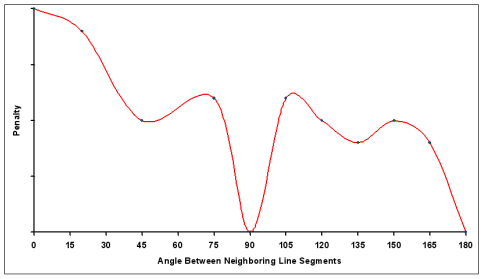


Figure 6. Prior on angles between neighboring line segments $-\log f(a)$. The preference for straight lines, 90°, 45°, and 135° bends can be seen in local minima.

to help avoid local minima. The simulated annealing starts with a small σ, so that a good fit with the boundary points is emphasized. We then gradually increase σ, increasing the emphasis on the angles. Without this annealing, the optimization tends to lock into local phenomena with "good" 45°, 135° and 90° angles before it has a chance to find a good fit for the entire building point cloud. Figure 7 shows an example of this. The final result of this optimization can be seen in Figure 8.

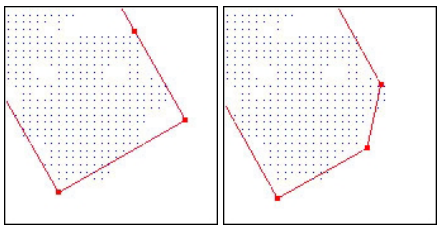


Figure 7. Left: Minimization using annealing. Right: Minimization not using annealing. The 135° local minima is visible.

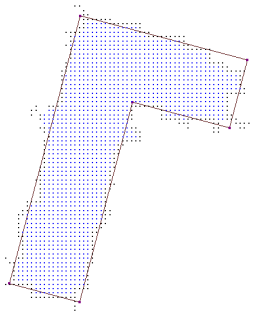


Figure 8. Building Footprint after maximum a posteriori estimation.

5. Results

We have applied our algorithm to our whole data set consisting of 380 buildings. We visually compare our results with aerial images, architectural drawings from the planning department, and ground truth in case of ambiguities.

We observe that for most buildings, the footprints that we create were extremely similar in shapes to the architectural blueprints, even in the presence of overhanging trees. Of the 380 footprints that we reconstructed, roughly 86% were good matches. In addition, in the 14% of the regions that we had poor results on, nearly all were tree regions that had been misclassified as buildings. If we consider only the initial regions that really corresponded to buildings, about 93% of the footprints we created were good matches to the architectural footprints. We find that while small details such as stairwells and awnings may be lost, the rough shape and size of the buildings are well preserved in our reconstruction. We believe that these results are very promising, particularly considering the amount of overhanging trees and boundary noise in our data set. Some of the results of our algorithm are presented in Figure 9. In most cases, our algorithm produced buildings constructed entirely out of right angles. This is an encouraging result as it reflects the preference for reconstructing right angles in our algorithm, which account for 3677 of the 4781 angles in our ground truth.

We notice two problems in the results. First, the initial ordering occasionally terminates prematurely, skipping parts of the region. This occurs almost entirely in tree regions that had been mistakenly classified as buildings during the preprocessing. Tree regions tend to have much less of a continuous structure, so tracing a path around the border of a tree region is more complicated. Because our problem does not attempt to process trees, we do not expect to be able to obtain any meaningful footprints out of these tree regions, and we can attribute these errors to errors in the classification step. In fact, it might be possible to use this information to improve the original classification results. We show some of these results in Figure 10.

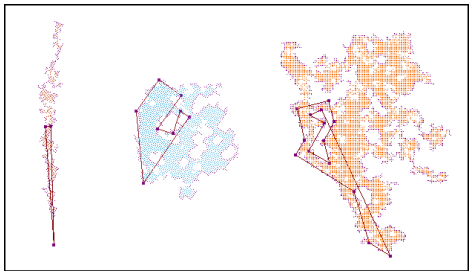


Figure 10. Tree regions reported as buildings where the initial ordering did not approximate the region shape.

