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

Xiaowei Zhang
Purdue University

Christopher May
Purdue University

Gen Nishida
Purdue University

Daniel Aliaga
Purdue University

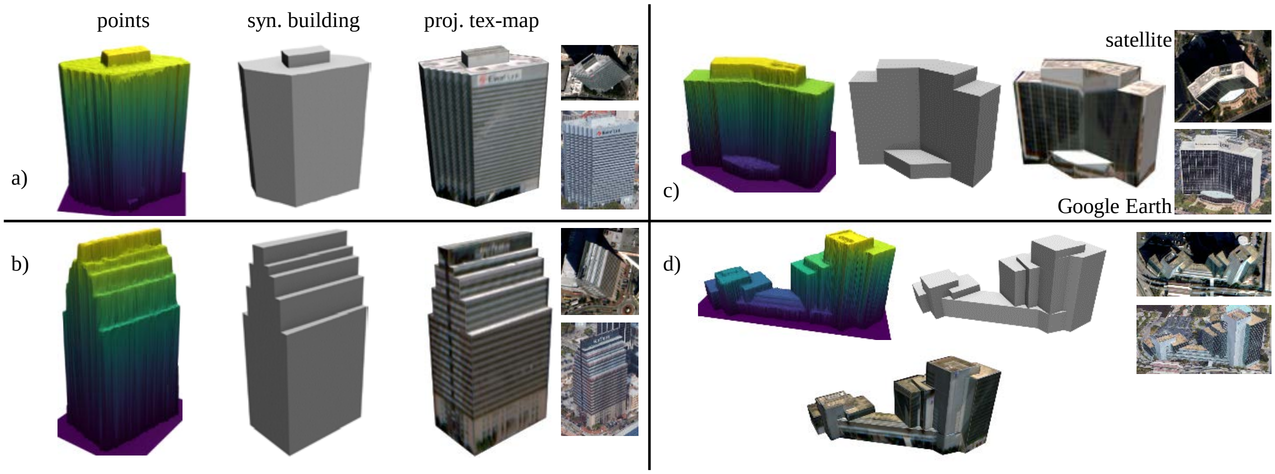

Figure 1: *Examples of 3D Building Models.* a-d) Our method automatically creates lightweight procedural buildings from satellite-based point clouds despite noise, occlusions, and incomplete coverage.

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.

CCS CONCEPTS

• **Computing methodologies → Image-based rendering**; **Mesh geometry models**; **Shape analysis**.

KEYWORDS

urban modeling, level-of-detail, regularization, satellite imagery, 3D modeling

ACM Reference Format:
Xiaowei Zhang, Christopher May, Gen Nishida, and Daniel Aliaga. 2020. Progressive Regularization of Satellite-Based 3D Buildings for Interactive Rendering. In *Symposium on Interactive 3D Graphics and Games (I3D '20), May 5–7, 2020, San Francisco, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3384382.3384526

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.

I3D '20, May 5–7, 2020, San Francisco, CA, USA
© 2020 Association for Computing Machinery.
ACM ISBN 978-1-4503-7589-4/20/05…$15.00
https://doi.org/10.1145/3384382.3384526

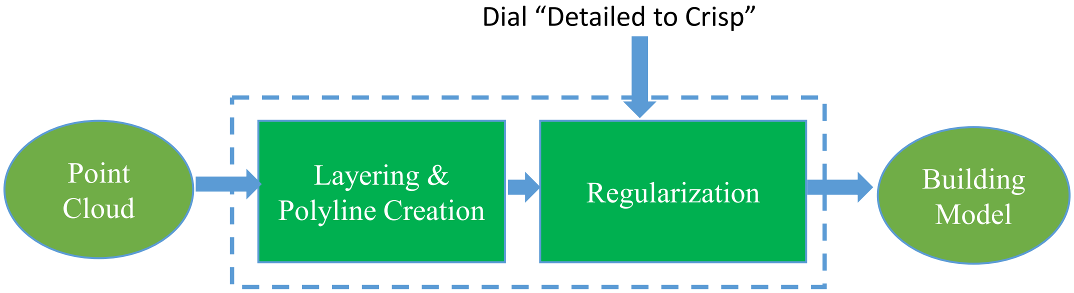

Figure 2: *Pipeline.* The pipeline of our geometry synthesis method.

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 crisper 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.

