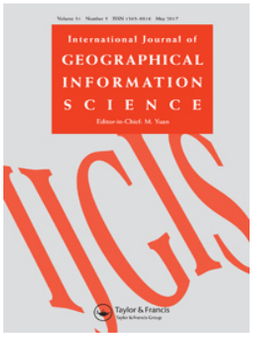


International Journal of Geographical Information Science

ISSN: 1365-8816 (Print) 1362-3087 (Online) Journal homepage: http://www.tandfonline.com/loi/tgis20

3D tree modeling from incomplete point clouds via optimization and L_1-MST

Jie Mei, Liqiang Zhang, Shihao Wu, Zhen Wang & Liang Zhang

To cite this article: Jie Mei, Liqiang Zhang, Shihao Wu, Zhen Wang & Liang Zhang (2017) 3D tree modeling from incomplete point clouds via optimization and L_1-MST, International Journal of Geographical Information Science, 31:5, 999-1021, DOI: 10.1080/13658816.2016.1264075

To link to this article: https://doi.org/10.1080/13658816.2016.1264075

INTERNATIONAL JOURNAL OF GEOGRAPHICAL INFORMATION SCIENCE, 2017
VOL. 31, NO. 5, 999–1021
http://dx.doi.org/10.1080/13658816.2016.1264075

Taylor & Francis
Taylor & Francis Group

3D tree modeling from incomplete point clouds via optimization and L_1-MST

Jie Mei[a], Liqiang Zhang[a], Shihao Wu[b,c], Zhen Wang[a] and Liang Zhang[a]

[a]State Key Laboratory of Remote Sensing Science, Beijing Normal University, Beijing, China; [b]Institute of Computer Science, University of Bern, Swiss, Switzerland; [c]Shenzhen VisuCA Key Lab, Shenzhen, China

ABSTRACT

Reconstruction of 3D trees from incomplete point clouds is a challenging issue due to their large variety and natural geometric complexity. In this paper, we develop a novel method to effectively model trees from a single laser scan. First, coarse tree skeletons are extracted by utilizing the L_1-median skeleton to compute the dominant direction of each point and the local point density of the point cloud. Then we propose a data completion scheme that guides the compensation for missing data. It is an iterative optimization process based on the dominant direction of each point and local point density. Finally, we present a L_1-minimum spanning tree (MST) algorithm to refine tree skeletons from the optimized point cloud, which integrates the advantages of both L_1-median skeleton and MST algorithms. The proposed method has been validated on various point clouds captured from single laser scans. The experiment results demonstrate the effectiveness and robustness of our method for coping with complex shapes of branching structures and occlusions.

ARTICLE HISTORY
Received 7 July 2016
Accepted 20 November 2016

KEYWORDS
Incomplete point cloud; tree skeleton; point density; dominant direction; optimization; L_1-MST

1. Introduction

Terrestrial laser scanning (TLS) technology now provides dense and accurate 3D point clouds of objects from which shape geometry or topology can be derived. Effective 3D tree reconstruction from TLS point clouds is thus an important research topic (Iovan *et al*. 2008, Dorigo *et al*. 2010, Yao and Wei 2013). It is also a challenging task since both missing data and noise are inevitable due to data acquisition via laser scanner. Many state-of-the-art methods have been proposed to reconstruct a wide variety of tree structures (Raumonen *et al*. 2013, Hackenberg *et al*. 2014, Calders *et al*. 2015). A natural approach to tree modeling is to first reconstruct tree skeletons, and then apply further geometry completion through predefined rules and heuristics (Livny *et al*. 2010). In terms of the skeleton-based 3D tree modeling, extracting tree skeletons from point clouds is one of the most important steps. Complex structures, severe occlusions, and wide variations in trees present difficulties for existing methods to accurately reconstruct the tree skeletal shape and topology. Therefore, extracting high-quality tree skeletons

CONTACT Liqiang Zhang zhanglq@bnu.edu.cn

from imperfect point clouds remains a central and challenging problem in the tree reconstruction domain.

Our method focuses more on the topological correctness than on the geometrical accuracy of the 3D tree reconstruction. In order to recover missing data more reasonably, we propose to integrate the point density of local regions and the dominant direction of each point into the completion process. More specifically, inspired by the superiority of the L_1-median skeleton algorithm (Huang *et al*. 2013), by which the curve skeletons can be extracted directly from raw point clouds without prior assumptions on the shape geometry or topology, we propose a method to efficiently extract tree skeletons from incomplete TLS point clouds, where the core technique is to consider both branch dominant direction and local point density in our optimization. The method can also handle trees subject to many human interventions, for example, those in urban parklands.

2. Related work

Many curve skeleton extraction approaches operate on complete mesh models. Verroust and Lazarus (2000) utilized the length of edges in a spanning tree to cluster the points and extracted skeletal curves from an unorganized collection of scattered points. Runions *et al*. (2007) extended the open leaf venation model (Runions *et al*. 2005) to 3D, and grew skeletal structures within tree envelopes by using points as local attractors. Bucksch and Lindenbergh (2008, 2009) divided points into octree cells, and created curve skeletons by connecting local extractions in adjacent cells. Their method can process a large number of point clouds in linear time complexity. Yet, the quality of the tree modeling results is often affected by the varying point densities. Lin and Hyyppa (2012) explored multiecho-recording mobile laser scanning for improving individual tree crown reconstruction via a mobile mapping system equipped with LiDAR.

To model main branches from incomplete tree point clouds, Xu *et al*. (2007) developed a heuristic-based approach, in which general knowledge of tree structure is utilized to generate tree meshes and then small twigs and leaves are synthetically added to form the crown geometry. However, detailed knowledge about the structure of various tree species is not always available. Based on light scattering properties obtained from scanned sample intensities, Côté *et al*. (2009) synthesized minor tree and leaf geometry. Although their method is relatively insensitive to occlusions, it requires labor-intensive parameterization. Later, TLS point clouds are combined with tree-level structural attributes (Côté *et al*. 2012) to reproduce more accurate branch structures and foliage spatial distributions. In the method of Livny *et al*. (2010), the foliage with a number of leaf clusters is distributed in the tree crown. Tiny twigs and leaves inside these clusters are created procedurally. Although the method is robust to noise, it is not flexible enough to describe branch dominant directions, and thus is unable to handle point clouds with large regions of missing data.

To cope with significant missing data, Tagliasacchi *et al*. (2009) used point normal information at each point to compensate for missing data, and then extracted curve skeletons based on a local cylindrical prior. However, point normals are usually unavailable or difficult to obtain from a raw point cloud. Aimed at this, the L_1-median algorithm was applied directly to extract curve skeletons from incomplete point clouds (Huang

et al. 2013). Although this method does not require prior assumptions on the shape geometry or topology, it is difficult to generate precise tree skeletons from incomplete point clouds because of the natural complexity of the trees. For generating accurate 3D tree models from incomplete point clouds, the refined bounding cylinders are used iteratively to capture branches. The performance of this method not only largely depends on fitting 3D shapes to the input data but also is sensitive to occlusions.

A hybrid approach which is adaptive to inhomogeneous point clouds was employed to reconstruct 3D trees (Aiteanu and Klein 2014). In densely sampled regions, the principal curvatures are applied to generate the skeleton of the branches, whereas in sparsely sampled regions a spanning tree-based algorithm is used to extract skeletons. The spanning tree-based algorithm often generates misconnections in neighboring branches. Zhang *et al*. (2014) presented a data-driven method for modeling tree from a single laser scan. The visible branches are constructed by using a cylinder matching algorithm, and non-visible branches are synthesized by a hierarchical particle flow. This method can reconstruct the branch structures in the regions of missing data. Wang *et al*. (2014) proposed a structure-aware global optimization (SAGO) method to model trees from incomplete TLS datasets. They first extracted the approximate tree skeleton by utilizing a distance minimum spanning tree (DMst) algorithm and then computed the stretching directions of the branches. Based on these stretching directions, the SAGO recovers missing data by employing a global optimization approach. Afterwards, the DMst was reapplied to obtain the refined tree skeleton from the optimized data. This method can effectively reconstruct 3D tree models from incomplete TLS point clouds. Nevertheless, the optimized point cloud of the twigs often becomes very diffuse which affects the reconstruction performance. To overcome the difficulty, the points in the regions with high point density should move to the regions with sparse points whereas the points in the regions with low point density should not move far away from their original positions (Wang *et al*. 2016). Moreover, points should not be moved along the direction of the incorrect skeleton lines or moving far away to form incorrect branches.

3. Overview of our methodology

The framework of our method is shown in Figure 1. Specifically, coarse tree skeletons are first extracted from the input point cloud by utilizing the L_1-median algorithm. Meanwhile, the dominant direction of each point and local point density of the point cloud are computed. Then, a data completion scheme is developed to recover missing data. It is an iterative optimization process that takes both dominant direction and point density into account. The optimized point cloud is merged with the input point cloud to form a new point set as the input of the next iteration. Also, the dominant direction of each point and local point density are recomputed for the new point set. The data completion scheme is reapplied to the new input point cloud and this process is repeated until a sufficiently accurate tree skeleton is generated. Next, we present a L_1-minimum spanning tree (MST) algorithm to refine tree skeletons from the optimized point cloud, which integrates the advantages of both L_1-median skeleton and MST algorithms. Finally, the radius of every node on the skeleton is computed so that the skeleton can be expanded into a real 3D tree model.