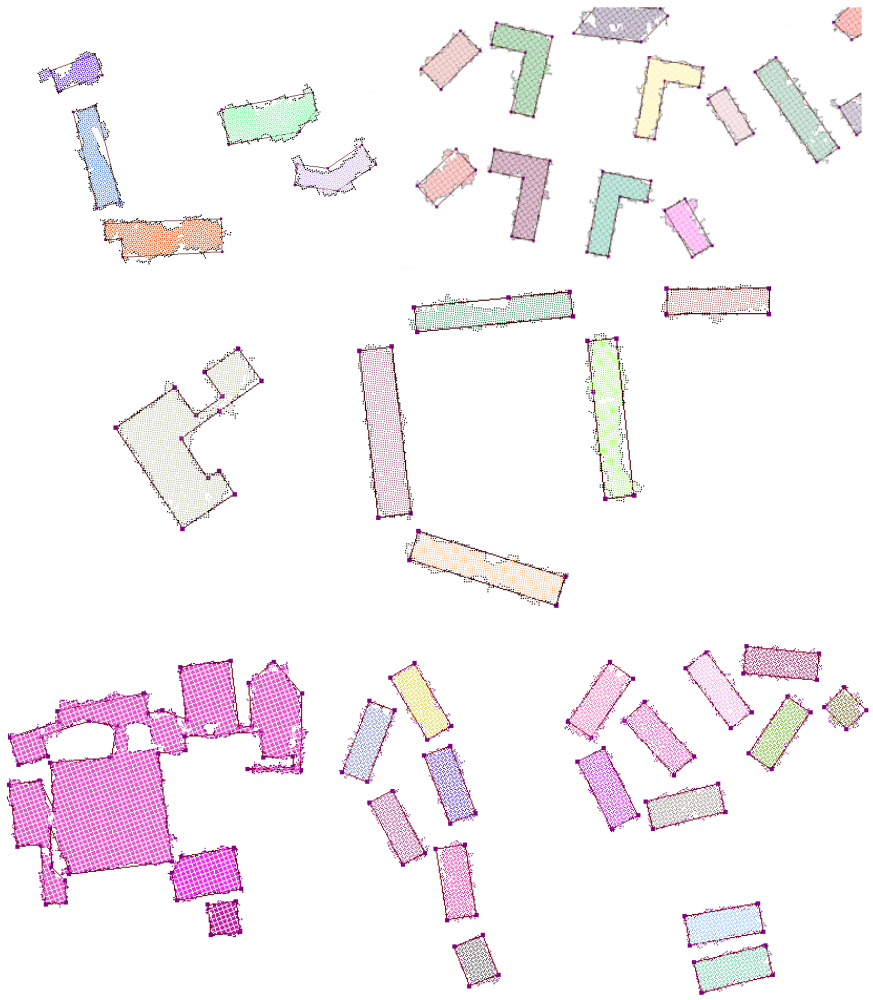


Figure 9. Buildings from various sections of our college campus with computed building footprints shown.

Second, some buildings have been approximated with an incorrect number of wall segments in section 4.2. This makes an accurate maximum a posteriori estimation more difficult. Because the optimization does not adaptively adjust the number of wall segments, it is unlikely to generate a correct building footprint after this initial number of sides has been picked. We can see some cases of this in Figure 11.

However, these two cases account for a small percentage of buildings, and our results are very close matches with the building footprints in general. Another advantage of our algorithm is that we can compute a confidence value for each building. By looking at the probability that the optimization converged on, we have a general idea how well each building footprint fit our model. We can visualize these results and confirm that the building footprints that were not able to be correctly approximated with right angles, tended to have lower probability. This statistic could be useful if we need to decide how much trust to place in the resulting footprints.

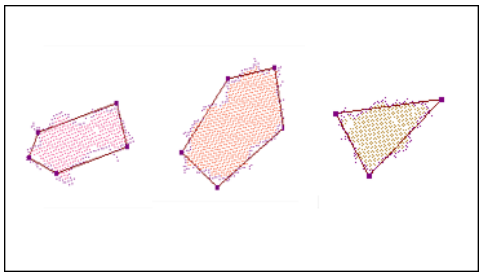


Figure 11. Rectangular buildings approximated with an incorrect number of wall segments.

6. Conclusion and Further Work

In conclusion, we present a viable option for quickly and automatically creating building footprints for large regions of buildings. Our algorithm functions well in the presence of noise, and has an easily programmable prior for an extension into other applications. This technique could be used to make building footprints more available for roof modeling or other urban planning decisions.

As many of the the problems that we encounter are associated with the initial ordering of the boundary points, further work could be conducted on determining a more robust method for this initial polygon approximation.

Because our approach uses an easily programmable prior to steer the optimization, it would be possible to apply our algorithm to other problems such as road modeling or determining likely positions of power and telecommunication lines. Furthermore, we would like to develop a method to automatically learn the prior for an application based on training data. This could extend the general idea we present into a much more robust shape extraction algorithm.

