Progressive Regularization of Satellite-Based 3D Buildings for Interactive Rendering

Xiaowei Zhang
Purdue University

Gen Nishida
Purdue University

Christopher May
Purdue University

Daniel Aliaga
Purdue University

Figure 1: Examples of 3D Building Models. a-d) Our method automatically creates lightweight procedural buildings from satellite-based point clouds despite noise, occlusions, and incomplete coverage.

ABSTRACT
Automatic creation of lightweight 3D building models from satellite image data enables large and widespread 3D interactive urban rendering. Towards this goal, we present an inverse procedural modeling method to automatically create building envelopes from satellite imagery. Our key observation is that buildings exhibit regular properties. Hence, we can overcome the low-resolution, noisy, and partial building data obtained from satellite by using a two stage inverse procedural modeling technique. Our method takes in point cloud data obtained from multi-view satellite stereo processing and produces a crisp and regularized building envelope suitable for fast rendering and optional projective texture mapping. Further, our results show highly complete building models with quality superior to that of other compared-to approaches.

1 INTRODUCTION
Automatically building crisp lightweight 3D models from point-cloud data at a large scale is beneficial for many interactive applications including training and simulation and entertainment. The major inputs include 1) ground/aerial images or LIDAR data, and 2) satellite images. While satellite data is available in very large quantities, potentially worldwide, such point clouds suffer from relatively low-resolution, noise, non-standard camera/projection models, partial coverage, and occlusions. These aspects hinder creating crisp detailed building models.
One key observation is that (man-made) buildings exhibit "regular" properties such as a division into one or more floors, parallel walls, walls meeting at one of a set of predetermined angles (e.g., 90 or 135 degrees), inter and intra-floor alignments, symmetrical arrangements, straight or smoothly curved walls, and other features. We exploit this observation via an inverse procedural modeling approach to determine the procedural parameter values for a building envelope. This methodology significantly improves the resilience to partial/noisy data and produces crispier and more accurate models as compared to alternative satellite-based methods. Some results are shown in Figure 1.

Our automatic approach has two main stages (Figure 2). During a first stage, 2.5D point-clouds obtained from multi-view stereo satellite reconstruction are used to create a tree of layers and a set of line segments for each layer. During a second stage, a set of regularity constraints are pursued to arrive at parameter values producing a watertight and crisp procedural building model.

Our results yield improvements over other methods applied to the same data. Moreover, our building reconstruction produces compact and crisp models with an accuracy averaging 92% in our tested urban regions.

Our main contributions include:

- an approach to automatically create a tree of layers from satellite-based 2.5D point cloud data;
- a method to progressively enforce a set of inter- and intra-layer architectural constraints to convert a point cloud into a closed procedural building envelope; and
- a technique to find the best building model from different versions of models ranging from detailed ones to crisp ones.

2 RELATED WORK

In recent years, many researchers have been focused on building reconstruction from point clouds. Musialski et al. [2013] and Wang et al. [2018] provide reviews of urban modeling and reconstruction. In addition, with the rapid development of deep learning, many works are using deep networks to help reconstruct and render buildings from points clouds. While numerous papers address building modeling from ground/aerial/LIDAR data, very few works address full building modeling from satellite data. Even in the highest-resolution commercially available satellite imagery (i.e., WorldView3), the main structure of a building occupies on average 90x90 pixels on the ground plane. Aside from the obvious relatively low-resolution of satellite imagery, there are several other aspects that differentiate the problem of satellite multi-view stereo reconstruction from ground/aerial multi-view stereo reconstruction [Ozcanli et al. 2015; Qin 2017]. First, satellites use pushbroom-like cameras producing images different than standard frame cameras. Usually a rational polynomial coefficient (RPC) model is used. Such RPCs are hard to calibrate, require iterative processes, need many ground control points, and performing 3D to 2D as well as 2D to 3D mapping is difficult [Zheng et al. 2015]. Second, the image quality is challenged because the viewing angles of satellite sensors are greatly limited by the orbit (i.e., not very off-nadir), images of an area might be days/weeks/months apart yielding different illumination and potentially physical changes, and radiometric quality is lower despite attempts of atmospheric corrections (see Figure 3). One interesting approach is that of Facciolo et al. [2017] that won the 2016 IARPA automatic multi-view stereo challenge – their work shows the best quality possible with typical satellite images. Nonetheless, we discuss and compare to some ground/aerial methods that start with a point cloud as does our method. Those works could be roughly divided into 4 classes: Planar Primitive Fitting, Volumetric Primitive Fitting, Semantic Reconstruction and Deep Learning approaches.