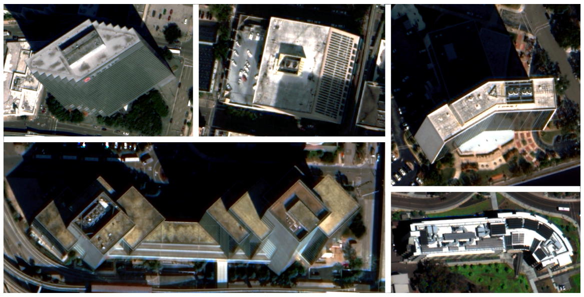

Figure 3: *Satellite Images.* Example satellite images containing shadows, noise, reflections, occlusions, and vegetation.

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 primitives. Instead, we look for simpler line segments or curved primitives in 2D layer images. In our pipeline, we also make use of high-level architectural regularization terms to improve the quality of our models.

2.2 Volumetric Primitive Fitting

These approaches use a customized 3D model library which includes cube, sphere, cylinder, and other basic solids. The 3D building reconstruction is done by directly fitting the primitives. Nan et al. [2010] presented an interactive tool called Smart-Boxes to reconstruct building structures directly from point clouds using cuboid primitives. This method achieves appealing results with significant user interaction. Xiao and Furukawa [2012] proposed a reconstruction method for indoor environments based on constructive solid geometry (CSG). They first split the 3D space into a set of horizontal slices, each of which shares the same horizontal structure, and each horizontal slice is used to find 2D rectangular primitives. In the end, they extend the 2D primitives into 3D models. Usually these methods suffer from the limitation of using a fixed primitive library and when it comes to the noisy, incomplete satellite point cloud, it's also difficult to guarantee the accuracy of reconstruction.

2.3 Semantic Reconstruction

These methods start with segmenting the point cloud into meaningful labels (e.g., ground, building, roof, facade, etc..). and then applying the aforementioned primitive fitting methods in order to perform the reconstruction. Nguatem and Mayer [2017] segment point clouds into meaningful structures, such as the ground, facades, roofs and roof superstructures, and then use polygon sweeping to fit predefined templates for buildings, They produce compelling results but only have a fixed set of template buildings. To obtain a level-of-detail representation of urban scenes, Verdie et al. [2015] extract a large set of plane candidates after classifying the point clouds, and then a surface model is extracted from a set of 3D arrangements based on a min-cut formulation. Leotta et al. [2019] apply a semantic segmentation and extract roof points from the point cloud. Then they iteratively apply RANSAC [Fischler and Bolles 1981] to detect one or more core roof shapes. These roof shapes are then extended to the ground plane. 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 one of 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.

Figure 4: *Observation.* Man-made buildings exhibiting regular properties.

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.

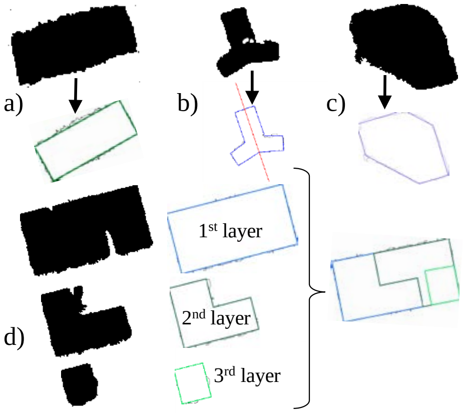

Figure 5: *Architectural Priors.* Geometry synthesis enforces some/all of a) parallel walls, b) symmetry (about an axis), c) predetermined corner angles, and d) inter- and intra-layer alignment.

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

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

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:

$$min_{P_c} ||(\alpha R(B(P_k)) + (1-\alpha)S(B(P_k))) - A(B(P_k))||, \tag{2}$$

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.

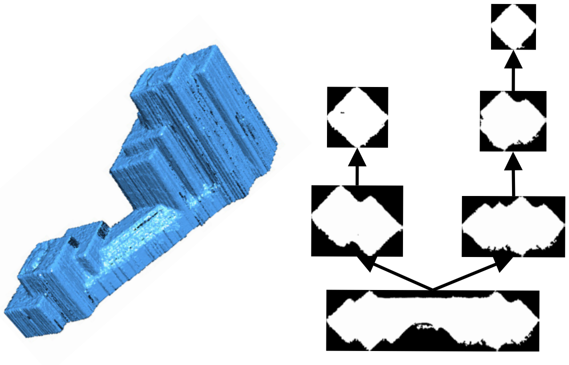

Figure 6: *Layering.* We show the input point cloud as a tree of layers. To the left is the point cloud and to the right is the tree of decomposed layers.

