

Article

# Hierarchical Modeling of Street Trees Using Mobile Laser Scanning

Jingzhong Xu <sup>1,\*</sup> , Jie Shan <sup>2</sup>  and Ge Wang <sup>1</sup> 

<sup>1</sup> School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China; 2016282130085@whu.edu.cn

<sup>2</sup> Lyles School of Civil Engineering, Purdue University, 550 Stadium Mall, West Lafayette, IN 47907, USA; jshan@purdue.edu

\* Correspondence: jz\_xu@whu.edu.cn

Received: 28 May 2020; Accepted: 17 July 2020; Published: 19 July 2020



**Abstract:** This paper proposes a novel method to reconstruct hierarchical 3D tree models from Mobile Laser Scanning (MLS) point clouds. Starting with a neighborhood graph from the tree point clouds, the method treats the root point of the tree as a source point and determines an initial tree skeleton by using the Dijkstra algorithm. The initial skeleton lines are then optimized by adjusting line connectivity and branch nodes based on morphological characteristics of the tree. Finally, combined with the tree point clouds, the radius of each branch skeleton node is estimated and flat cones are used to simulate tree branches. A local triangulation method is used to connect the gaps between two joint flat cones. Demonstrated by street trees of different sizes and point densities, the proposed method can extract street tree skeletons effectively, generate tree models with higher fidelity, and reconstruct trees with different details according to the skeleton level. It is found out the tree modeling error is related to the average point spacing, with a maximum error at the coarsest level 6 being about 0.61 times the average point spacing. The main source of the modeling error is the self-occlusion of trees branches. Such findings are both theoretically and practically useful for generating high-precision tree models from point clouds. The developed method can be an alternative to the current ones that struggle to balance modeling efficiency and modeling accuracy.

**Keywords:** laser point clouds; skeleton; tree model; triangulation

---

## 1. Introduction

Three-dimensional tree models have great significance for cities, tourism, and ecological landscapes, both in physical and virtual worlds. However, due to the wide variety of species, shape differences, and structural complexity, constructing realistic tree models that effectively depict realistic tree geometry and topological information is difficult and therefore continues to be a prime focus of ecology, agriculture, forestry, computer graphics and remote sensing researchers. Over the last few decades, different types of methods have been proposed for tree modeling, including rule-based [1,2], sketch-based [3,4] and image-based [5,6] methods. Among them, the rule-based methods are the only ones that use a computer to generate tree models based on the physical growth process of trees. Due to limited consideration on the actual shape of individual trees, such methods struggle with modeling trees of complex structures. The sketch-based methods need professional tree outline drawings as input. Tree models are then created according to the growth laws of trees or the models in a tree library. The image-based methods are based on actual multiview images using image segmentation and image matching techniques. Unfortunately, these kinds of methods may not accurately represent the actual tree structures, and the modeling results are often not sufficient to extract the tree structure.

Recently, laser scanning technology has become an effective means for modeling trees because it can acquire 3D point clouds of trees directly. Although airborne and space-borne laser scanning technologies can acquire the canopy points of forestry, supplying a detailed tree vertical structure is nearly impossible. As such, high fidelity tree modeling methods therefore are mostly based on terrestrial laser scanning data. Some methods [7,8] use the alpha-shape algorithm to reconstruct the canopy first, from which tree models are then reconstructed by combining the trunk diameter and tree height information. Although this approach can determine the shape of a tree, distinguishing the tree branches based on the modeling result is difficult. Therefore, some researchers have begun developing modeling methods based on tree skeletons. For example, Bucksch et al. [9,10] proposed a graph-based point cloud skeletonizing algorithm, which first divides the point clouds into multiple voxels and extracts a graph from the points in the octree cells. The graph nodes are removed or merged through a series of rules and the final tree skeletons are generated. This method can process a large number of point clouds in linear time complexity, but the resulting skeletons are dependent on the size of the octree cells. Further work is required to confirm the topological correctness of the skeletons. A few studies [11–13] developed a tree skeleton extraction method using a constrained Laplacian smoothing algorithm. This method directly deals with the point clouds of trees and is not sensitive to noise. Nevertheless, the quality of the skeleton lines depends on the chosen Laplacian weighting matrix. The method is less efficient for complex tree structures. Verroust and Lazarus [14] proposed a method for extracting tree skeleton lines from point clouds using a level set method. The method first uses the neighborhood of each point to form a neighborhood map and then calculates the geodesic distances from each point to the root node. Points having the same geodesic distance are located in the same level set; the center points of all the adjacent level sets are connected to obtain the final skeleton of the point clouds.