The main contributions of this work are as follows:

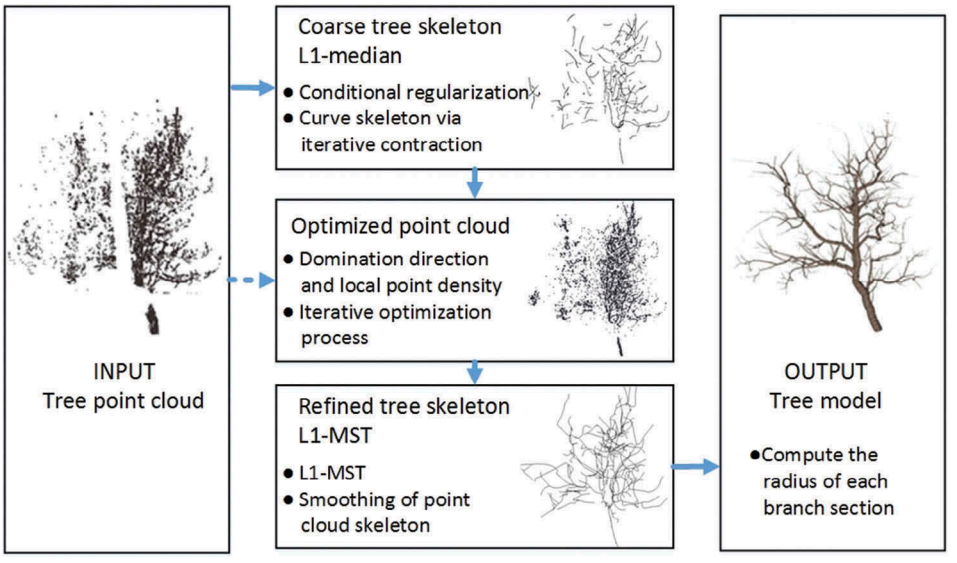


Figure 1. Overview of the proposed approach.

(i) A novel framework for reconstructing 3D trees from a TLS point cloud is developed. The framework makes the modeling results robust to noise and missing data. Moreover, it allows for unified processing of multiple trees of different types without pre-segmentation.

(ii) A data completion scheme is proposed to recover regions of missing data. It is a dominant direction and point density-aware iterative optimization process without heavy parameter tuning. The process provides a better overall approximation of the tree branches compared with the optimization reported in Wang *et al*. (2014).

(iii) A L_1-MST algorithm that integrates the advantages of the L_1-median and the MST is presented to extract fine tree skeletons from TLS point clouds. The skeleton extraction makes no prior assumptions on the shape geometry or topology.

4. Point cloud optimization

TLS tree point cloud data is generally incomplete and noisy due to severe occlusions caused by the objects in front of trees or self-occlusion. It is difficult to derive the real structures of the branches in regions of missing data. To achieve accurate tree skeletons, the key is to complete the raw point cloud. Based on the dominant direction of each point and the local point density, we develop an iterative optimization algorithm to recover the missing data and obtain more complete data. Figure 2 shows the optimization of an incomplete tree point cloud with large missing data. The dominant direction of each point and local point density can be computed from the input data by employing the L_1-median.

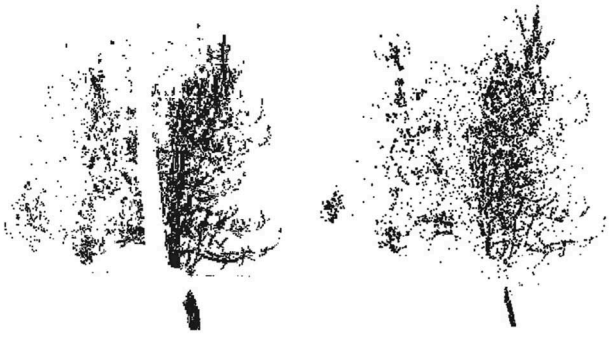


Figure 2. The optimization of an incomplete tree point cloud. (a) The raw point cloud. (b) The optimized point cloud.

4.1. *Extraction of coarse tree skeletons*

In this section, we apply the L_1-median (Huang *et al*. 2013) to extract the coarse tree skeletons from the input point clouds.

Given a set of points $Q = \{q_j\}_{j\in J}$, we use Equation (1) to obtain tree skeletons that result in an optimal set of projected points $X = \{x_i\}_{i\in I}$.

$$\arg\min_{X} \sum_{i\in I}\sum_{j\in J} \|x_i - q_j\|\theta(\|x_i - q_j\|) + R(X), \tag{1}$$

where the first term is the position of L_1-median in Q; the second term $R(X)$ is a regularization term that adds a repulsion force when a skeleton branch is formed locally. I is the set of the projected points X, and J indexes the set of the input points Q. A fast decaying smooth function $\theta(r) = e^{-r^2/(h/2)^2}$ defines the weight and the support radius h which is the size of the local neighborhood for L_1-median construction.

We define the directionality degree of x_i within a local neighborhood to describe the distribution of the point set.

$$\sigma_i = \sigma(x_i) = \frac{\lambda_i^2}{\lambda_i^0 + \lambda_i^1 + \lambda_i^2}, \tag{2}$$

where the eigenvalues $\lambda_i^0 \le \lambda_i^1 \le \lambda_i^2$ form an orthogonal frame that is the principal component of the point set. σ_i approaches to 1 which means λ_i^0 and λ_i^1 are smaller compared to λ_i^2. Therefore, the more points around x_i are aligned along a branch.

We start from an initial set of the sample points that are contracted based on an initial neighborhood size $h_0 = 2d_{bb}/\sqrt[3]{|J|}$, where d_{bb} is the diagonal length of the input Q's bounding box, and $|J|$ is the number of points in Q. Then we employ the directionality degree measure σ in Equation (2) to confirm whether a point is labeled as the branch point after the contraction. We compute σ_i for all non-branch points x_i. If σ_i>0.9, x_i is considered as a candidate of the branch point since the points in the neighborhood of x_i are well-aligned skeleton-wise.

To identify branch points from these candidates, we set a seed point x_0 that has the largest value σ. We then trace from it to the nearby candidates along the dominant Principal Component Analysis direction. The tracing process stops when there is no candidate in the local neighborhood satisfying with $\cos\left(\angle(\overrightarrow{x_i x_{i-1}}, \overrightarrow{x_i x_{i+1}})\right) \leq -0.9, i = \cdots, -1, 0, 1, \cdots$. The procedure repeats from a new seed with the largest σ among the remaining candidates until all candidates are processed.

The neighborhood size gradually increases while we fix the branch points that may result in skeleton branches being disconnected. To solve this problem, we select the bridge points at both ends of an identified skeleton branch. If two bridge points appear in a neighborhood, and the angle between the corresponding skeleton branches is larger than 155°, the branches are connected. If there are more than two bridge points in a neighborhood, we connect the corresponding branches to the midpoint of these bridge points. As a result, the branches extracted under different neighborhood sizes are connected, and a coarse tree skeleton is formed.

4.2. *Dominant directions and density of points*

To obtain accurate skeletons from the incomplete tree point cloud, we need to define the dominant direction of each point to guide the movement of points along the corresponding branches.

In general, connections of the branches in the above obtained skeleton are incorrect in a few regions. Most of the generated skeleton lines are similar to real branches. Thus we use skeleton points to compute the dominant direction of each point. For a point and its nearest neighborhood which are both on the skeleton, the direction from the point, which is closer to the root node, to the other one is regarded as the dominant direction of the two skeleton points.

The dominant directions of the points in the raw point cloud are derived through the dominant directions of the skeleton points. We obtain the *k*-nearest neighbors of a skeleton point *i* from the raw point cloud. The dominant directions of the *k* points in the raw point cloud are the same as the dominant direction of *i*. Figure 3 illustrates the

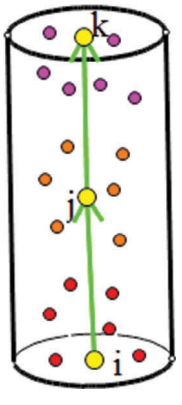


Figure 3. The dominant directions of points. The yellow nodes are skeleton points. The red, brown, purple nodes are raw points (different colors represent nearest points of different skeleton points). The green line represents dominant direction.

dominant directions of the points. The yellow points *i, j,* and *k* are skeleton points. The red, brown, and purple ones are the points in the point cloud. The six nearest red points of *i* have the same dominant direction as *i*, that is, *i* points to *j*.

Similar to the distance weight function θ in Equation (1), we define the following equation to represent the local weight density d_j of each point.

$$d_j = 1 + \sum_{j' \in J \setminus \{j\}} e^{-\left\|q_j - q_{j'}\right\|^2 / (h_0/2)^2}, \tag{3}$$

where h_0 is the initial smallest radius.

