A Light-weight Approach for Automatically Reconstructing Large-scale Trees

Wenmeng Zhou, Yao Yu, Yu Zhou and Sidan Du
Department of Electronic Science and Technology
Nanjing University
Nanjing, China
wenmengeeic@gmail.com, {nackzhou, allanyu, coff128}@nju.edu.cn

***Abstract*—Different from previous tree modeling approaches, our method is based on the idea of making tree reconstruction as quick as possible and simplifying the representation of final results while keeping the tree model visually acceptable. Each tree is represented by Billboard model. We first get the shape mask of a tree by projecting LiDAR point cloud onto 2D camera plane. Then we use the shape fitting method to obtain the corresponding rotation axes and bounding boxes for the main trunk and tree crown. We get the corresponding texture and correct the misalignment artifacts by texture completion. Finally, we rotate each textured polygon around the rotation axis to a certain degree. We demonstrate the effectiveness of our system with some LiDAR data sets and compare our tree modeling scheme with other state-of-the-art reconstruction algorithms to show its advantages in terms of speed and memory footprint.**

Keywords-LiDAR point cloud; Tree modeling; Large-scale reconstruction; Texture completion

I. INTRODUCTION

Trees are ubiquitous in the real world but are very complex in geometry and various in structure from each other, which makes it difficult to model them in a realistic way. Different methods of modeling trees have been proposed and great progress has been made because trees play an important role in enriching the realism of virtual environments. These methods can be mainly divided into two categories: One is procedural modeling. By forming a set of grammar like L-system[10], different shapes of trees can be generated even those with extreme complexity. However, this method needs a lot of parameters[18], the computational cost is very high and it is difficult to conform the resulting geometry shape exactly to a specified one[1, 14, 15]. Another one is reconstructing 3D model directly from real world data such as images[9, 12, 16, 17], or laser-scanned point clouds[6, 20]. The problem is that due to the large variance in trees, fully automatic algorithm for reconstructing trees is very difficult and computationally expensive.

In this paper, we present a light-weight method to automatically reconstruct large-scale trees in a short time, while the resulting model is visually acceptable as shown in Figure 1. This method is motivated by trying to find a method to automatically reconstruct a large amount of trees from laser-scanned point clouds and images. We adopt Billboard model, which just uses a few polygons. These polygons are rotated around the symmetric axis of a tree to a certain angle and are pasted with transparent texture so that this model will look like 3D model rather than 2D planes.

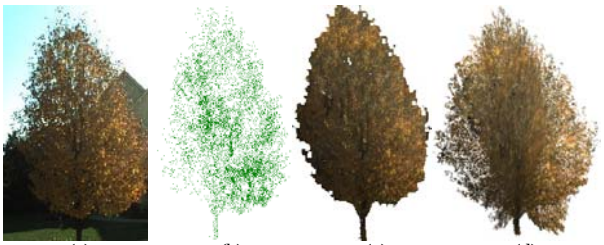


Figure 1. Reconstruction of laser-scanned trees using our method: a) photograph; b) point cloud; c) An image of texture after being completed; d) final result of reconstruction.

The texture mapped to the rotated polygons is generated automatically from images captured by Ladybug camera. Our proposed algorithm of texture completion is able to extract the correct texture of trees from images with misalignment artifacts and replace the wrong parts with the right ones, as illustrated in Figure 1(c).

The main contribution of this paper is to address an automatic and light-weight method to reconstruct trees in large scale, especially in shape fitting and texture completion. This method is evaluated by reconstructing a large number of trees from point clouds acquired by terrestrial LiDAR (Light Detection And Ranging) device. The algorithm is efficient in terms of speed and memory footprint and the results appear vivid.

II. RELATED WORK

Traditionally, rule based methods are used to design virtual trees[5]. De Reffye et al. [3] designed rules according to botanical knowledge. Weber et al. [18] use rules related to the geometrical characteristic of trees. Though these methods can create vivid tree models, they need a lot of parameters and sometimes professional knowledge. In order to increase the realism of 3D model of trees, considerable research attention has been paid to reconstruction of trees from real world data, e.g. images or laser-scanned point clouds.