Based on prior knowledge of trees, the center points of the clusters in different layers form the skeleton nodes by a hierarchical clustering method [15]. Based on this, the points that are not connected to the main skeleton are considered to be leaves. Then the Hermite curves and Murray criterion are used to split and optimize the skeleton nodes. Finally, 3D models of the trees are generated where the trunks are simulated by triangulation and the leaves are simulated by a quadrilateral texture projection. This method can only deal with trees with simple branches and structures and is difficult to accurately simulate branches with complex skeletons. Another method [16] obtains tree skeleton points by a hierarchical clustering method similar to the one in [15]. Considering the fact that tree branches have obvious grading characteristics, the authors of [17] rank the skeleton points into different levels, and use a generalized cylinder to create a 3D branch model. This method can reconstruct a tree model automatically. However, since the skeleton nodes are obtained by the hierarchical clustering method, there is still room for improvement. In [18], three steps are involved to generate tree models. First, the point clouds are segmented into clusters by the k-means algorithm, followed by a cylinder detection method to subdivide the clusters. Second, an adjacency graph is built from the clusters by detecting the neighborhood information for each cluster. The final step uses the shortest path method to determine the tree skeleton and B-spline to model the tree. This kind of method can deal with trees with complex branches but cannot reconstruct trees with leaves. Livny et al. [19] proposed a tree reconstruction method based on global optimization of tree skeletons. This method first generates a Branch Structure Graph (BSG) from the point clouds. Then, the Dijkstra algorithm is used to create a tree graph by extracting the minimum weight path. Finally, the tree growth method by spatial distribution constraints is iteratively used to obtain a smooth tree skeleton. This method can deal with trees that have leaves and generate tree models automatically, however, it requires point clouds with high density and its modeling results are prone to local distortion due to the iterative smoothing process.

Due to the characteristics of LiDAR scanning technology, occlusions often occur during the data collection. In order to make up for the deficiencies of LiDAR point clouds, some scholars have successfully integrated aerial images to reconstruct building models [20,21]. However, traditional

photogrammetry methods can only obtain surface information instead of vertical structure information of trees. Scholars begin to use the prior knowledge of trees to directly reconstruct the tree model from the point cloud to repair the occlusion area. For example, Wang et al. [22,23] proposed a method to reconstruct tree models from incomplete terrestrial laser scanning (TLS) point clouds. They firstly use the intensity information to distinguish tree branches from the segmented tree point clouds. Then a Distance Minimum spanning tree (DMst) algorithm is used to extract the approximate tree skeleton. Finally, a Structure Aware Global Optimization (SAGO) approach is designed to recover missing data by employing a global optimization method. This method is less sensitive to incomplete data, and can model trees that have large occluded regions. However, the skeletons in the regions where the branches and twigs are completely occluded still cannot be fully extracted. Mei et al. [24] integrate the advantages of the L1-median and the Minimum Spanning Tree (MST) to improve the modeling result from incomplete TLS point clouds. This method can extract tree skeletons from the optimized point cloud automatically without prior assumptions on the shape geometry or topology. But the modeling result still has similar problems in areas where the branches and twigs are completely occluded.