4.3. *Point cloud optimization*

We overlay the input point cloud with the skeleton points obtained by using L_1-median to form a new input point cloud. In the input point cloud, the dominant direction of each point and the local point density are estimated. Then new skeleton points are created by employing the following optimization algorithm. The optimized skeleton points overlapped with the input point cloud are used to regenerate an input dataset. Afterwards, the dominant direction of each point and local point density are recomputed to replace the previous ones. The data completion scheme is then reapplied to the new dataset. This process is repeated until the regions of missing data are filled with points.

Similar with the method of Wang *et al*. (2016), the force to the point cloud and the constraint for contracting the original point cloud toward the skeleton are added to the optimization process. The repulsive force $F_r(i)$ and constraint force $F_s(i)$ to recover regions of missing data is introduced to implement the above procedure:

$$\arg\min \sum_{i \in \text{ point cloud}} \|F_r(i) + \lambda Fs(i)\|^2, \tag{4}$$

where λ is a parameter to balance the two forces, and $\|.\|$ denotes the norm of a vector.

If point *i* is not a skeleton point, then

$$F_r(i) = \sum_{j \in \mathbf{\Omega}_i} f_r \times d_j \tag{5}$$

$$F_s(i) = d_{\max}(i) \times [(U_i - P_i) + \varphi(P_{si} - P_i)]. \tag{6}$$

If *i* is a skeleton point, we have

$$F_r(i) = \sum_{j \in \mathbf{\Omega}_i} f_r \tag{7}$$

$$F_s(i) = U_i - P_i, \tag{8}$$

where $f_r = O_i{}^T \times \frac{P_i - P_j}{|P_i - P_j|} \times O_i \times \exp(-|P_i - P_j|) / \sum_{j \in \mathbf{\Omega}_i} \exp(-|P_i - P_j|)$, O_i is the dominant direction of *i*, U_i is the original location of *i*, P_i, and P_j represent the locations of *i* and *j* after the optimization, P_{si} represents the location of the skeleton point corresponding to P_i, and d_j represents the point density of *j*. φ represents the ratio between the

contraction to the skeleton points and toward the original points. $d_{\max}(i)$ represents the highest density of the neighboring points at i, which actually controls the size of the constraint force. During the optimization process, the points in the positions with high density should move to the positions with low point density whereas points in the positions with low density should not move far away from their original positions. Compared to the average point density d of neighboring points around i, $d_{\max}(i)$ prevents points on an incorrect twig from being expanded to form a wrong branch. To prevent points from moving far away or moving along the direction of the incorrect skeleton, we constrain the points to move very short distance during the optimization process. The point cloud is refined iteratively and the process does not stop until an accurate tree model is obtained.

5. Refinement of tree skeletons

In this section, we propose the L_1-MST algorithm to obtain the fine tree skeletons.

5.1. *L_1-Minimum Spanning Tree (MST)*

During extraction of the tree skeletons, L_1-median uses the neighborhood radius to connect the bridge points at both ends of the branches. That is, if a bridge point falls into the neighborhood radius of another bridge point, the two bridge points are connected each other. However, the iterative contraction of L_1-median will stop if the sample points become skeleton points, that is, the sample points have connected with other points. Some branch skeletons are not connected at the moment (see Figure 4(b)). Thus, L_1-median cannot accurately describe the local spatial distribution of tree points. As a result, the extracted skeletons are lack of details and some of twigs are often missing.

The MST is a spanning tree in which the sum of the edge weights is no larger than those of any other spanning trees so that the short edges are first connected. The main advantage of the MST is that it can preserve the local spatial structure of the point cloud. However, the MST is sensitive to noise. As shown in the red rectangles in Figure 4(c), it is observed that there are some incorrect topological relationships among the branches in the generated tree skeleton by using the MST.

In this paper, we develop a new algorithm, termed L_1-MST algorithm, that integrates advantages of the L_1-median and MST to extract fine tree skeletons from the optimized point cloud (Figure 4(d)). The steps of the L_1-MST algorithm are as follows:

(i) L_1-median is applied to extract the tree skeleton from the optimized point cloud. Then we obtain the skeleton points S_{L1}, skeleton edges U_{L1}, and root node. The skeleton points S_{L1} are overlaid with the optimized point cloud to form a new input point cloud O.

(ii) In order to repair the unconnected edges in U_{L1}, we find out the certain points in the optimized point cloud, and put them into S_{L1}. Then we obtain their connectivity with the skeleton points by using the MST.

(iii) To ensure the weight sum of all edges is the smallest, and the weight sum of the edges from each point to the root node is the smallest in the L_1-MST, the points

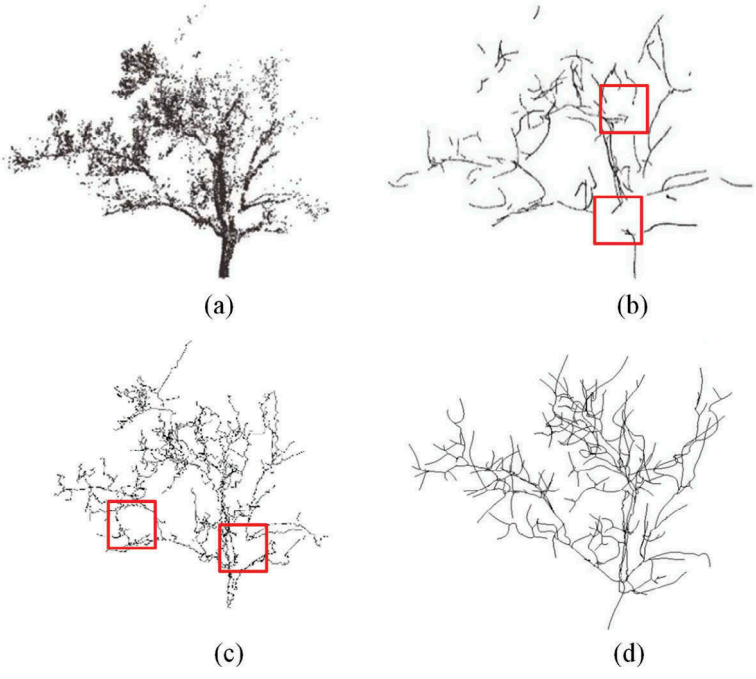


Figure 4. Extraction of the tree skeleton using different algorithms. (a) The tree point cloud. (b) The tree skeleton extracted by the L1-median. Some of branches are not connected. (c) The tree skeleton extracted using the MST, which preserves more local spatial features of the branches, but introduces incorrect topologies. (d) The tree skeleton extracted through our method.

in the optimized point cloud that can be put into S_{L1} will be satisfied with the following conditions: As shown in Equation (9), the distance d_{ij} between two points i and j in O or S_{L1} is taken as the weight of their edge. The distance d_{ij} between a point i in the skeleton points and a point j in O is taken as the weight of their edge. The distance d_{ij} between two skeleton points i and j divided by c is taken as the weight of their edge, which can ensure the edge between two skeleton points remains unchanged. By using Equation (10), we obtain the new skeleton points P_s from Q which can connect with U_{L1}.

$$w(i,j) = \begin{cases} d_{ij}/c & i,j \in S_{L1} \\ d_{ij} & i,j \in Q \\ d_{ij} & i \in S_{L1}, j \in Q \\ d_{ij} & i \in Q, \in S_{L1} \end{cases}, \tag{9}$$

where c is a constant for controlling the edges between two points in S_{L1}.

$$\arg\min_{x \in Q} \sum_{a,b \in S_{L1}} \left(w_{a,b} + w_{a,x} + w_{b,x}\right) + \sum w_{MST}(x), \tag{10}$$

where $w_{a,b}$ is the weight of the edge between skeleton points a and b; $w_{a,x}$ is the weight of the edge between a and point x in O; and $w_{b,x}$ is the weight of the edge

between b and x. $w_{MST}(x)$ is the weight sum of all edges from x to the root node obtained by using the MST.

(iv) The points in P_s are pushed into S_{L1}. Thus, we obtain the complete skeleton points $S_{L1\text{-}MST}$ and skeleton edges $U_{L1\text{-}MST}$. Finally, the outliers are removed from $S_{L1\text{-}MST}$, and we can get the final tree skeleton.