Images based methods have been widely used in the past years. Reche-Martinez et al. [12] use a set of carefully registered photographs to generate a volumetric representation of the tree foliage and its branches. Neubert et al. [9] adopt particle flows to estimate an approximate voxel-based model. Quan et al. [11] and Tan et al. [16] use structure from motion to generate 3D point cloud from a series of images. Then point cloud is used to generate the triangle mesh model of branches and leaves. All the methods above need multiple carefully registered images as input and trees should be segmented out accurately from background. Instead of relying on multiple images, Tan et al. [17] create an approximate but simple branching structure from a single image with user sketches. But it needs user interaction and can not be performed automatically.

As laser scanning technology is developing very fast, approaches that reconstruct trees from laser-scanned point cloud are becoming more and more popular. Xu et al. [19] present a semi-automatic method to produce the skeleton of trees by clustering points in a spanning graph and shortest path algorithm. Then smaller branches and leaves are synthesized to form the tree crown. Bucksch et al. [2] reconstruct the geometry of tree by clustering points using space partitioning, followed by connecting adjacent clusters. The two methods above use pre-defined resolution for clustering or partitioning, which are unreliable when trees have different densities in different parts. Runions et al. [13] use a space colonization algorithm to grow skeleton within an envelope under the constraints of attraction points which are controlled by users. Leaves are then randomly added. Zhu et al. [21] propose a novel approach for tree crown reconstruction based on an improvement of alpha shape modeling. Livny et al. [6, 7] automatically reconstruct the branching structure of trees using global optimizations and propose a lobe-based representation for tree modeling. The foliage part of trees is abstracted and represented by lobe-geometry, which captures the main characteristics of tree crown while the model representation takes few memory.

Our method takes both laser-scanned point clouds and images as input. Different from those methods that reconstruct branches of trees and then add leaves, we use a few polygons with transparent texture to model trees, which takes low computational cost. Besides, texture can be generated from images even those with misalignment artifacts. The specific description of our model is introduced in Section IV.

III. Overview

The pipeline of reconstruction of trees from a large-scale data set is illustrated by Figure 2: The input consists of the laser-scanned point clouds and the image sequences captured by cameras. Semantic segmentation is used to extract point cloud of each single tree. The shape fitting and texture completion process are the core steps of our method, which determine the quality of reconstruction.

The remaining part of this paper is organized as follows. Section IV introduces the Billboard model we use to reconstruct trees. In section V we discuss preprocessing steps. In section VI, we describe the shape fitting algorithm, followed by the algorithm of texture completion. We demonstrate experimental results in Section VII and give discussions and conclusions about our method in Section VIII.

