2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
Women in Computer Vision (WiCV) Workshop
Las Vegas, Nevada, USA, 26 June - 1 July, 2016

Procedural Editing of 3D Building Point Clouds (ICCV 2015)

İlke Demir
Purdue University
idemir@purdue.edu

Daniel G. Aliaga
Purdue University
aliaga@purdue.edu

Bedrich Benes
Purdue University
bbenes@purdue.edu

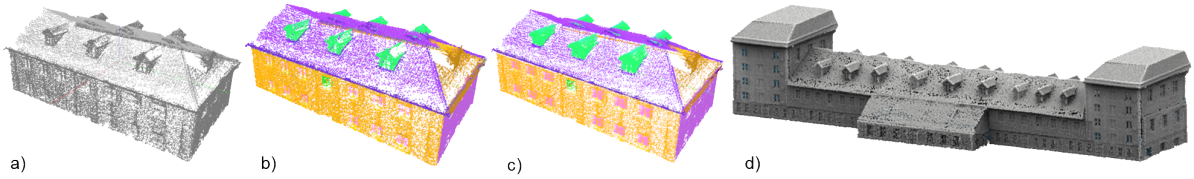

Figure 1: ***Pipeline.*** An input point cloud (a) is semi-automatically segmented (b), the segments are used for completion with consensus models (c), and placed in a tree. Then, the tree is parsed and rules are discovered to enable procedural editing (d).

1. Introduction

Thanks to the recent advances in computational photography and remote sensing, point clouds of buildings are becoming increasingly available, yet their processing poses various challenges. In our work, we tackle the problem of point cloud completion and editing and we approach it via inverse procedural modeling. Our approach consists of 1) semi-automatic segmentation of the input point cloud with segment comparison and template matching to detect repeating structures, 2) a consensus-based voting schema and a pattern extraction algorithm to discover completed terminal geometry, all encoded into a context-free grammar, and 3) an interactive editing tool where the user can create new point clouds by using procedural copy and paste operations, and smart resizing. We demonstrate our approach on editing of building models with up to 1.8M points.

2. Previous Work

Recently, there has been a significant increase in the availability of 3D point cloud datasets of buildings and other urban structures. This data representation often lacks high-level grouping or segmentation information. This lack hinders (i) directly and intuitively editing such point cloud building models, (ii) creating novel polygonal/point-based models preserving style, and (iii) obtaining complete building models. Some methods have centered on segmenting the point cloud into components (e.g., [5]). Others provided 3D polygonal urban reconstructions from point data (e.g., [3]), explored point editing tools (e.g., [1, 4]), or addressed model completion.

In contrast, our key motivation is to directly inspect the point cloud in order to segment and organize repeating structures into a hierarchical representation and then to enable directly performing a set of "what if" urban design and planning operations. Our method discovers (and organizes) repetition improving both completeness and quality of sampled structures. Further, the improvements are carried forward during editing, hence improving subsequent surface triangulations of novel structures as well as preserving the original structure.

3. Segmentation

The first stage of our approach is to segment the model by discovering dominant planes, extracting distance-based clusters, and finding additional repetition within each plane or cluster. Moreover, the user may interactively select a region of interest and generate additional segmentation.

Given a region of interest, we use a multi-pass partition algorithm to automatically segment the model into a set of planes and (non-planar) clusters. During each planar segmentation pass, the most dominant planes are found using a RANSAC-based plane-fitting method based on PCL. When the planes are thrown out, the remaining non-planar structures are mostly disjoint groups which can be clustered by using Euclidean distance-based clustering. We further decompose a planar or an Euclidian segment into a connected set of repeating point clouds.

4. Proceduralization

The second preprocessing stage of our algorithm converts the segments into a procedural representation. All segments with the same labels create a single representa-

tive *consensus model* (CM). The CM has the advantage of producing a result that is complete in the sense of joining multiple partial samples of the same underlying geometry. Figure 2 shows an example of a building before and after using consensus models. We refer to our paper for the details of CM computation.

After a CM is found, the segments are used in a top-down construction process of a tree containing all segments and their repetitions (Figure 3). The subtrees are checked for patterns and their applications. The spacing between each instance of a segment within a repeated application is found by an analysis of the minimum repeating distance and the frequency of application.