5.2. *Smoothing of tree skeletons*

The tree skeleton obtained using the L_1-MST algorithm holds the connection relationship among branches, but the extension of the tree branches is not natural. We need to smooth these tree branches so that they conform to the natural extension of the real branches. A Laplacian-based contraction process works well in obtaining a skeleton from the mesh (Tagliasacchi *et al*. 2009). Here we also use such a process to smooth the extracted skeleton. To keep them reflecting the tree shape and structure correctly, the point cloud should contract toward the tree skeleton along the dominant direction of the branch segment. L which is the Laplace operator used in this paper is represented by Equation (11). This process can reduce the disturbance of the noise in the original points.

$$Lij = \begin{cases} \omega_{ij} = e^{-d(i,j)} \times (\cot\theta_i + \cot\theta_j) & if\,(i,j) \in \mathbf{E} \\ \sum_{(i,k)\in \mathbf{E}}^{k} - \omega_{ik} & if \quad i = j \\ 0 & otherwise \end{cases}, \tag{11}$$

where L_{ij} is an element of L; i, j, and k are three points on the tree; θ_i is the angle between edge (i, j) and the line connecting i with the projection of i on the corresponding skeleton line; θ_j is the angle between edge (i, j) and the line connecting j with the projection of j on the corresponding skeleton line; d(i, j) is the distance between i and j. Figure 5 illustrates θ_i and θ_j. **E** is the set of edges; if i and j belong to the same segment or two adjacent branch segments, an edge is added to them. In Equation (11), if θ_i is close to zero, ω_{ij} enlarges quickly and i can be quickly shrunken. Here, the points are contracted to the skeleton points. To define the one-ring neighbors, we also project the points connecting with i on the plane whose normal vector from the projection of i to j.

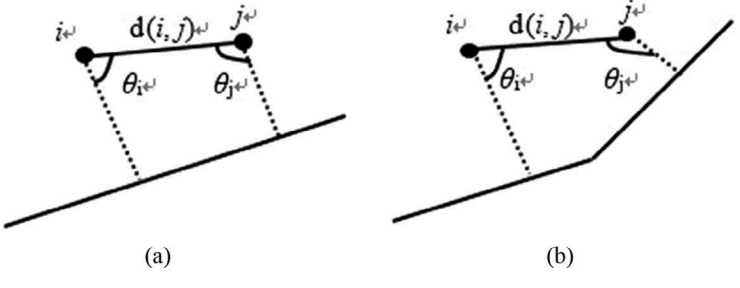


Figure 5. θi and θj in Equation (11). (a) Points i and j belong to the same skeleton line (b) i and j belong to different skeleton lines.

Then we use the Laplacian-based contraction process similar with the method of Tagliasacchi *et al*. (2009). The procedure is repeated for several iterations before the points become the skeleton points.

Equation (12) is solved to derive the skeleton points.

$$\begin{bmatrix} \mathbf{W}_L\mathbf{L} \\ \mathbf{W}_H \end{bmatrix} \mathbf{V}' = \begin{bmatrix} \mathbf{0} \\ \mathbf{W}_H\mathbf{V} \end{bmatrix}, \tag{12}$$

where $\mathbf{W}_L$ and $\mathbf{W}_H$ are diagonal matrices; $\mathbf{W}_L$ controls the smoothness, and $\mathbf{W}_H$ controls the similarity to the original points; the value of the *i*th diagonal element of $\mathbf{W}_L$ is defined as $\mathbf{W}_{L,i}$ while the value of the *i*th diagonal element of $\mathbf{W}_H$ is defined as $\mathbf{W}_{H,i}$; $\mathbf{V}'$ is the contracted point cloud, and $\mathbf{V}$ is the original one.

Because the Laplacian-based contraction can smooth the juts, the points at the end of twigs are contracted to disappear. We decrease the smoothness weight of the points at these positions in $\mathbf{W}_L$. Compared with the points which are not at the end of twigs, the points at the end of twigs usually protrude over surrounding points. Therefore, if *i* is one of these points, the sum of angles of the dominant direction of *i* and the directions of the lines between *i* and its surrounding points is small. Based on this, the value of W_L is defined in Equation (13).

$$W_{L,i} = \sum\nolimits_{j\in\mathbf{\Omega}_i} \sin\theta_2 \times \exp(-d(i,j)) / \sum_{j\in\mathbf{\Omega}_i} \exp(-d(i,j)), \tag{13}$$

where $\mathbf{\Omega}_i$ is the set of points adjacent to point *i* in $\mathbf{E}$; θ_2 is the angle between edge (*i, j*) and the dominant direction of *i*.

Figure 6(b) shows the smoothed result of a tree skeleton using the above smoothing process. We notice that the smoothed tree skeleton is more natural and consistent with the extension of the real tree branches.

Figure 6. Smoothing a tree skeleton. (a) The original tree skeleton. (b) The smoothed tree skeleton.

Finally, we compute the radii of trunks and branches using the method of Wang *et al*. (2014), and thus the smoothed tree skeletons are expanded into 3D tree models.

6. Experimental results

We perform both qualitative and quantitative evaluations of our method on several tree TLS point clouds. To further validate the performance of our method, we also compare our method with the related state-of-the-art approaches, such as the methods of Livny *et al*. (2010), Huang *et al*. (2013), and Wang *et al*. (2014). In some figures of Section 6.1, zoom-ins of tree branches or skeletons are utilized to magnify fine geometric detail, so that the readers can easily recognize the differences of the branch reconstructions.

The used point clouds have been captured by RIEGL LMS-Z360 and RIEGL LMS-Z620 scanners in a single scan. Since all the trees were scanned during the early spring and winter seasons, they contained relatively few leaves that consequently did not completely occlude the branches and twigs.

6.1. *Qualitative evaluations*

Figure 7 compares the tree skeletons extracted from an inhomogeneous point cloud by employing different approaches. The left side of the tree faces the scanner, so the points are dense, and the points in the right side are sparse and noisy due to self-occlusion. In the two tree point clouds shown in Figure 8, the amount of missing data is large. In Figure 9, the tree point cloud has more significant missing data (red rectangles of Figure 9(a)), and the branch sizes and shapes vary greatly. Compared with the other three methods, our method is more robust to noise and point nonuniformity since it consolidates the point cloud through the dominant direction and point density-based optimization process, and the L_1-MST can derive the correct connections among the branches from the optimized point cloud.

To further evaluate the robustness of our method to missing data, we removed a subset of points from a complete dataset. For example, the red box of Figure 10(a) illustrates that a subset is manually deleted from the complete point cloud in Figure 10(e). From zoom-ins in Figure 10(g), it is noted that the reconstructed tree shown in Figure 10(b) is very similar to the one in Figure 10(f) and the branches in the region of missing data are consistent, whereas in the region of missing data, the tree skeletons produced by the methods of Livny *et al*. (2010) and Wang *et al*. (2014) have some wrong geometries. It demonstrates that the presented optimization process has the ability to recover large missing data.

To highlight the robustness of our method to extraction of the fine tree skeletons with complex topologies among branches, the L_1-MST is used to extract the skeleton of a tree with dense branches and complex structures from the point cloud (Figure 11(a)). In general, the skeleton obtained using the method of Wang *et al*. (2014) preserves the global spatial structure of the tree correctly, but in some regions the connections among branches are wrong (see the red rectangles in Figure 11(c)). The skeleton obtained using our method can keep the branch structures well (red rectangles of Figure 11(d)).

Our method can also precisely model multiple trees from an imperfect point cloud. Figure 12(a) shows the point cloud of three trees with leaf crowns scanned far from the

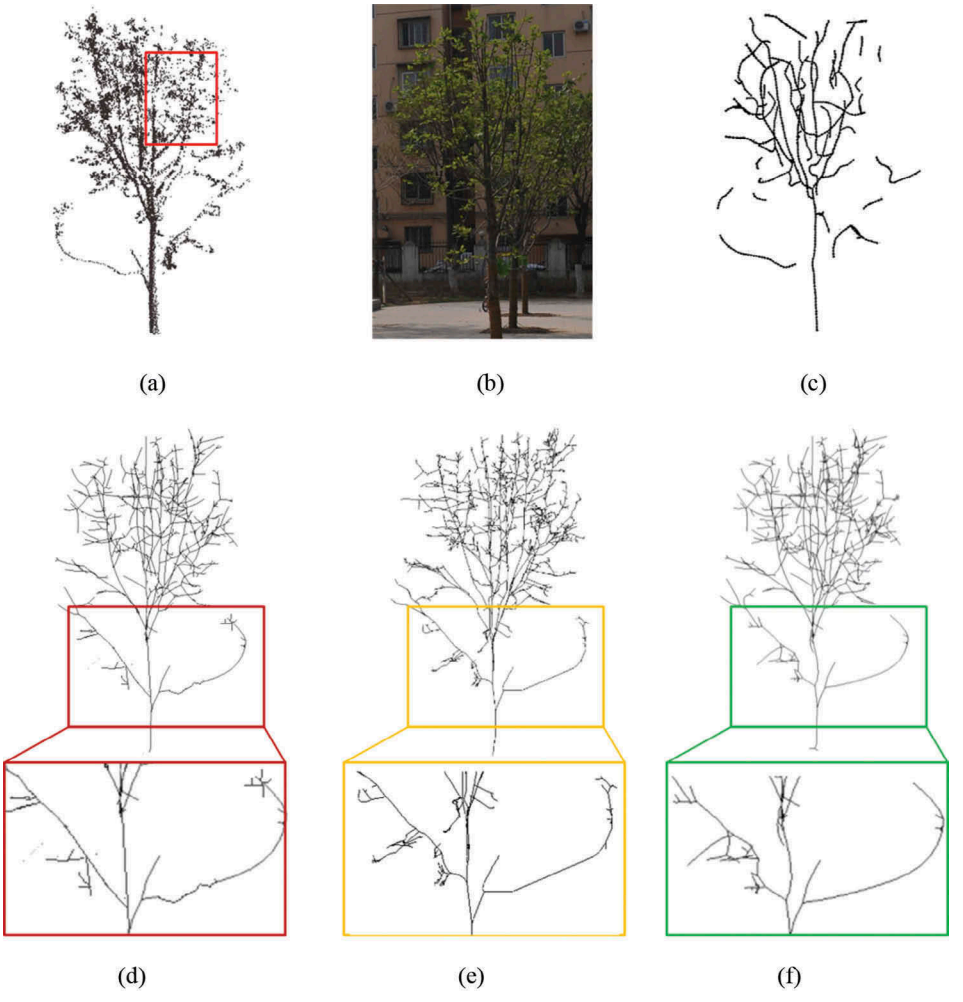


Figure 7. Reconstruction of a tree with nonuniform point density. (a) The raw point cloud. (b) Photograph of the tree. (c) The tree skeleton obtained using L1-median (Huang et al. 2013). (d) The tree skeleton obtained using our method. (e) The tree skeleton obtained using the method of Livny et al. (2010). (f) The tree skeleton obtained using the method of Wang et al. (2014).