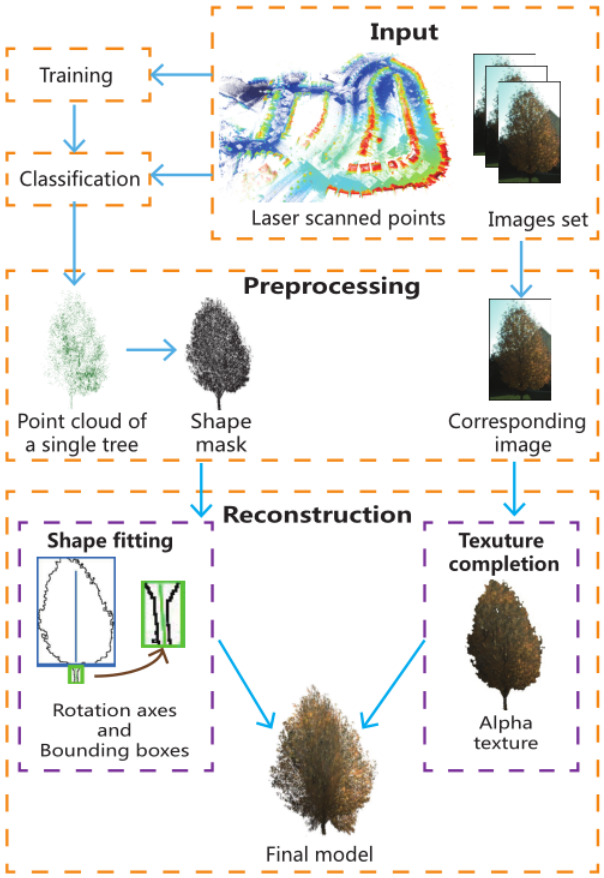


Figure 2. Pipeline of the reconstruction process: the point cloud of the whole scene is semantically segmented into different categories, such as houses or trees. The point cloud of each tree are then separated from each other. Shape mask is calculated and the corresponding texture image is selected from set of images in the process of preprocessing. Afterwards two rotation axes and the respective bounding boxes are got from shape mask and the texture is extracted and completed through texture completion algorithm. The final tree model consists of six polygons rotated around the two rotation axes and patched with transparent texture.

IV. Model Representation

We adopt Billboard model to reconstruct trees as illustrated in Figure 3. Trees are divided into two parts: the main trunk and tree crown. Because trees grow approximately in a direction that is vertical to the ground, the tree crown is rotated around a vertical axis. But the trunk part may be slant, so we need to treat it alone. If the proper rotation axis is found, simply by rotating the polygons around the symmetry axis to certain degrees so

that each polygon is uniformly distributed at 360 degrees, the resulting trees will look like 3D models.

As a result, we use six polygons to represent a tree, three for the main trunk and the rest for the tree crown. Proper rotation axes and bounding boxes are found using shape fitting. Then the six polygons are rotated around the axes and patched with alpha texture generated by texture completion. One resulting model is shown in Figure 3.

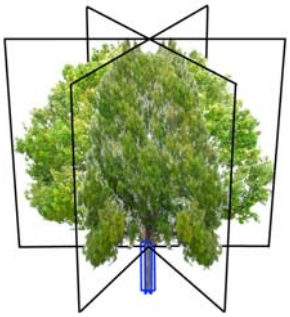


Figure 3. Billboard model is used to represent trees.

V. DATA PREPROCESSING

The data is acquired by a car equipped with a LiDAR, GPS/IMU system and ladybug camera, including laser-scanned point clouds and images. In order to carry on the progress of automatically reconstruct trees in large scale, we need to extract the point cloud and images for every single tree.

We adopt semantic segmentation algorithm proposed by Zhou et al. [20] to extract point cloud of each single tree. The corresponding image of a certain tree is selected using the following equation:

$$\arg\min_{i} \| \mathbf{x_i} - \mathbf{x_{center}} \|^2 \qquad (1)$$

where i is image number, x_i is the camera position and x_{center} is the center position of a tree. In other word, the corresponding image is the nearest one to the tree.

We calibrate the camera and calculate the relative rotation R_{lc} and translation T_{lc} between the coordinate systems of LiDAR and camera, thus getting the intrinsic matrix K and the extrinsic matrix $M_{lc} = [R_{lc} \mid -R_{lc}\, T_{lc}]$ in the local coordinate system of LiDAR. With GPS/IMU, the transformation between world and LiDAR coordinate systems is continuously got: $M_{wl} = [R_{wl} \mid -R_{wl}\, T_{wl}]$, where R_{wl} is the relative rotation matrix and T_{wl} is the translation matrix between world and LIDAR coordinate systems. In this way we can project each point in the world coordinate system onto image plane by the projection matrix $P = KM_{lc}M_{wl}$. After obtaining the corresponding image, we can project the point cloud of a tree onto the corresponding image plane using projection matrix and get the shape mask which will be used in Subsection VI-A.