2.1 Planar Primitive Fitting

These approaches usually start with extraction/detection of planar primitives (e.g., planes), and then generate the final model with a set of primitives or further optimize them so as to create compact and visually appealing 3D models. Currently, two common methods of plane detection in point clouds are Region Growing (RG) [Besl and Jain 1988; Chen 1989] and Random Sample Consensus (RANSAC) [Fischler and Bolles 1981]. Chen and Chen [2008]
describe a pipeline to reconstruct the geometry of buildings by detecting planar surfaces with a RG algorithm, and then a graph is created to represent the relationship between those planes. Finally a complete polyhedron is obtained after computing the plane connections. Li et al. [2016] extracts a large number of planes using the Efficient RANSAC algorithm [Schnabel et al. 2007] and then intersects those planes to form a set of axis-aligned candidate boxes. The final result is the subset of the boxes that have good data support and are smooth. Later, Nan and Wonka [2017] generalized the same idea to reconstruct general piecewise planar objects. Their method seeks an optimal combination of the intersected planes under manifold and watertightness constraints.

Other primitive extraction methods have also been proposed. Kelly et al. [2017] yield impressive models but their work assumes availability of 2D building footprints, polygonal meshes, and both aerial and street-level imagery. Zhou and Neumann [2010] also produce clean buildings but their LiDAR-based approach does not work well with satellite based point clouds.

Though some of these works produce crisp building models suitable for interactive rendering their approaches do not work well for the relatively noisy and low-resolution satellite data (see later comparisons in the Results section), and also they may encounter computation bottlenecks for large complex buildings and urban scenes (e.g., we used Nan and Wonka [2017] on a similar size urban environment as ours and encountered very long processing times). Solutions for such large models results in a huge number of candidate primitives and the computation may not be affordable. In contrast, our approach benefits from a simple layering strategy and we avoid the difficulty and inefficiency of finding the plane/surface intersections. The work does not include any architecturally inspired decimation or simplification to yield crisp and lightweight building models (note: in results we show a visual comparison).

### 2.4 Deep Learning

More recently, deep learning based methods have obtained excellent results for many applications in computer graphics. Several papers [Nishida et al. 2018; Sharma et al. 2017; Zeng et al. 2018] are using deep learning methods in their 3D reconstruction process. Sharma et al. [2017] build a neural network architecture, trained with many synthetic CAD models, to generate a CSG representation from an input 3D object. Their work is capable of predicting a variety of primitives but with low accuracy as the parameter extraction is done by performing classification on a discretized parameter space. In addition, the reinforcement learning steps during training require rendering a CSG model at every iteration, making the computation demanding. A small library of CSG primitives is also too limited to faithfully represent realistic scenes. In the same year, Nishida et al. [2018] proposed a pipeline to automatically generate a 3D building from a single image. Though the results are appealing, they require street-view input images. Later, Zeng et al. [2018] created a framework whose input is an aerial point cloud and their outputs are procedural models. However, their work is only focused on residential buildings for which they define specific shape grammars.

### 3 GEOMETRY SYNTHESIS

Our approach is based on a regularity assumption exploited via an energy-based optimization. The optimization seeks to alter an initial polygonal model so as to produce an output that most likely resembles the underlying structure even if partial/noisy data is given. We describe our key observation, the architectural priors, layering, and regularization.

#### 3.1 Observation

One key observation is that (man-made) buildings exhibit “regular” properties (Figure 4) such as a division into one or more floors, parallel walls, walls meeting at a set of predetermined angles (e.g., 90 or 135 degrees), co-planarity between wall segments of different building stories and between adjacent/nonadjacent wall segments within the same floor, symmetrical arrangements, straight or curved walls, and other features. Moreover, the balance of these characteristics per building varies geographically.
3.2 Architectural Priors

Given a hypothetical function $A$ maximized when the provided model is equal to ground truth, we can state our geometry synthesis goal as maximizing $A$. Instead of arbitrarily changing a building’s vertices, edges, and polygons we change them in a structured way via procedural parameters. In particular, we define $B(P_i)$ to be a parameterized procedural generation function where the parameter values $P_i$ define a building that uses a set of architectural properties to varying amounts. Thus, our goal is to maximize $A(B(P_i))$. This approach improves the resilience to noisy/partial data and produces crisp complete models.