scanner. The crowns of the three trees are thick, and the relationships among branches are complicated. Moreover, stripped regions of the missing data appear on their crowns. In addition, self-occlusions are serious as shown in the red boxes of Figure 12(e). Compared with the results in Figure 12(b–c), our method can accurately reconstruct the tree. In Figure 12(b), the branch shape from the bottom left to the upper right is affected by the missing data, but the connection of the branch is still retained correctly, whereas the branches are not reconstructed in Figure 12(d).

To further validate the performance of our method, we model the trees from the point cloud of a large urban environment. Figure 13(a) illustrates the point cloud of an urban scene containing buildings and people (see the photograph in Figure 13(b)). The main difficulty for modeling the trees in a large urban environment is data quality. Large

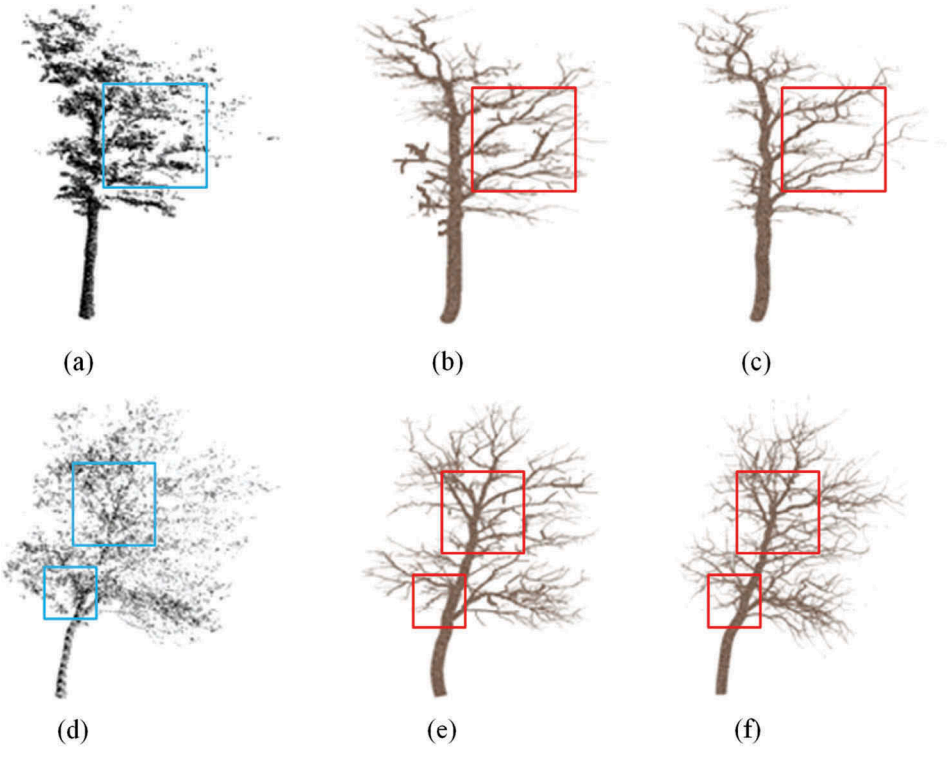


Figure 8. Reconstruction of 3D trees from an even more incomplete point cloud. The first column shows the raw tree point cloud. The second column shows the 3D tree models reconstructed using our method. The third column shows the 3D tree models reconstructed using the method of Wang et al. (2014).

distances between the scanner and scanned objects also imply low precision or high level of noise. As a result, the captured point cloud typically exhibits significant missing data due to occlusion, as well as uneven point density. The whole scene contains 22 trees composed of 1,173,621 points, and the computational time for modeling them is 26 min. The 3D tree models are shown in Figure 13(c). From Figure 13(b–c), it is noted that the structures and stretching directions of the branches in each modeled tree are retained well.

6.2. *Quantitative evaluations*

We quantitatively compare the performance of our method with those of the methods of Livny *et al*. (2010) and Wang *et al*. (2014).

We first simulate point clouds by the virtual scanning and reconstruct the corresponding tree models. In the simulation process, we set the point light source as a TLS simulator to resample the tree models where the scanner can be seen as a point light source. The TLS simulator is located at the same position as the real scanner would have been placed in the real world, and illuminates the target with the same horizontal and vertical angle spacing as the real scanner. Consequently, the

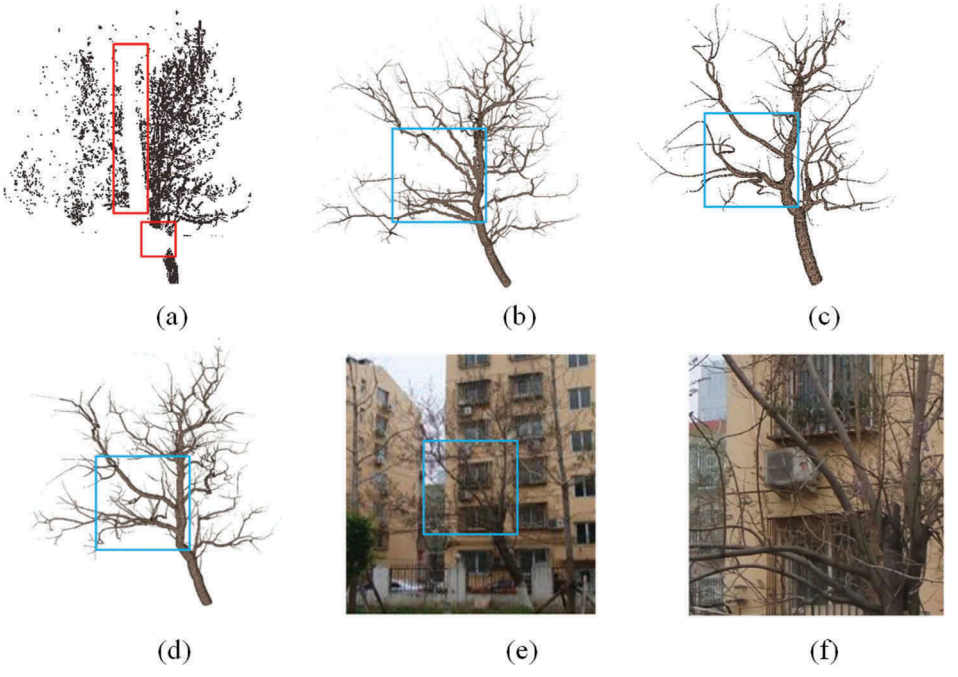


Figure 9. Modeling of a tree with a large amount of missing data. (a) The raw point cloud. (b) The tree model obtained using our method. (c) The 3D tree model obtained using the method of Wang et al. (2014). (d) The tree model obtained using the method of Livny et al. (2010). (e) Photograph of the tree. (f) Zoom-in of the branches in the blue rectangle in (e).

intersection points between the light ray and the tree model are the resampled point cloud. We use Ref-T to denote the point cloud resampled from the tree model through the process described above.

To validate the robustness and accuracy of our method in dealing with the incomplete point cloud, we imitate the data missing area by removing points from the original model. We first use the two tree models shown in Figures 9(b) and 10(f) to obtain Ref-T. Next, the points of some manually selected regions, as shown in the red boxes in Figures 9(a) and 10(a), are removed from Ref-T, and the rest data of Ref-T form a new tree model. We use Des-T to denote the point cloud resampled from this new tree model.