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.

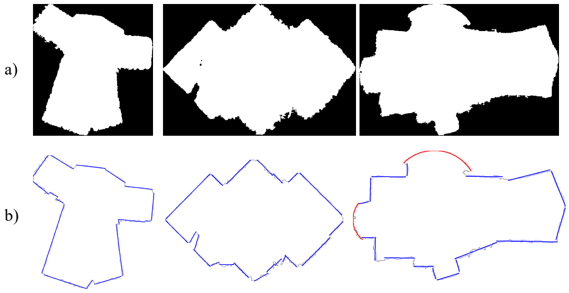

Figure 7: *RANSAC.* Layer examples of using a RANSAC-based method to detect line segments. a) Layer inputs, b) Line segments.

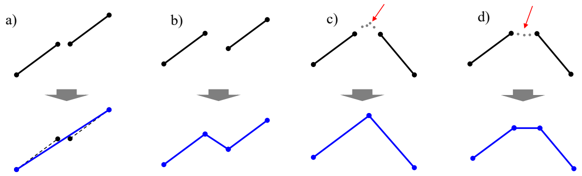

Figure 8: *Heuristics.* Heuristics for Connecting Line Segments to Define Each Layer. a) Almost collinear and close enough, b) Almost collinear, c) Some supporting points near the intersection, and d) No supporting points near the intersection.

3.4 Regularization

In this second phase, we iteratively alter the regularization parameters (e.g., thresholds, weights, α) 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.

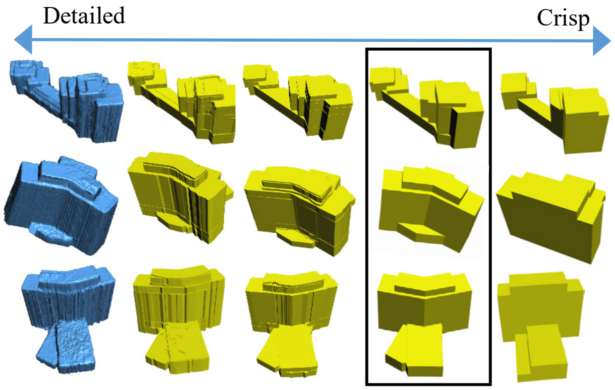

Figure 9: *Regularization. We alter regularization parameters until producing an output of desired detail/crispness.*

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 polyline 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

smaller than a threshold (e.g,. 3 degrees), the error is the angular difference (else zero).

- *Alignment Metric.* This function has both an inter- and intra-layer component. The inter-layer component seeks for a polyline 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 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 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.

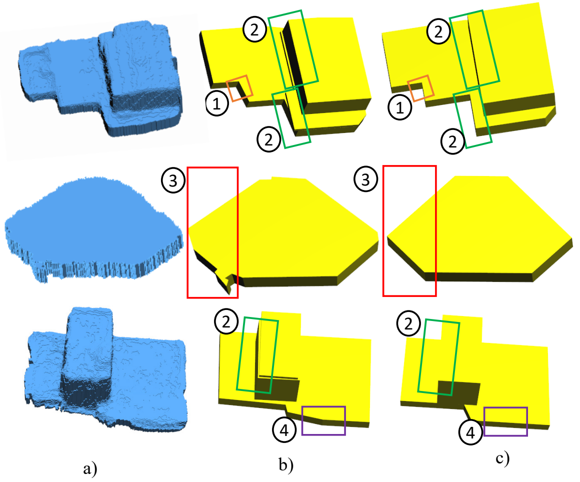

Figure 10: *Metrics. We show examples of the metrics used in our regularization. a) Incoming satellite-based point cloud, b) the output models before regularization, and c) the output models after applying specific regularization terms. "1" is corresponding to the corner regularization. "2" is corresponding to the alignment regularization. "3" is corresponding to the symmetry regularization. "4" is corresponding to the parallel regularization.*

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.