Our approach includes the following extensible set of architectural properties (Figure 5):

- **Symmetry:** buildings might exhibit symmetry.
- **Parallelism:** walls might be parallel to each other.
- **Predetermined Corners:** corners of a building often form known angles (e.g., 90 or 135 degrees).
- **Alignment:** often walls between adjacent floors, or nearby same-floor segments, are aligned to each other.

Figure 5: Architectural Priors. Geometry synthesis enforces some/all of a) parallel walls, b) symmetry (about an axis), c) predetermined corner angles, and d) inter- and intra-layer alignment.

However, in practice even if the listed properties are present we rarely have the ground truth data for defining the function $A$. Our conjecture is that the parameter set that maximizes $A$ can also be found by maximizing

$$\alpha R(B(P_i)) + (1-\alpha)S(B(P_i)),$$

where $R$ quantifies the regularization of a generated procedural model (i.e., how well the set of architectural priors are enforced) and $S$ measures the similarity of the produced model to a mesh constructed with the incoming point cloud. Intuitively, this expression implies that by balancing regularity with similarity, we can find an optimal combination in the sense of maximizing the similarity to ground truth. Kozinski et al. [2015] arrived at a similar conclusion. Hence, we can use expression (1) as a proxy to maximizing $A$.

In order to calibrate the level of regularization, we use a small set of example buildings to calibrate constant and varying procedural parameters. In particular, given a small sample of known building structures, we perform the following minimization:

$$\min_{P_c} ||(\alpha R(B(P_k)) + (1-\alpha)S(B(P_k))) - A(B(P_c))||,$$

where $k = [1, K]$ for $K$ known buildings, $P_k = \{P_c, P_d(I_k)\}$, $P_d(I_k)$ are parameters computed from satellite image set $I_k$ for building $k$, and $P_c$ are constant for all buildings in this geographic region (or everywhere). Thus, once we have $P_c$, at runtime we can produce crisp procedural models from the satellite images despite incompleteness and noise, and without needing a priori known models.

3.3 Layering and Polyline Creation

The first phase of geometry synthesis is dividing the point data into a tree of horizontal layers. We observe that (i) buildings may consist of segments of different vertical heights and (ii) some non-adjacent segments of the building might be of the same height. To perform the layering, we place all points into a grid with voxel size equal to the satellite image pixel size of 0.3m. Then, we use a relative layering threshold to determine when the intersection-of-union (IOU) between the current layer and next horizontal single-cell voxel slice is big enough to begin a next layer. See Figure 6 for one example.

Figure 6: Layering. We show the input point cloud as a tree of layers. To the left is the point cloud and to the right is the tree of decomposed layers.
Our layering scheme produces a tree of layers, as opposed to a linear array of layers, because of the aforementioned second observation: a building might have non-adjacent segments of the same height. When creating a new layer, we solve an approximate connected component problem to determine if the new layer has multiple disjoint components (e.g., a lower part of a building then becoming two separate towers in the upper part).

We choose a representative slice for each layer and compute a single closed polyline per layer. Since a layer might consist of multiple horizontal one-cell slices through the grid, we must obtain a consensus of the layer geometry. We attempted several consensus estimation schemes and found the best one to be choosing the slice that is most similar to all other slices in the layer. Then, we use a RANSAC-based method to determine line segments with significant support (i.e., line segments that pass through, or nearly through, a sufficient number of points). The line segment determination algorithm makes use of a support threshold parameter and a closeness threshold parameter. See Figure 7 for some examples. To form the closed polyline, our method uses several heuristics. Figure 8 describes visually some of the heuristics. For example, almost collinear and close-by line segments are replaced with a best fitted single line segment and almost coincident line segment start-end points are snapped together. Other scenarios (e.g., b, c and d of Figure 8) are shown as well, collectively using various parameter values which we have determined empirically.

3.4 Regularization

In this second phase, we iteratively alter the regularization parameters (e.g., thresholds, weights, $\alpha$) so as reduce the difference between the values produced by the similarity and regularization metrics and the desired values as per equation 2 (Figure 9). This iteration continues under user control or under an external optimization loop. The resulting set of regularization parameters are later used to produce building models. The parameters can be used globally or new values can be computed for each region. In our case, we compute them once and use in all areas.