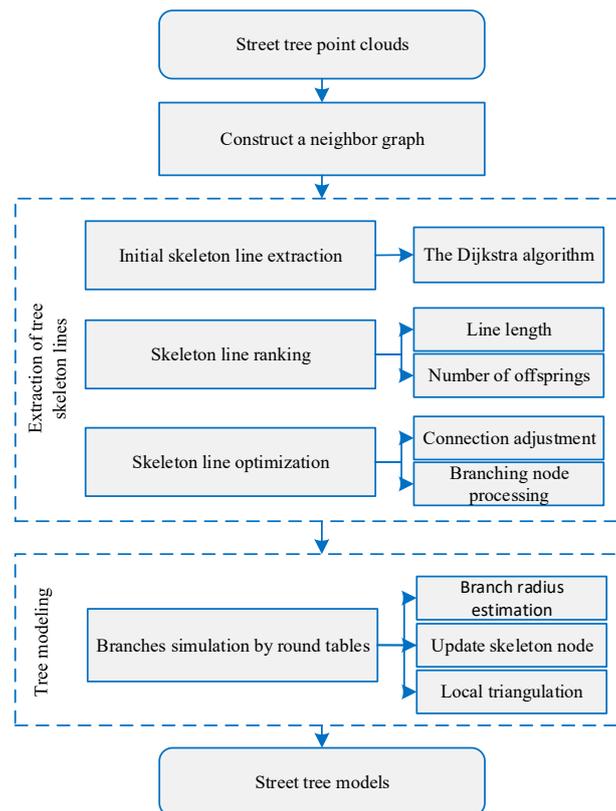
In summary, despite the fact that advanced terrestrial laser scanning technology can provide detailed tree point clouds, a few challenges still remain for reconstructing tree models, mostly due to the variety of species and complex shapes of trees, uneven laser cloud density, tree leaf occlusion, and other factors. To fill this gap, this paper proposes an automatic reconstruction method of trees from mobile laser scanning (MLS) point clouds by a skeleton ranking method. Based on the neighborhood graph of a tree point cloud, our method uses the Dijkstra algorithm to obtain the initial skeleton lines of trees. To obtain natural, smooth tree branches, the skeleton lines are ranked according to the weight of each skeleton line. Then, the skeleton node and their connectivity are adjusted by a length and direction priority rule. The final 3D tree models are simulated by flat cones. The main contribution of this work is the method for reconstructing tree models hierarchically, including tree skeleton ranking, optimization, and tree branch simulation to generate tree models with different levels of details. We also demonstrate that a hierarchical tree modeling needs to achieve an optimal balance between modeling accuracy and modeling fidelity.

The remainder of this paper is organized as follows: Section 2 describes the proposed methodology of tree modeling from laser point clouds, including the extraction of tree skeleton lines and tree modeling process. Section 3 analyses the performance of the proposed method. Section 4 presents quantitative evaluation and discussions, while Section 5 describes the conclusion and future work.

## 2. Methods

Considering the fact that point clouds of trees are discrete and irregular, it is difficult to generate tree models directly from point clouds. Therefore, the aim of the method proposed in this paper is to construct tree models in the following steps: tree skeleton line extraction, skeleton line optimization, and tree model generation (see Figure 1).

The MLS point cloud is firstly filtered by the adaptive TIN models method [25] to get ground points and nonground points. The latter is then subject to a dimensionality analysis to further remove planar points [26]. The remaining points are then projected to grids to remove objects of low density and low height. Finally, individual trees are segmented by a binary connected component labeling method [27]. In the following sections, these trees will be modeled one by one. Considering that the intensity measurements of MLS are easily affected by the range and incidence angle [28,29], point cloud intensity cannot always effectively distinguish between leaves and trunks, the proposed method is implemented directly on points of individual street tree in this paper. In order to ensure the effectiveness of modeling results, street trees data need to contain point clouds from tree branches.