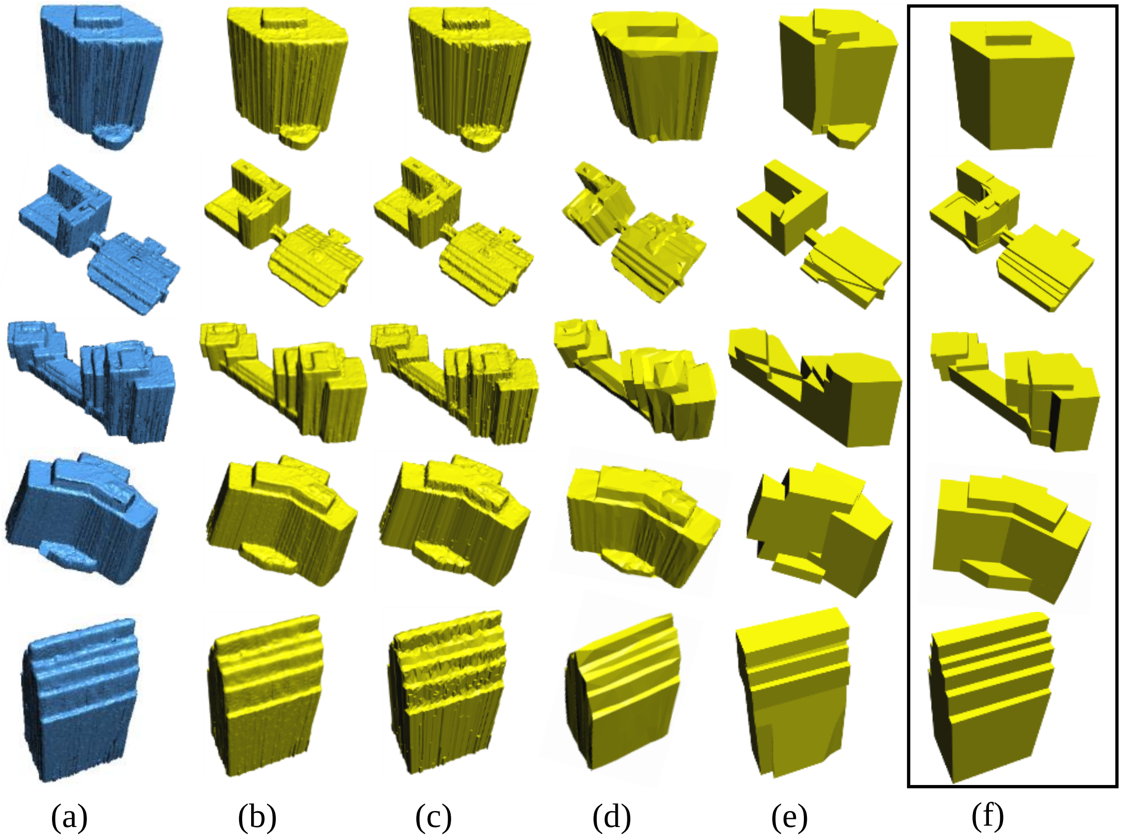

Figure 11: *Geometry Comparison.* a) Incoming satellite-based point cloud, b) Poisson surface reconstruction, c) 2.5D dual contouring, d) QSlim [Garland and Heckbert 1997] of b), e) PolyFit, and f) Our method.

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

Zone	#Vertices	#Edges	#Faces
A1	164.6	67.2	183
A2	224.24	90.93	248.5

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

Zone	2D Correctness	2D Completeness	3D Correctness	3D Completeness
A1	0.93	0.90	0.92	0.92
A2	0.95	0.80	0.92	0.83

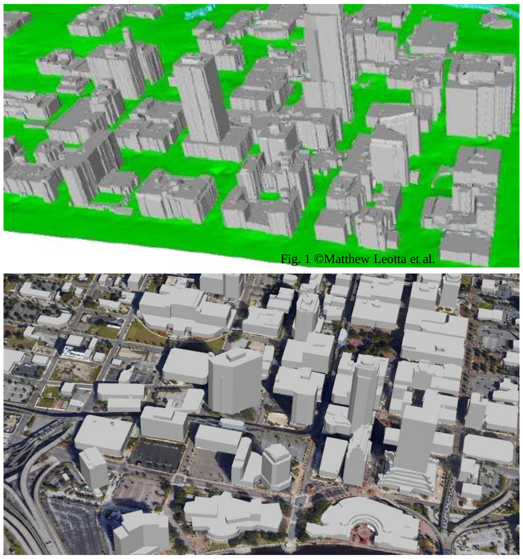

Figure 12: *Comparison.* The top row is a result image cropped from Leotta et al. [2019]. The bottom row is generated by our system.