3.4.1 Metrics. The similarity and regularization metrics are calculated via a set of functions computed using each layer’s poly-lines. The similarity metric computes the IOU between the current poly-line and the original poly-line of a layer. The regularization metric makes use of several sub-metrics explained in the following paragraphs.

- **Symmetry Metric.** This function seeks an axis of reflective symmetry for the provided poly-line. For a proposed axis, the subset of the poly-line on one side of the axis is reflected over the axis. Then, if the reflected poly-line and existing poly-line are very similar (e.g., determined via IOU), it indicates a strong reflective symmetry. Our approach performs a gradient descent to compute the rotation angle and 2D intercept point that most reduces the aforementioned error sum. We prime the optimization by first evaluating the cost function with a sampling of axis rotations (e.g., one every 10 degrees) and assume all axis pass through the midpoint of the poly-line.

- **Corner Metric.** This function is evaluated for wall-to-wall corner angles of 90, 135, or 180 degrees. Given two adjacent wall segments that exceed a length threshold (e.g., 1 meter), we determine the typical corner angle to which they are most similar. If the actual angle is within a threshold of the typical corner angle (e.g., 10 degrees), we compute an error metric proportional to the angular difference (else zero).

- **Parallel Wall Metric.** This function computes the angular difference between each wall segment and the most parallel other wall segment in the layer. If the angular difference is...
we found the symmetry and curved-wall regularization to occur
wall segments parallel, the symmetry regularization will make the
buildings in terms of 2D and 3D completeness and correctness using
the least amongst these 5 metrics.

• Alignment Metric. This function has both an inter- and intra-
layer component. The inter-layer component seeks for a poly-
line segment of the layer above it and also a segment beneath it that are closest in both orientation and distance. The intra-
layer component seeks for each segment in the poly-line another segment closest in orientation and distance (but not adjacent). If the paired segments from the intra- or inter-
layer components are similar enough, they are considered candidates to be aligned. The error value is a weighted sum of the orientation difference and distance value.

• Curved Wall Metric. This function seeks to find a sequence of polyline segments that approximately form a circular arc of at least a pre-specified minimal angular span. If found, the sequence is considered a candidate to become a (circular) curved wall.

3.4.2 Calibration Process. The calibration process described at the end of Section 3.2 is performed via an additional loop placed over both the layering and regularization phases. The calibration process first decomposes a set of known models into a tree of layers. Then, a simple minimization is computed to reduce the average difference between the computed layers and the known-model layers, thus yielding best values for all the parameters described in the aforementioned layering and regularization phases. We perform this calibration once using five buildings that were manually modeled in one test area. We iteratively alter the parameters and confirm a reasonable convergence by visual inspection.

4 RESULTS
Our method is implemented using OpenCV, OpenGL, and QtUrban, and it runs on an Intel i7 workstation with a NVIDIA GTX 1080. We have applied our method to two test areas in the United States captured by WorldView3 satellite images: a portion of (A1) Jacksonville, Florida (1.9 km$^2$) and (A2) UC San Diego, California (1 km$^2$). Collectively, the areas have a few hundred buildings and medium to tall buildings have from 20 to a few hundred windows/doors each. The 2.5D point cloud dataset we use was produced by an implementation based on Rothermel et al. [2016, 2012]. Our entire method runs automatically yielding 14 buildings per minute.

4.1 Regularization
Figure 10 shows examples of the progression of models through our pipeline including applying regularization. As you can see, the corner regularization will make the corner angles into typical angles (e.g., 90), the parallel regularization will make the almost parallel wall segments parallel, the symmetry regularization will make the model hold the symmetry property, and the alignment regularization will remove misalignment between layers. In our experiments, we found the symmetry and curved-wall regularization to occur to the least amongst these 5 metrics.

4.2 Geometry Synthesis
Table 1 reports statistics about the synthesized building geometries. Table 2 contains the globally averaged accuracy of our produced buildings in terms of 2D and 3D completeness and correctness using
an implementation of the testing method of Bosch et al. [2017]. Our models are compared against a manually-refined high-resolution aerial LIDAR capture of the test area. The overall accuracy is above 90%. Further, we compare our approach to a similar set of prior methods as in the recent paper by Zeng et al. [2018]. In particular, we show in Figure 11 a visual comparison between Poisson surface reconstruction, dual contouring [Zhou and Neumann 2010], Polyfit [Nan and Wonka 2017] and our method. We also compare to a surface simplification method QSlim [Garland and Heckbert 1997] to demonstrate that general polygonal simplification does not maintain the expected geometric and architectural properties. Overall, our approach produces the best crisp and regularized models.