**Figure 1.** Workflow of the proposed hierarchical tree modeling approach.

## 2.1. Extraction of Tree Skeleton Lines

### 2.1.1. Initial Skeleton Lines Extraction

For tree point clouds, the path to the root point through the center of the trunk is usually shorter than the path through the surface point of the trunk to the root point. That is, for the end point of a branch, among all the paths that connect its neighboring points to the root of the tree, the shortest path can be regarded as the skeleton line of the branch. Therefore, the method proposed in this paper uses the Dijkstra algorithm to extract the initial skeleton lines of trees. The tree skeleton line extraction process is as follows:

- (1) Individual tree points are organized into a connected undirected graph  $G(N, E)$ , in which  $N$  represents the point cloud of the tree and  $E$  represents the edge between points. The Euclidean distance between two points is defined as the weight of the edge. When two points are not connected, its weight is initialized as infinite.
- (2) Take the point with the lowest height as the root node of the tree and an unprocessed point as the target node. Mark the root node as the parent node of the tree. Then, set the distance of the edge to root node as zero and initiate the distances of edges between other points as infinity. Since tree point clouds are surface points of the tree, the root node found by the lowest height is a ‘pseudo’ root node of the tree, which is not the center position of the root. In order to eliminate this effect, the subsequent processing will use the original tree point to fit the true position of the root node.
- (3) Find the child node unmarked that is closest to the parent node in the connected graph  $G(N, E)$  and mark it as the current node.
- (4) Find all the next child nodes for the current node and then compare the sum of the distance from the root node to the current node and the distance from the current node to the next child node with the distance from the root node to the next child node. If the former distance is smaller than the latter one, update the path and repeat step (3), otherwise go to step (5).

- (5) Find the next unmarked child node that is closest to the root node and mark it as the current node, then repeat step (4); if the target node is reached, then the shortest path between the root node and the target node is obtained.
- (6) Repeat step (2) to step (5), when each point of the tree has one shortest path to the root node, then the shortest path search is completed.

### 2.1.2. Tree Skeleton Line Ranking

Considering that tree branches have obvious grading characteristics and ranking skeleton lines into different levels can maintain the shape of trees with high precision [13,14], this paper demonstrates how the proposed method facilitates skeleton line extraction through optimization. Based on the initial tree skeleton lines, the ranking procedure is as follows:

(1) Identify the main skeleton line (level 0) (Algorithm 1).

(a) First, mark branch nodes. Since each node has one shortest path to the root node, there are multiple skeleton lines in a tree. Before ranking tree skeleton lines, branch nodes of the tree need to be marked first:

Initialize all skeleton nodes as unprocessed and traverse each skeleton line of the tree. If the node of the skeleton line is unprocessed, then mark it as processed and continue traverse to the next node of the line. If the node is processed, then mark it as a branch node and stop traversing this skeleton line.

If all skeleton lines of the tree are traversed, stop marking branch nodes.

(b) Next, update valid skeleton lines of the tree: if all nodes of one skeleton line (including branch points) are unprocessed, then update the skeleton line to be valid and mark nodes of the skeleton line as processed. If one branch node of the skeleton line has been processed, then replace the skeleton line by the path from the starting node to the branch node of the line and update the skeleton line to be valid.

If all skeleton lines are updated, stop the processing.

(c) Finally, traverse the updated skeleton lines, and find the skeleton line which contains the root node. Mark the branch node that has the shortest distance to the root node as the level 0 node, so the path from the level 0 node to the root node is the main skeleton line (level 0).

(2) Search the offspring skeleton lines iteratively (level #).

(a) Count the total levels of the tree skeleton lines.

(b) Set the 0-level branch node as the end point, traverse the updated skeleton line. If the skeleton line has one branch node with the shortest path to the 0-level branch node, then mark the branch node as level 1 node. Because there may be many 1-level branch nodes, mark each path between the 1-level node and the end point to get all level 1 skeleton lines.