VI. TREE RECONSTRUCTION

Given the point cloud of a certain tree and the corresponding image, plus the shape mask, we can reconstruct trees based on two steps: shape fitting and texture completion.

A. Shape Fitting

Rotation axes and bounding boxes are calculated from the shape mask using a heuristic method as follows: First, we use morphology operations to fill holes and remove noisy points of shape mask. Second, edge image is extracted using canny operator. We define a function $f(y) = x_{max} - x_{min}$, where $edge(x,y) \neq 0$, $edge(x,y)$ is the binarized value of pixel(x,y) in the edge image, x_{max} and x_{min} are the maximum and minimum x-coordinate values that meet the defined requirement and have the same y-coordinate value. In other words, $f(y)$ is the function of width of the tree in each line. By finding the first none-zero point of function $\partial f / \partial y$, we can get an approximate area of the main trunk, as shown in Figure 4(a). Then RANSAC algorithm[4] is run on the approximate area of the main trunk to detect the two edge lines of main trunk. The middle line of the two edge lines is considered as the rotation axis for the main trunk. The rotation axis of the tree crown is a vertical line determined by the central x-coordinate value of tree crown such as the blue line in Figure 4(b).

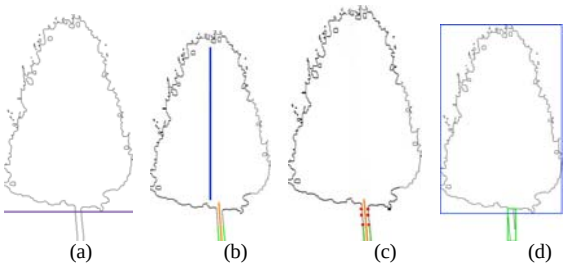


Figure 4. Process of shape fitting: a) approximate detection of the main trunk; b) rotation axes; c) accurate detection of the main trunk; d) bounding boxes.

$f(y)$ is tend to be larger than the width of the main trunk when some leaves fall down, making the detection of main trunk not very accurate. We refine the definition of function f by restrict the x_{max} and x_{min} in each line to the most nearest two points to the rotation axis of the main trunk which is obtained in the above step. As illustrated in Figure 4(c), we use the red points to calculate the trunk width in each line. Then we use the same method as before to detect the area of the main trunk again and a more accurate bounding box of the main trunk can be obtained like the green box in Figure 4(d). The blue one is the bounding box of tree crown which is acquired according to the approximate height of main trunk in Figure 4(a).

After determining the rotation axes and bounding boxes of the two parts, we also need to estimate the real size of the final model. As mentioned in Section V, trees are represented using six polygons. The size of each polygon can be determined by a scaling factor $s = (z_{max} - z_{min})/img_{height}$, where z_{min} and z_{max} are respectively the minimum and maximum z-coordinate values of the 3D bounding box of a tree, img_{height} is the height of the shape mask. Then by multiplying the factor s with the height and

width of the bounding box, we can estimate the size of each polygon.

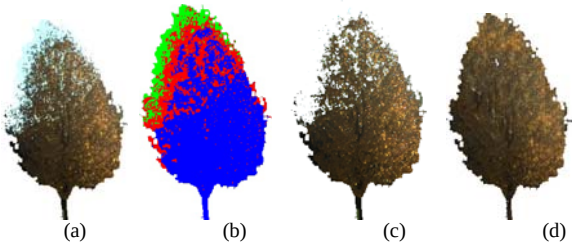
For those trees which do not have a single main trunk(a tree with a single main trunk is determined only if it has two edge lines of the main trunk in the edge image), such as Figure 5, we treat them as the tree crown and using only three polygons to represent this kind of trees.

Figure 5. Trees with more than one main trunk.

B. *Texture Completion*

Given the shape mask and the corresponding image, the coarse texture is extracted as Figure 6(a). Due to the some pixel offsets and the hollow parts of trees, some wrong texture is included such as the color of sky. We need to remove those texture and replace them with the right ones.