The normalized differences (Côté *et al*. 2009) are also used here. They are derived from the point cloud simulated from new models and the real tree models. In order to calculate the normalized difference of points in each voxel, Ref-T and Des-T are discretized in a 3D volume whose voxels are of a 0.2 m side length. Obviously, the smaller this difference is, the more similar the two models are. Figure 14 shows the quantitatively performance comparison of our method with those of Wang *et al*. (2014) and Livny *et al*. (2010) by using the normalized difference between Ref-T and Des-T. It is noted from these results that the mean values and the standard deviations of the normalized differences are all near zero, which shows that Des-T is similar to Ref-T. However, by comparing the results of Figure 14(a,b), it is clear that the mean values and the standard deviations of the normalized differences of the new method are consistently smaller

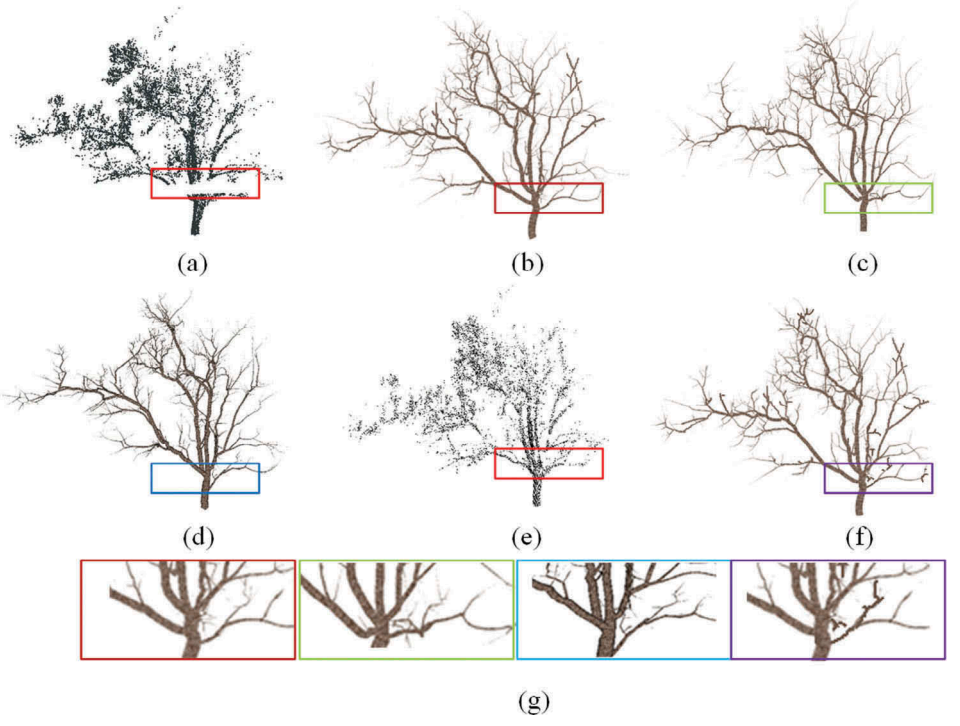


Figure 10. Comparisons of the modeling results from a point cloud. (a) The point cloud from which some points are removed. (b) The tree model reconstructed from the point cloud in (a) using our method. (c) The tree model reconstructed from the point cloud in (a) using the method of Wang et al. (2014). (d) The tree model reconstructed from the point cloud in (a) using the method of Livny et al. (2010). (e) The original point cloud. (f) The tree model reconstructed from the point cloud in (e) using our method. (g) The zoom-ins of the branches in the rectangles in (b), (c), (d), and (f), respectively.

than those of Wang *et al*. (2014). The results illustrated in Figure14(c,d) reflect the same situation. These results clearly validate the higher accuracy of our tree modeling results in dealing with the missing data.

In the following, the accuracy of the radii of our reconstructed branch sections is validated. By taking field measurements, we have obtained diameter at breast height (DBH) data for 22 trees. For each tree, the points in the region at a 1.25–1.35 m height are intentionally removed because the DBH is the diameter measured at 1.3 m above the ground. The DBH of the experimental data is estimated by using the method of Wang *et al*. (2014). As shown in Figure 15, the red line is a diagonal line that indicates an ideal reconstructed DBH with respect to the measured DBH. The black line is the fitted line, which indicates the relationship between our reconstructed DBH and measured DBH. The fitting linear equation $y = 0.8009x+6.0174$ of the points is close to the diagonal line. $R^2 = 0.9135$, which means that the measured DBHs are very close to the true DBHs.

Apart from geometric accuracy, we are also, if not more, interested in the topological correctness. Therefore, we quantitatively evaluate the topological aspects of the reconstructed models. In the tree model, each branch is taken as a node. We first label the

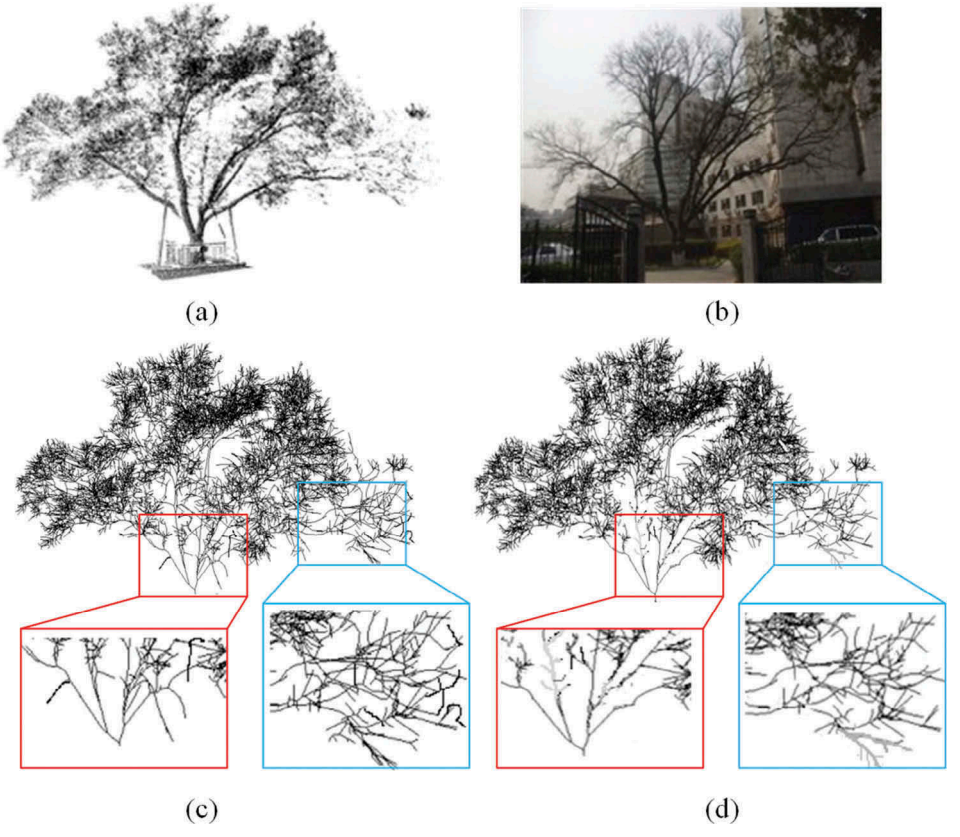


Figure 11. Extraction of the tree skeleton from a point cloud. (a) The tree point cloud. (b) Photograph of the tree. (c) The tree skeleton extracted using the method of Wang et al. (2014). (d) The tree skeleton extracted using our method.

nodes according to a certain order, and then those nodes are stored into a 2D array **Edge**. If the nodes i and j are connected, Edge $[i, j] = 1$; otherwise, Edge $[i, j] = 0$.

Given two arrays $\mathbf{Edge_r}$ and $\mathbf{Edge_s}$. Edge_r stores the nodes of the tree model reconstructed from Ref-T, and $\mathbf{Edge_S}$ stores the nodes of the tree model reconstructed from Des-T. Given two nodes i and j from $\mathbf{Edge_r}$, we find the two nodes k and l that have the closest spatial positions with i and j in $\mathbf{Edge_s}$. If the connection relationship of i and j is different from that of k and l, we record it as a topological error. In this way, the size of the nodes between $\mathbf{Edge_r}$ and $\mathbf{Edge_s}$ with different connection relationships can be computed, thus we can obtain the overall topological error between two tree models. Figure 16 illustrates the quantitative evaluation for topological correctness. The nodes i_r, j_r, k_r, and l_r in the tree model reconstructed from Ref-T have the closest spatial positions with i_s, j_s, k_s, and l_s in the tree model reconstructed from Des-T. It is noted that the connection relationship of i_r and j_r is similar with that of i_s and j_s while the connection relationship of j_r and l_r is different from that of j_s and l_s, which we take as a topological error.