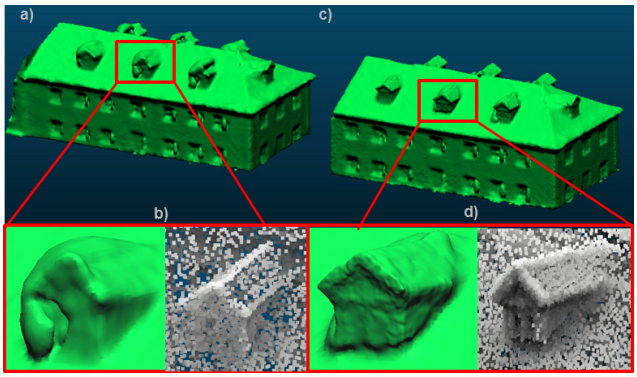

Figure 2: **Consensus Model.** Segments are ICPed for CM. Poisson reconstruction is improved original (a,b) to (c,d).

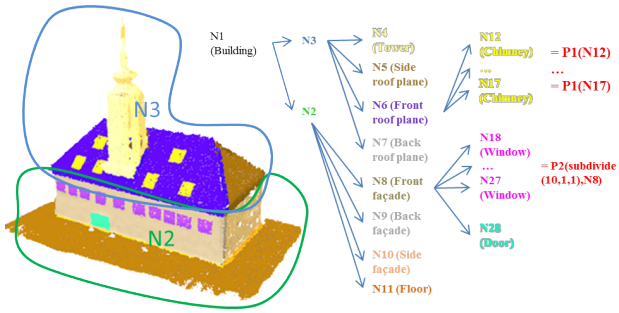

Figure 3: **Tree and patterns.** Building segments (left) are put into the tree (right). Nodes can be invisible (N1, N2, N3) or visible (colored). If a regular spacing of a pattern is found (e.g., the pink windows), the subdivide directive applies (P2). Otherwise it is expressed as a different application (e.g., roof windows, P1).

5. Editing

At runtime, our interactive tool allows directly altering the generated split grammar and supports interactive editing, based on the parse tree constructed in the previous section. Editing begins by selecting a region of interest; initial attachments are automatically computed and represented as a large sparse linear system of equations. There are six types of attachment constraints computed based on the adjacency graph of the nodes for a resize operation, and are solved using sparse linear least squares. Further, we explicitly address resampling at the boundary between two nodes just placed next to each other. We define a thin neighborhood region enclosing the seam. Then, points within the seam are re-sampled using EAR [2] so as to match the density of the surrounding region. This tool also enables copying and then inserting or replacing an existing rule or terminal of the model.

6. Results and Discussion

We present results and comparisons of our method applied to point clouds. Our software is written in C/C++ using Qt, OpenCV, Point Cloud Library (PCL), and EAR [2]. We show various completion, robustness, editing, comparison, and user interaction examples and evaluation in our paper. We have used our approach to edit building models with up to 1.8M points requiring a few minutes preprocessing time and supporting interactive editing.

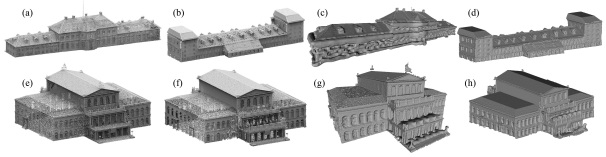

Figure 4: **Reconstruction**. Original points (a,e) reconstructed with Poisson (c,g), and edited points (b,f) reconstructed with Poisson (d,h).

References

[1] M. Arikan, M. Schwärzler, S. Flöry, M. Wimmer, and S. Maierhofer. O-snap: Optimization-based snapping for modeling architecture. *ACM Trans. Graph.*, 32:6:1–6:15, 2013. 1

[2] H. Huang, S. Wu, M. Gong, D. Cohen-Or, U. Ascher, and H. R. Zhang. Edge-aware point set resampling. *ACM Trans. Graph.*, 32(1):9:1–9:12, Feb. 2013. 2

[3] H. Lin, J. Gao, Y. Zhou, G. Lu, M. Ye, C. Zhang, L. Liu, and R. Yang. Semantic decomposition and reconstruction of residential scenes from lidar data. *ACM Trans. Graph.,*, 32(4), 2013. 1

[4] L. Nan, A. Sharf, H. Zhang, D. Cohen-Or, and B. Chen. Smartboxes for interactive urban reconstruction. *ACM Trans. Graph.,*, 29(4):Article 93, 2010. 1

[5] A. Toshev, P. Mordohai, and B. Taskar. Detecting and parsing architecture at city scale from range data. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 398–405, June 2010. 1