Another area of research could be using a similar Bayesian approach to the one that we present so as to compute the entire 3D model for a building rather than the footprint. A different approximation method would be required as well, as border points and neighboring angles would not be enough to define the prior for entire buildings. This could provide a useful and novel solution to the building modeling problem.

7. Acknowledgments

We would like to thank Airborne1 Corporation for helping us acquire the LIDAR data. This research is partially supported by the Multi-disciplinary Research Initiative (MURI) grant by U.S. Army Research Office under Agreement Number DAAD19-00-1-0352, the NSF grant ACI-0222900, and the NSF-REU grant supplement CCF-0222900. We also thank James David for assisting us with the refinement of some of the ideas presented in this work

References

[1] A. Alacrity and J. Bethel. Heuristic filtering and 3D feature extraction from LIDAR data. In *ISPRS Commission III, Symposium*, 2002.

[2] C. Baillard, C. Schmid, A. Zisserman, and A. Fitzgibbon. Automatic line matching and 3D reconstruction of buildings from multiple views. In *ISPRS Conference on Automatic Extraction of GIS Objects from Digital Imagery, IAPRS Vol.32, Part 3-2W5*, pages 69–80, Sep 1999.

[3] C. Brenner. Towards fully automated generation of city models. *ISPRS*, 33, 2000.

[4] A. Brunn and U. Weidner. Extracting buildings from digital surface models. *IAPRS*, 32, 1997.

[5] G. Dahl and B. Realfsen. Curve approximation and constrained shortest path problems, 1996.

[6] J. Diebel, S. Thrun, and M. Bruning. A bayesian method for probable surface reconstruction and decimation. *IEEE Transaction on Graphics*, 2005.

[7] D. H. Douglas and T. K. Peucker. Algorithms for the reduction of the number of points required to represent a line or its caricature. *The Canadian Cartographer*, 10:112–122, 1973.

[8] C. Frueh and A. Zakhor. An automated method for large-scale, ground-based city model acquisition. *International Journal of Computer Vision*, 60(1):5–24, 2004.

[9] N. Haala and C. Brenner. Generation of 3D city models from airborne laser scanning data. In *Proceedings EARSEL workshop on LIDAR remote sensing on land and sea*, pages 105–112, Tallin, Estonia, 1997.

[10] N. Haala, C. Brenner, and K. Andres. 3D urban GIS from laser altimeter and 2D map data. *ISPRS*, 32:339–346, 1998.

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

[12] T. Jones, F. Durand, and M. Desbrun. Non-iterative, feature-preserving mesh smoothing. *ACM Trans. Graph.*, 22(3):943–949, 2003.

[13] Z. Kim and R. Nevatia. Automatic description of complex buildings from multiple images. *Comput. Vis. Image Underst.*, 96(1):60–95, 2004.

[14] A. Levin, A. Zomet, and Y. Weiss. Learning to perceive transparency from the statistics of natural scenes. In *NIPS*, pages 1247–1254, 2002.

[15] H. Maas and G. Vosselman. Two algorithms for extracting building models from raw laser altimetry data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54(2-3):153–163, 1999.

[16] T. Mitchell. *Machine Learning*. McGraw-Hill, 1997.

[17] N. Molino, Z. Bao, and R. Fedkiw. A virtual node algorithm for changing mesh topology during simulation. *ACM Trans. Graph.*, 23(3):385–392, 2004.

[18] W. Ribarsky, T. Wasilewski, and N. Faust. From urban terrain models to visible cities. *IEEE Computer Graphics and Applications*, 22(4):10–15, 2002.

[19] F. Rottensteiner, J. Trinder, S. Clode, K. Kubik, and B. Lovell. Building detection by dempster-shafer fusion of LIDAR data and multispectral aerial imagery. In *Proceedings, ICPR '04*, volume 2, pages 339–342, Washington, DC, USA, 2004. IEEE Computer Society.

[20] H. Stone. Approximation of curves by line segments. *Math. Comput.*, 15:40–47, 1961.

[21] I. Suveg and G. Vosselman. Automatic 3D building reconstruction. *SPIE*, 4657-4677:59–69, 2002.

[22] R. Szeliski and D. Terzopoulos. From splines to fractals. In *Proceedings, SIGGRAPH '89*, pages 51–60, New York, NY, USA, 1989. ACM Press.

[23] G. Vosselman. Building reconstruction using planar faces in very high density heights data. *IAPRS*, 32:87–92, 1999.

[24] S. You, J. Hu, U. Neumann, and P. Fox. Urban site modeling from LIDAR. In *ICCSA (3)*, pages 579–588, 2003.