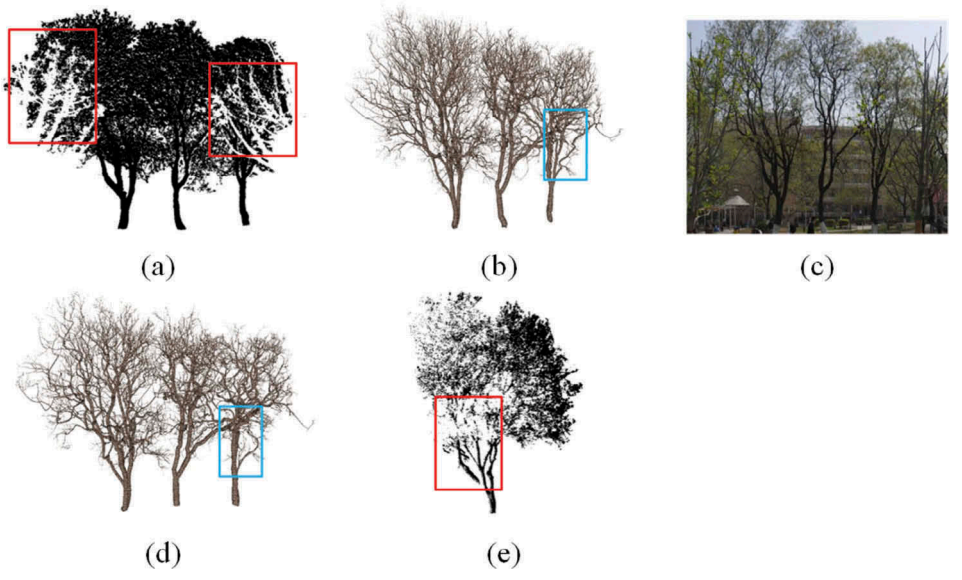


Figure 12. Reconstruction of multiple trees with thick crowns and large occlusions. (a) The raw point cloud. (b) The 3D tree model obtained using our method. (c) The tree photo. (d) The 3D tree model obtained using the method of Wang et al. (2014). (e) The point cloud of the left tree viewed from another location.

From Table 1, it is noted that our reconstructed trees has a higher accuracy. Table 2 lists the topological errors, that is, the percentages of the misconnected nodes among all nodes, obtained by the three methods. From the table, it is noted that the topological errors obtained by using our method are the smallest.

We also evaluate the modeling efficiency of the proposed method. From Table 3, it is noted that the computational cost for tree reconstruction depends on the volume and quality of the input data. In general, the computational cost is less and acceptable when the number of the input point cloud is not large. For the large-scale scene with large regions of missing data, more time is required to reconstruct the 3D tree models.

6.3. *Limitations of the presented method*

The limitations of our method lie in the following aspects:

(i) Our method algorithmically repairs the branches in regions of missing data only by using structural information on the visible parts of those branches. Thus, the skeletons in areas where the branches and twigs are completely occluded cannot be extracted. It also cannot obtain the branch skeletons of trees with thick leaves. Moreover, for the tree point cloud with large missing data, our method may miss certain fine structures and produce erroneous outputs. For instance, some branches and twigs fail to be generated (compare Figure 10(b,f)).

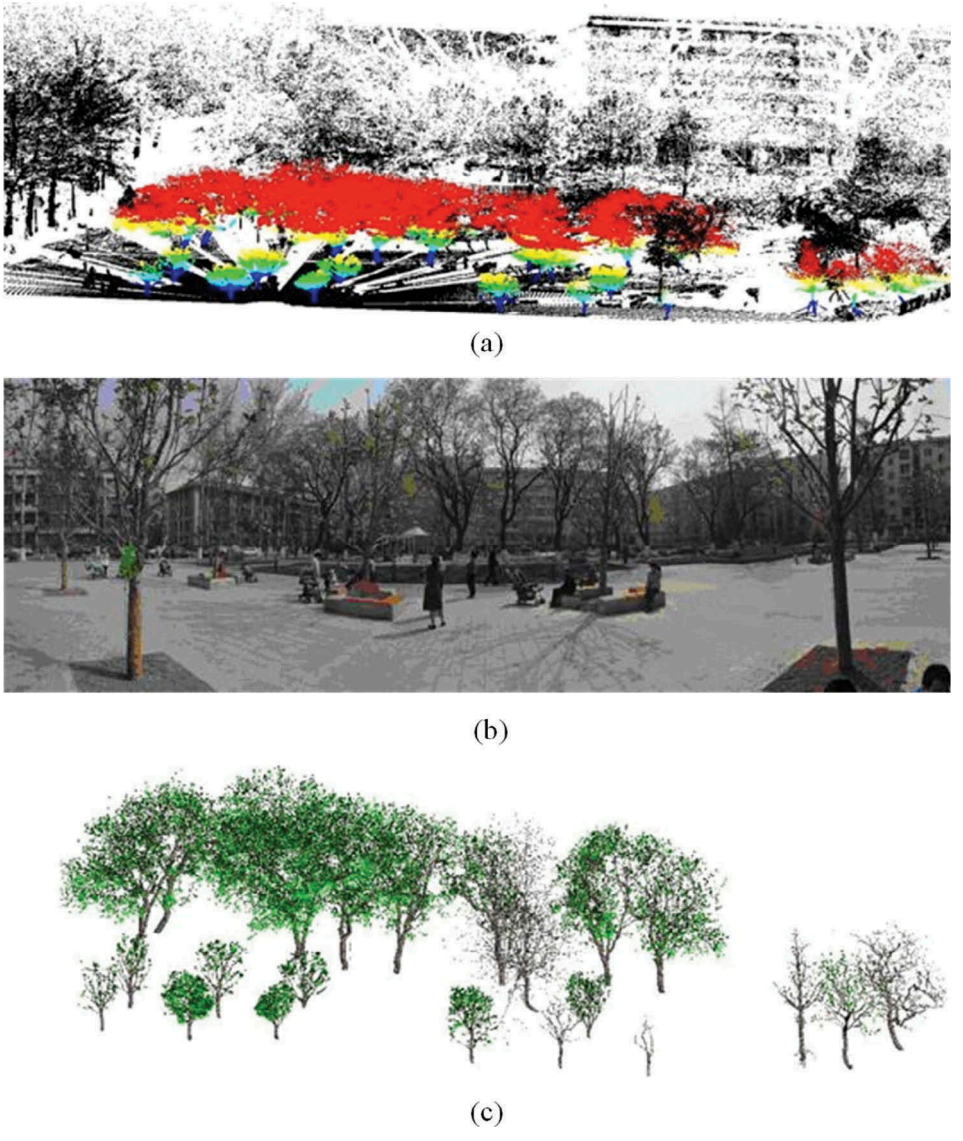


Figure 13. Multi-tree modeling. (a) Hypsometric tints of point cloud in the scene. (b) The scene photo. (c) The reconstructed 3D tree models.

(ii) For a large point cloud, the tree skeleton extraction is time sensitive. Solutions to speed up the modeling process are needed.

(iii) Our method enhances the modeling accuracy of branches and twigs compared with the methods of Livny *et al*. (2010), Huang *et al*. (2013), and Wang *et al*. (2014). However, for the trees containing very close branches, our method is still difficult to separate the points of one branch from other branches only by using

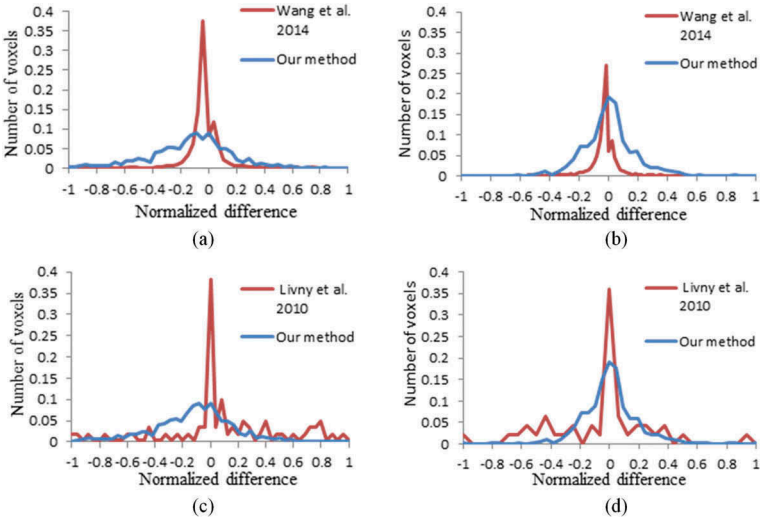


Figure 14. Comparison of the normalized differences between Ref-T and Des-T. (a) For the tree in Figure 9, the normalized differences generated by our method and that of Wang et al. (2014), respectively. (b) For the tree in Figure 10, the normalized differences generated by our method and that of Wang et al. (2014), respectively. (c) For the tree in Figure 9, the normalized differences generated by our method and that of Livny et al. (2010), respectively. (d) For the tree in Figure 10, the normalized differences generated by our method and that of Livny et al. (2010), respectively.