Leotta et al. [2019] focused on building reconstruction from satellite-based point clouds in fact, also Worldview 3 based. They developed neural networks to do semantic segmentation and then to find roof points. They extract roof shape primitives by applying RANSAC [Fischler and Bolles 1981]. The final results are refined by the boundary and the continuity of the model. Nonetheless, as you can see in Figure 12 (copied with permission from their paper), our work essentially extends such an approach to further produce crisp and lightweight building models. Although these are not the same urban areas in the figure, the quality of our solution is notably cleaner and crisper. In addition, Leotta et al. [2019] report geometric accuracy for several areas also using Bosch et al. [2017]. Their average values for the same terms as in Table 2 are 0.905, 0.73, 0.895, 0.75. As seen, our approach is consistently more accurate by 7% on average which visually amounts to a significant spread.

Finally, we show in Figure 13 many close-ups of reconstructed buildings from both areas, textured with projected satellite images. Views of our additional buildings are in supplemental figures.
Progressive Regularization of Satellite-Based 3D Buildings for Interactive Rendering

I3D ’20, May 5–7, 2020, San Francisco, CA, USA

Figure 11: Geometry Comparison. a) Incoming satellite-based point cloud, b) Poisson surface reconstruction, c) 2.5D dual contouring, d) QSlim [Garland and Heckbert 1997] of b), e) PolyFit, and f) Our method.

Table 1: Building Complexity. Average number of vertices, edges, and faces in buildings by our method.

<table>
<thead>
<tr>
<th>Zone</th>
<th>#Vertices</th>
<th>#Edges</th>
<th>#Faces</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>164.6</td>
<td>67.2</td>
<td>183</td>
</tr>
<tr>
<td>A2</td>
<td>224.24</td>
<td>90.93</td>
<td>248.5</td>
</tr>
</tbody>
</table>

Table 2: Geometric Accuracy. Accuracies for our areas in terms of the metric by Bosch et al. [2017].

<table>
<thead>
<tr>
<th>Zone</th>
<th>2D Correctness</th>
<th>2D Completeness</th>
<th>3D Correctness</th>
<th>3D Completeness</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>0.93</td>
<td>0.90</td>
<td>0.92</td>
<td>0.92</td>
</tr>
<tr>
<td>A2</td>
<td>0.95</td>
<td>0.80</td>
<td>0.92</td>
<td>0.83</td>
</tr>
</tbody>
</table>

5 CONCLUSIONS AND FUTURE WORK

We have presented a method to automatically reconstruct crisp and compact procedural buildings from satellite-based 2.5D point clouds suitable for interactive rendering (and projective texture mapping). The building envelope stage determines a tree of layers and a set of architecturally-inspired regularization metrics are used to produce a building model. The level of regularity of the model can be chosen, or calibrated, to produce a best result. The combination of regularization and procedural inference enables us to be resilient to noise, occlusions, and partial-coverage as is typical for satellite
Figure 13: Examples. We show close-ups of our buildings using projective texture mapping.

data. Our accuracy ranges from 83% to 98% and our comparisons to other approaches show the improvement our method provides.

Our approach has the following limitations.

- **Local regularization.** Our approach cannot automatically determine globally-adequate regularization parameters – instead local parameter values are used.

- **Non-regular structures.** We did not attempt to determine if a structure was particularly non-regular and as such we could adapt our parameter usage automatically.

- **Roof structures.** Our modeling process did not include roof structures and this caused lower accuracy for buildings with complex non-flat roofs (e.g., the tallest building shown at the bottom of Figure 13 does not have a correct roof structure).

- **2.5D assumption.** At present we assume only 2.5D building structures and this could be extended to full 3D buildings assuming the point cloud data becomes fully 3D.

Our approach has several avenues of future work. First, we could develop a scheme for geographically-dependent regularization when scaling to other regions. Second, our system could be extended to include rooftops, which it currently does not handle.

ACKNOWLEDGMENTS

This research was supported in part by the Intelligence Advanced Research Projects Activity (IARPA) via Department of Interior/Interior Business Center (DOI/IBC) contract number D17PC00280. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. Disclaimer: The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of IARPA, DOI/IBC, or the U.S. Government. Additional support came from National Science Foundation grants #10001387 and #1835739.

REFERENCES