(c) Set one 1-level branch node as the end node and traverse the updated skeleton lines again. If the current skeleton line contains the branch node, search the branch node with the shortest path to the end node and mark it as the level 2 node. Mark the path between the 2-level node and the end node as a level 2 skeleton lines. When all 1-level branch nodes are processed, stop 2-level branch nodes search and level 2 skeleton lines marking.

(d) Set one 2-level branch node as the end point, use the same method to search the next level branch nodes and mark the next level branches.

The above steps are carried out iteratively until all branch nodes are marked and the skeleton line ranking process is ended.

Considering that tree leaves are usually located at the ends of the branches, this method takes end nodes of initial skeleton lines as leaf nodes, and these nodes are not taken in the subsequent skeleton line optimization and tree model reconstruction.

---

**Algorithm 1:** The main skeleton line ranking.

---

**Input:** S, initial tree skeleton lines; r, tree root node index; Node, tree skeleton node;  
**Output:** M, the queue of the main skeleton line;  
**Initialize:** branch node array b\_mark[][] as unprocessed;  
 Su, updated tree skeleton lines;  
 1-level branch node stack L1;  
**for each** skeleton line in S **do**  
**for each** node i **do**  
**if** b\_mark[i] unprocessed **then**  
 set b\_mark[i] processed;  
**else**  
 set b\_mark[i] branchnode;  
 break;  
**end if**  
**end for**  
**end for**  
**for each** skeleton line in S **do**  
**for each** node i **do**  
 Su<-i;  
**if** b\_mark[i] is branchnode **then**  
 Break;  
**end if**  
**end for**  
**end for**  
**for each** skeleton line in Su **do**  
**for each** node i **do**  
**if** b\_mark[i] is branchnode and i has minimum path to r **then**  
 minID=i;  
**end if**  
**end for**  
 L1<-minID;  
**end for**  
**for each** skeleton line in Su **do**  
**for** j>= SearchID(L1) & j<= r **do**  
 M<-Node(j);  
**end for**  
**end for**

---

### 2.1.3. Skeleton Line Optimization

Since laser scanning points are randomly distributed on the surface of tree trunks, the skeleton line obtained by the shortest distance rule usually has such problems as disordered branches and unreasonable directions. In order to overcome these problems and get more natural and smooth branch models, the proposed method further improves the tree skeleton lines by adjusting the skeleton line connection and changing the branch points. The optimization steps (connection adjustment and branch node processing) are then conducted.

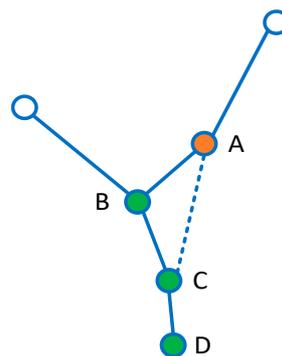
#### (1) Connection adjustment

Considering that actual tree branches are connected naturally and smoothly, the proposed method uses a principle of direction priority to adjust the connection of the skeleton nodes. As shown in Figure 2, A is a node to be processed, its parent node is B, C is the parent node of B, and the parent node of C is D. Because the distance between A and B is smaller than the distance between A and C, B is regarded as the parent of A by the shortest path method. However, the axial angle between B and A is too large and does not conform to the smooth connection rule. In this paper, we consider that C is

the true parent of A, which should be connected with C instead of B according to the shortest distance. Then, C becomes the parent of A, and B becomes the brother node of A. Similarly, it can be judged whether the axial angle between C and A (that is, the angle between direction DC and CA in Figure 2) is in keeping with the angle rule. If it does not meet the angle criterion, the true parent of A has yet to be found according to the method.