We use the Gaussian Mixture Model (GMM) and the Expectation Maximization (EM)[8] algorithm to cluster pixels in the space of HSV, because hue value is more constant than RGB value when illumination changes. The clustering result is shown in Figure 6(b). We compute the average values in hsv channels of all pixels and the cluster which has the nearest values is considered as the right cluster of texture for trees. Then the pixels belong to the cluster of wrong texture are removed, the resulting image containing correct texture, called img_{ct}, is shown in Figure 6(c).



(a) (b) (c) (d)

Figure 6. Texture completion: a) coarse texture; b)result of GMM clustering; c) removing the wrong texture; d) resulting texture after completion.

After removing the wrong texture, we need to fill it with correct ones. We define the texture candidate block T_i in texture completion to be a square block with a fixed size s in the img_{ct}. S_T is defined as a set of those overlapping blocks T_i . The block size s starts with an initial value and is reduced to make sure that at least one block T_i is full of texture. With s determined, img_{ct} is divided into a set of tiling blocks of size s, which we refer as S_B. Each block is assigned a label of texture status L_i = {0, 1, 2}. L_i = 0 indicates the texture of current block needs to be replaced, 1 means texture is correct and 2 means the block is empty.

After obtaining the set of texture candidate S_T and the set of tiling blocks S_B, we use Algorithm 1 to finish texture completion. The degree of connectivity for a block is the number of blocks whose L_i = 1 in the 4-neighbour. B_k is a block from S_B with the most degree of connectivity. Each iteration we select the best T_j from S_T to replace B_k in order to minimize the cost function of B_k, which is represented using equations as follows:

$$\arg\min_{T_j} \sum_{B_i \in N(B_k)} E_{hist}(T_j, B_i) + \sum_{B_i \in N(B_k)} E_{boundary}(T_j, B_i) \quad (2)$$

where $N(B_k)$ is the blocks in the neighbourhood of B_k. $E_{hist}(T_j, B_i)$ is the Bhattacharyya distance of histograms of each block:

$$d(H_1, H_2) = \sqrt{1 - \frac{1}{\sqrt{\overline{H_1}\overline{H_2}N^2}} \sum_I \sqrt{H_1(I) \cdot H_2(I)}} \quad (3)$$

Where H_1 and H_2 are the histograms of two image blocks. $E_{boundary}(T_j, B_i)$ is the sum of the squared differences of all pixels's RGB values in the boundary area between two blocks. When all the B_k whose L_i = 0 in S_B have been replaced, the algorithm stops. Then the shape mask is used to remove the over completed texture. The completed texture is shown in Figure 6(d).

```
1 while ∃L_i ∈ L, st.L_i = 0 do
2     compute degrees of connectivity for all blocks
      whose L_i = 0;
3     select the block with the most degrees B_k from
      S_B;
4     select the best candidate block T_j from S_T to
      minimize the cost in the neighbourhood of B_k;
5     replace B_k with T_j, update L_k = 1;
6 end
```

Algorithm 1: Texture completion algorithm.

When texture is filled completely, bounding boxes obtained in subsection VI-A along with the shape mask are used to extract the corresponding texture and set other parts transparent. Then the alpha texture is patched onto the corresponding polygons which make the final model that will be illustrated in next section.

VII. Experimental Results

We have tested our proposed method on the data set obtained in real world. The reconstructed trees appear realistic and vivid, as illustrated in Figure 1, 7. We use the point cloud to get shape mask which encloses the geometry information and generate the right texture from images even those with some misaligned artifacts by texture completion. A capture of resulting trees from a large-scale scene is shown in Figure 8. The ground image is extracted from google earth and trees are registered to it

by GPS coordinates. The reconstruction for different kinds of trees in Figure 8 demonstrates that our method is efficient for various trees. We can conclude that although our method of modeling trees does not maintain a high level of detail but it can capture the main characteristics of trees and make the reconstructed trees visually acceptable.