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

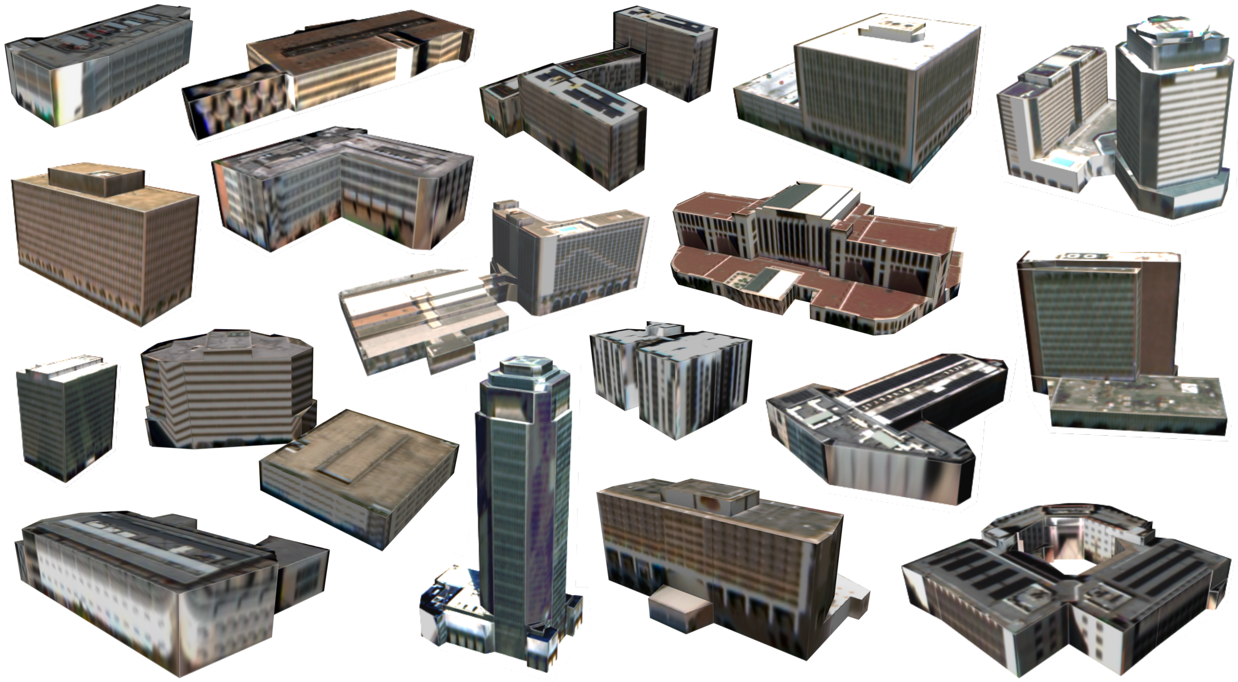

Figure 13: *Examples.* We show close-ups of our buildings using projective texture mapping.

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

P. J. Besl and R. C. Jain. 1988. Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10, 2 (March 1988), 167–192. https://doi.org/10.1109/34.3881

M. Bosch, A. Leichtman, D. Chilcott, H. Goldberg, and M. Brown. 2017. Metric Evaluation Pipeline for 3d Modeling of Urban Scenes. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences* (May 2017), 239–246.

D. S. Chen. 1989. A data-driven intermediate level feature extraction algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11, 7 (July 1989), 749–758. https://doi.org/10.1109/34.192470

Jie Chen and Baoquan Chen. 2008. Architectural Modeling from Sparsely Scanned Range Data. *International Journal of Computer Vision* 78 (07 2008), 223–236. https://doi.org/10.1007/s11263-007-0105-5

Gabriele Facciolo, Carlo De Franchis, and Enric Meinhardt-Llopis. 2017. Automatic 3D reconstruction from multi-date satellite images. In *IEEE Computer Vision and Pattern Recognition Workshops*. 57–66.

Martin A. Fischler and Robert C. Bolles. 1981. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. *Commun. ACM* 24, 6 (June 1981), 381–395. https://doi.org/10.1145/358669.358692

Michael Garland and Paul S. Heckbert. 1997. Surface Simplification Using Quadric Error Metrics. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 209–216. https://doi.org/10.1145/258734.258849

Tom Kelly, John Femiani, Peter Wonka, and Niloy J. Mitra. 2017. BigSUR: Large-scale Structured Urban Reconstruction. *ACM Transactions on Graphics* 36, 6 (Nov. 2017).

Mateusz Kozinski, Raghudeep Gadde, Sergey Zagoruyko, Guillaume Obozinski, and Renaud Marlet. 2015. A MRF shape prior for facade parsing with occlusions. In *IEEE Computer Vision and Pattern Recognition*. 2820–2828.