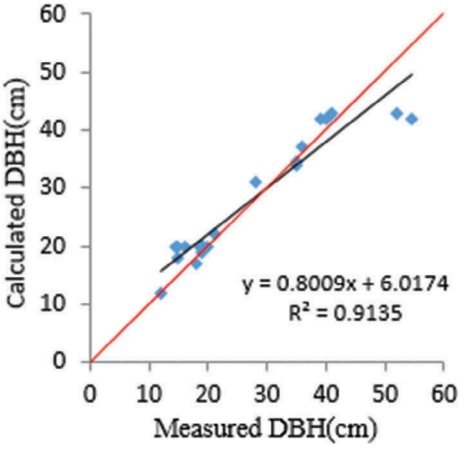
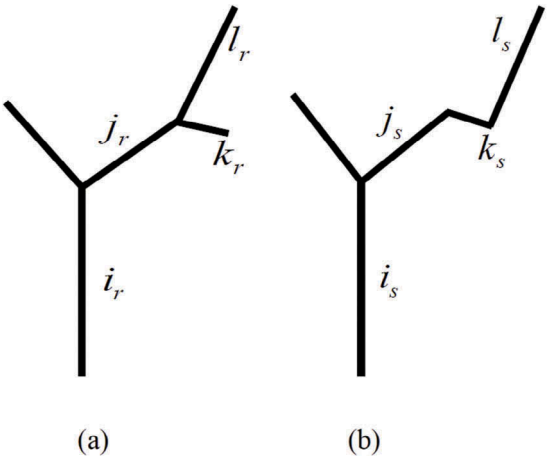


Figure 15. The measured DBH of trees and the calculated DBH of our modeled trees.

spatial information. As a result, it often connects the points by mistake, and thus produces skeletons with incorrect topologies (highlighted in Figure 12(d)).

(iv) Our method focuses more on the topological structure of the tree model and less on the geometrical properties such as volumes, lengths, and diameters of the branches.



	i_r	j_r	k_r	l_r
i_r	0	1	0	0
j_r	1	0	1	1
k_r	0	1	0	1
l_r	0	1	1	0

Edger

	i_s	j_s	k_s	l_s
i_s	0	1	0	0
j_s	1	0	1	0
k_s	0	1	0	1
l_s	0	0	1	0

Edges

Figure 16. The quantitative evaluation of the topological error of the reconstructed trees. (a) The tree model reconstructed from Ref-T. (b) The tree model reconstructed from Des-T. (c) Edger and Edges.

Table 1. Normalized differences between the three generations using three methods.

Methods		Figure 9	Figure 10
Our method	Average	0.0234	0.0276
	Std.	0.0268	0.0478
The method of Livny *et al.* (2010)	Average	0.0255	0.0300
	Std.	0.0540	0.0626
The method of Wang *et al.* (2014)	Average	−0.0249	−0.0233
	Std.	0.1311	0.1160

Table 2. The topological errors generated using the three methods.

	Figure 9	Figure 10
Our method	6.88%	3.21%
The method of Wang *et al.* (2014)	25.21%	26.87%
The method of Livny *et al.* (2010)	35.96%	33.01%

Table 3. Computation efficiency for 3D tree reconstruction.

Figures showing the data	The number of point cloud	Computation cost(min)
Figure 7	11,855	3.6
Figure 8(a)	10,642	3.2
Figure 8(d)	13,130	3.9
Figure 9	6990	1.3
Figure 10	10,503	3.1
Figure 11	111,833	53.9

7. Conclusions

In this paper, we have proposed an approach for modeling trees from incomplete point clouds. The iterative optimization process integrating with the dominant direction of each point and local point density is developed to repair the regions of missing data. The L_1-MST extracts tree skeletons from the optimized point cloud automatically without

prior assumptions on the shape geometry or topology. Qualitative evaluations illustrate that the modeled tree skeletons using our method can produce credible visual results, and quantitative evaluations show that our modeled trees keep the topological relationships among branches correctly and have a high reconstruction accuracy. At the same time, our method is very robust to missing data and noise.

In future work, we will improve the optimization techniques that speed up the convergence, and integrate shape geometry and topology into the optimization process to further promote large-scale tree modeling performance. Similar with Zhang *et al*. (2014), we try to obtain tree growth rules from the recovered tree models, and recognize the specific types of trees from visible branching patterns. Thus, we can simulate tree growth accurately or incorporate aspects of the trees genotype into our models to allow them to react to the environments.

Acknowledgments

The authors would like to thank Assoc Prof Shawn Laffan and the reviewers for their thoughtful and detailed comments which have helped them to improve the scientific contribution as well as the presentation of this paper.

Disclosure statement

No potential conflict of interest was reported by the authors.

References

Aiteanu, F. and Klein, R., 2014. Hybrid tree reconstruction from inhomogeneous point clouds. *The Visual Computer*, 30 (6–8), 763–771. doi:10.1007/s00371-014-097

Bucksch, A. and Lindenbergh, R., 2008. CAMPINO—A skeletonization method for point cloud processing. *ISPRS Journal of Photogrammetry and Remote Sensing*, 63 (1), 115–127. doi:10.1016/j.isprsjprs.2007.10.004

Bucksch, A., Lindenbergh, R.C., and Menenti, M., 2009. SkelTre-fast skeletonisation for imperfect point cloud data of botanic trees. *The Visual Computer*, 26, 1283–1300. doi:10.2312/3DOR/3DOR09/013-020

Calders, K., *et al*., 2015. Nondestructive estimates of above-ground biomass using terrestrial laser scanning. *Methods in Ecology & Evolution*, 6, 198–208. doi:10.1111/2041-210X.12301

Côté, J.-F., *et al*., 2012. A fine-scale architectural model of trees to enhance LiDAR-derived measurements of forest canopy structure. *Agricultural and Forest Meteorology*, 166–167, 72–85. doi:10.1016/j.agrformet.2012.06.007

Côté, J.-F., *et al*., 2009. The structural and radiative consistency of three-dimensional tree reconstructions from terrestrial lidar. *Remote Sensing of Environment*, 113 (5), 1067–1081. doi:10.1016/j.rse.2009.01.017

Dorigo, W., *et al*., 2010. An application oriented automated approach for co-registration of forest inventory and airborne laser scanning data. *International Journal of Remote Sensing*, 31, 1133–1153. doi:10.1080/01431160903380581

Hackenberg, J., *et al*., 2014. Highly accurate tree models derived from terrestrial laser scan data: a method description. *Forests*, 5 (5), 1069–1105. doi:10.3390/f5051069

Hackenberg, J., *et al*., 2015. Non destructive method for biomass prediction combining TLS derived tree volume and wood density. *Forests*, 6, 1274–1300. doi:10.3390/f6041274

Huang, H., *et al*., 2013. L1-medial skeleton of point cloud. *ACM Transactions on Graphics*, 32 (4), 65–1. doi:10.1145/2461912.2461913

Iovan, C., Boldo, D., and Cord, M., 2008. Detection, characterization, and modeling vegetation in urban areas from high-resolution aerial imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 1 (3), 206–213. doi:10.1109/JSTARS.2008.2007514

Lin, Y. and Hyyppa, J., 2012. Multiecho-recording mobile laser scanning for enhancing individual tree crown reconstruction. *IEEE Transactions on Geoscience and Remote Sensing*, 50 (11), 4323–4332. doi:10.1109/TGRS.2012.2194503

Livny, Y., *et al*., 2010. Automatic reconstruction of tree skeletal structures from point clouds. *ACM Transactions on Graphics*, 29 (6), 151. doi:10.1145/1882261.1866177

Raumonen, P., *et al*., 2013. Fast automatic precision tree models from terrestrial laser scanner data. *Remote Sensing*, 5, 491–520. doi:10.3390/rs5020491

Runions, A., *et al*., 2005. Modeling and visualization of leaf venation patterns. *ACM Transactions on Graphics*, 24 (3), 702–711. doi:10.1145/1073204.1073251

Runions, A., Lane, B., and Prusinkiewicz, P., 2007. Modeling trees with a space colonization algorithm. In: D. Ebert, S. Mérillou, eds. *Proceedings of the Third Eurographics conference on Natural Phenomena, Eurographics Association Aire-la-Ville, Switzerland*. Switzerland, 7, 63–70. doi:10.2312/NPH/NPH07/063-070

Tagliasacchi, A., Zhang, H., and Cohen-Or, D., 2009. Curve skeleton extraction from incomplete point cloud. *ACM Transactions on Graphics*, 28 (3), 71. doi:10.1145/1531326.1531377

Verroust, A. and Lazarus, F., 2000. Extracting skeletal curves from 3D scattered data. *The Visual Computer*, 16 (1), 15–25. doi:10.1007/PL00007210

Wang, Z., *et al*., 2014. A structure-aware global optimization method for reconstructing 3-D tree models from terrestrial laser scanning data. *IEEE Transactions on Geoscience and Remote Sensing*, 52 (9), 5653–5669. doi:10.1109/TGRS.2013.2291815

Wang, Z., *et al*., 2016. A local structure and direction-aware optimization approach for three-dimensional tree modeling. *IEEE Transactions on Geoscience and Remote Sensing*, 54 (8), 4749–4757. doi:10.1109/TGRS.2016.2551286

Xu, H., Gossett, N., and Chen, B., 2007. Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Transactions on Graphics*, 26 (4), 19. doi:10.1145/1289603.1289610

Yao, W. and Wei, Y., 2013. Detection of 3-D individual trees in urban areas by combining airborne LiDAR data and imagery. *IEEE Geoscience and Remote Sensing Letters*, 10 (6), 1355–1359. doi:10.1109/LGRS.2013.2241390

Zhang, X., *et al*., 2014. Data-driven synthetic modeling of trees. *IEEE Transactions on Visualization and Computer Graphics*, 20 (9), 1214–1226. doi:10.1109/TVCG.2014.2316001