The advantages of our approach are the automation and the quick speed of reconstructing large-scale trees. Besides, our model performs well in memory footprint. Some quantitive evaluation of our approach is shown in Table 1. The times are recorded on an Intel i5-2300 2.8GHz machine. Table 2 lists the comparison between our method and the one of Livny [7, 6](Among the two tables, the Time of Pipeline does not contain the time of extracting individual trees which is done offline prior to the reconstruction process). It's clearly that our pipeline runs faster than Livny's method. The usage of memory for model representation does not differ greatly but the size of final reconstructed trees shows that our model can save a lot of memory due to the Billboard model we have adopted.

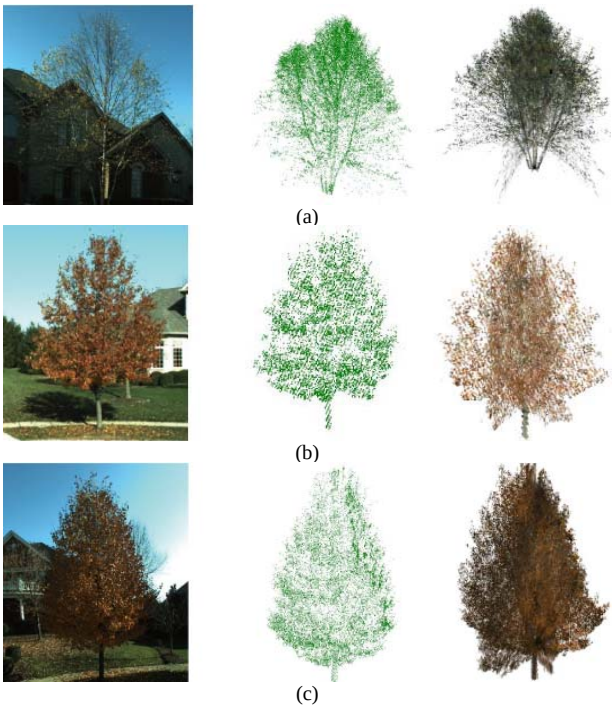


Figure 7. Some trees reconstructed from data set. Column 1 are the original images. Column 2 are the point cloud of each tree. Column 3 are the final results using our approach.

TABLE I. RESULTING MODEL SIZE AND TIME OF THE TREE RECONSTRUCTION IN FIGURE 7.

Figure	#points	Model Size	Texture Size	Time of Pipeline
7(a)	45936	208B	46.6kB	0.58s
7(b)	7792	416B	15.2kB	0.37s
7(c)	95529	416B	42.8kB	1.32s

TABLE II. THE COARSE TIME AND MEMORY FOOTPRINT FOR ONE TREE USING OUR METHOD AND LIVNY'S.

Method	Time of Pipeline	Model and Texture Size	Final tree size
Ours	< 2s	around 40kB	around 40kB
Livny[6,7]	> 20s	around 240kB	25MB

VIII. CONCLUSIONS

In this paper we presented an approach for automatically reconstructing trees in large-scale by using several polygons with alpha texture images. The geometric parameters are determined in the shape fitting process, followed by texture completion which generates correct texture while do not need carefully registered images. We demonstrated this method using data collected in an area of real world and it runs very quickly and the resulting model consumes few memory.

The quick speed and few memory usage are the key advantages of our approach. In the future, we plan to improve the model to reconstruct trees to a more precise level while preserving the advantages we have in speed and memory footprint. We are also interested in finding ways to convert existing models to ours to simplify the model representation and reduce memory footprint.

ACKNOWLEDGMENT

This work is supported by National Natural Science Foundation of China (Grant No.61100111/F020501) and Jiangsu Province (Grant No. BK2011563).

REFERENCES

[1] B. Beneš, O. Šťava, R. Mech, and G. Miller. Guided procedural modeling. In *Computer Graphics Forum*, volume 30, pages 325–334. Wiley Online Library, 2011.