Matthew J. Leotta, Chengjiang Long, Bastien Jacquet, Matthieu Zins, Dan Lipsa, Jie Shan, Bo Xu, Zhixin Li, Xu Zhang, Shih-Fu Chang, Matthew Purri, Jia Xue, and Kristin Dana. 2019. Urban Semantic 3D Reconstruction From Multiview Satellite Imagery. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*.

Minglei Li, Peter Wonka, and Liangliang Nan. 2016. Manhattan-World Urban Reconstruction from Point Clouds, Vol. 9908. 54–69. https://doi.org/10.1007/978-3-319-46493-0_4

Przemyslaw Musialski, Peter Wonka, Daniel G Aliaga, Michael Wimmer, Luc Van Gool, and Werner Purgathofer. 2013. A survey of urban reconstruction. In *Computer Graphics Forum*, Vol. 32.

Liangliang Nan, Andrei Sharf, Hao Zhang, Daniel Cohen-Or, and Baoquan Chen. 2010. SmartBoxes for Interactive Urban Reconstruction. *ACM Trans. Graph.* 29, 4, Article 93 (July 2010), 10 pages. https://doi.org/10.1145/1778765.1778830

Liangliang Nan and Peter Wonka. 2017. PolyFit: Polygonal Surface Reconstruction from Point Clouds. (2017).

William Nguatem and Helmut Mayer. 2017. Modeling urban scenes from pointclouds. In *IEEE International Conference on Computer Vision*. 3837–3846.

Gen Nishida, Adrien Bousseau, and Daniel G. Aliaga. 2018. Procedural Modeling of a Building from a Single Image. *Computer Graphics Forum (Eurographics)* 37, 2 (2018).

Ozge C Ozcanli, Yi Dong, Joseph L Mundy, Helen Webb, Riad Hammoud, and Victor Tom. 2015. A comparison of stereo and multiview 3-D reconstruction using cross-sensor satellite imagery. In *IEEE Computer Vision and Pattern Recognition Workshops*. 17–25.

Rongjun Qin. 2017. Automated 3d recovery from very high resolution multi-view satellite images. In *ASPRS (IGTF) annual Conference*. 10.

M Rothermel, N Haala, and D Fritsch. 2016. A MEDIAN-BASED DEPTHMAP FUSION STRATEGY FOR THE GENERATION OF ORIENTED POINTS. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences* 3, 3 (2016).

Mathias Rothermel, Konrad Wenzel, Dieter Fritsch, and Norbert Haala. 2012. SURE: Photogrammetric surface reconstruction from imagery. In *Proceedings LC3D Workshop, Berlin*, Vol. 8. 2.

R. Schnabel, R. Wahl, and R. Klein. 2007. Efficient RANSAC for Point-Cloud Shape Detection. *Computer Graphics Forum* (2007). https://doi.org/10.1111/j.1467-8659.2007.01016.x

Gopal Sharma, Rishabh Goyal, Difan Liu, Evangelos Kalogerakis, and Subhransu Maji. 2017. CSGNet: Neural Shape Parser for Constructive Solid Geometry. *CoRR* abs/1712.08290 (2017). arXiv:1712.08290 http://arxiv.org/abs/1712.08290

Yannick Verdie, Florent Lafarge, and Pierre Alliez. 2015. LOD Generation for Urban Scenes. *ACM Transactions on Graphics* 34, 3 (2015).

R. Wang, J. Peethambaran, and D. Chen. 2018. LiDAR Point Clouds to 3-D Urban Models: A Review. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 11, 2 (Feb 2018), 606–627. https://doi.org/10.1109/JSTARS.2017.2781132

Jianxiong Xiao and Yasutaka Furukawa. 2012. Reconstructing the World's Museums. In *ECCV*.

Huayi Zeng, Jiaye Wu, and Yasutaka Furukawa. 2018. Neural Procedural Reconstruction for Residential Buildings. In *The European Conference on Computer Vision (ECCV)*.

Enliang Zheng, Ke Wang, Enrique Dunn, and Jan-Michael Frahm. 2015. Minimal solvers for 3d geometry from satellite imagery. In *IEEE International Conference on Computer Vision*. 738–746.

Qian-Yi Zhou and Ulrich Neumann. 2010. 2.5D Dual Contouring: A Robust Approach to Creating Building Models from Aerial LiDAR Point Clouds. In *European Conference on Computer Vision*.