[2] A. Bucksch, R.C. Lindenbergh, and M. Menenti. Skeltre-fast skeletonisation for imperfect point cloud data of botanic trees. In *Eurographics 2009 Workshop on 3D Object Retrieval*, pages 13–20. The Eurographics Association, 2009.

[3] P. de Reffye, C. Edelin, J. Françon, M. Jaeger, and C. Puech. Plant models faithful to botanical structure and development. In *ACM SIGGRAPH Computer Graphics*, volume 22, pages 151–158. ACM, 1988.

[4] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[5] H. Honda. Description of the form of trees by the parameters of the tree-like body: Effects of the branching angle and the branch length on the shape of the tree-like body. *Journal of theoretical biology*, 31(2):331–338, 1971.

[6] Y. Livny, S. Pirk, Z. Cheng, F. Yan, O. Deussen, D. Cohen-Or, and B. Chen. Texture-lobes for tree modelling. In *ACM Transactions on Graphics (TOG)*, volume 30, page 53. ACM, 2011.

[7] Y. Livny, F. Yan, M. Olson, B. Chen, H. Zhang, and J. El-Sana. Automatic reconstruction of tree skeletal structures from point clouds. In *ACM Transactions on Graphics (TOG)*, volume 29, page 151. ACM, 2010.

[8] Todd K Moon. The expectation-maximization algorithm. *Signal Processing Magazine, IEEE*, 13(6):47–60, 1996.

[9] B. Neubert, T. Franken, and O. Deussen. Approximate image-based tree-modeling using particle flows. In *ACM Transactions on Graphics (TOG)*, volume 26, page 88. ACM, 2007.

[10] P. Prusinkiewicz and A. Lindenmayer. The algorithmic beauty of plants. 1991.

[11] L. Quan, P. Tan, G. Zeng, L. Yuan, J. Wang, and S.B. Kang. Image-based plant modeling. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 599–604. ACM, 2006.

[12] A. Reche-Martinez, I. Martin, and G. Drettakis. Volumetric reconstruction and interactive rendering of trees from photographs. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 720–727. ACM, 2004.

[13] A. Runions, B. Lane, and P. Prusinkiewicz. Modeling trees with a space colonization algorithm. In *Eurographics Workshop on Natural Phenomena*, volume 2. The Eurographics Association, 2007.

[14] O. Šťava, B. Beneš, R. Mech, D.G. Aliaga, and P. Krištof. Inverse procedural modeling by automatic generation of l-systems. In *Computer Graphics Forum*, volume 29, pages 665–674. Wiley Online Library, 2010. EG10.

[15] J.O. Talton, Y. Lou, S. Lesser, J. Duke, R. Mech, and V. Koltun. Metropolis procedural modeling. *ACM Transactions on Graphics (TOG)*, 30(2):11, 2011.

[16] P. Tan, G. Zeng, J. Wang, S.B. Kang, and L. Quan. Image-based tree modeling. *ACM Transactions on Graphics (TOG)*, 26(3):87, 2007.

[17] Ping Tan, Tian Fang, Jianxiong Xiao, Peng Zhao, and Long Quan. Single image tree modeling. *ACM Trans. Graph.*, 27(5):108:1–108:7, December 2008.

[18] J. Weber and J. Penn. Creation and rendering of realistic trees. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 119–128. ACM, 1995.

[19] H. Xu, N. Gossett, and B. Chen. Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Transactions on Graphics (TOG)*, 26(4):19, 2007.

[20] Y. Zhou, Y. Yu, G. Lu, and S. Du. Super-segments based classification of 3d urban street scenes. *Int J Adv Robotic Sy*, 9(248), 2012.

[21] C. Zhu, X. Zhang, B. Hu, and M. Jaeger. Reconstruction of tree crown shape from scanned data. *Technologies for E-Learning and Digital Entertainment*, pages 745–756, 2008.

Figure 8. Reconstructed trees in a large-scale scene.