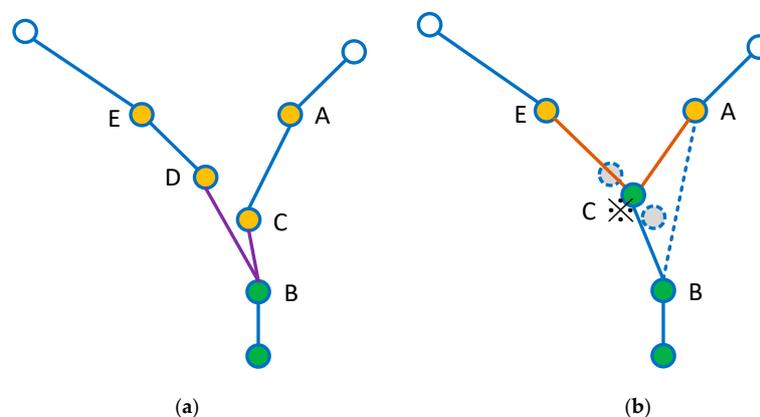
## (2) Branch node processing

Because the trunk of a tree is thicker than the branches, it has more point clouds than the branches. Therefore, the initial skeleton lines usually contain some unnatural connections between the skeleton nodes.



**Figure 2.** Schematic of skeleton line connection adjustment (A is a node to be processed and the parent node of A need to be adjusted from B to C).

To solve this problem, the proposed method improves the skeleton connection effect at the branching area by merging the skeleton lines according to an angle criterion. As shown in Figure 3, C is a child node located at a branch, and one of its brother nodes is D. B is a parent of C and D. A is a child of C, and E is the child of D. Because the axial angle between C and D is too small (the angle between direction BC and BD in Figure 3a), we consider that B, C and D are actually located on the same branch. As such, the skeleton line of B, C and B, D should be merged. Here, the average position  $C'$  between C and D is generated as the merging result. Then, the new node  $C'$  becomes the parent node of A, and the axial angle between the node A and E (the angle between direction  $C'A$  and  $C'E$  in Figure 3b) needs to be checked. If the angle is larger than the angle threshold  $\Delta\theta$ , the parent node of A needs to be adjusted to B, where the angle threshold  $\Delta\theta$  can be obtained by multiple experiments. To follow the same principle, the axis angle between  $C'$  and A needs to be checked in the next step.



**Figure 3.** Schematic of branching node processing: (a) Branching node before processing; (b) Branching node after processing. C and D need to be merged by the angle criterion. The new node  $C'$  is generated by the average position of C and D and becomes the parent node of A, which needs to be adjusted for the angle condition. The parent node of A is finally changed from  $C'$  to B.

## 2.2. Tree Modeling

### 2.2.1. Branch Radius Estimation

Based on the skeleton line of the trees, the radius of the skeletal nodes can be used to extend the branches to generate tree models. Due to the uneven point density and tree branch occlusion, it is difficult to accurately obtain all the radii of the skeleton nodes directly from the laser point clouds. To address this issue, two cases are considered to estimate the radius for the skeleton node. If the parent node has only one child node, then the ratio between the child radius and parent radius is taken to be proportional to the ratio of the supported lengths for the child and parent nodes according to the WBE model [30]. The radius of the child node can be estimated by the following formula:

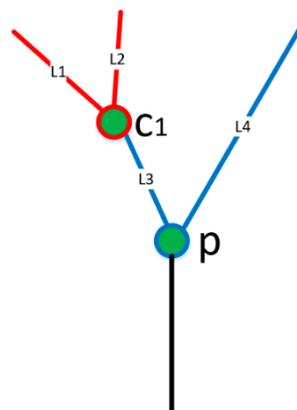
$$R_c = R_p \times \left(\frac{L_c}{L_p}\right)^{1.5} \quad (1)$$

where  $R_p$  is the radius of the parent node,  $R_c$  is the radius of the child node,  $L_p$  is the total branch length supported by the parent node, and  $L_c$  is the total branch length supported by the child node.

If the parent node has more than one child, each child is assigned a radius based on an extension of Murray's rule [31]. The child branch radii are scaled by the estimation based on multiple tests that children supporting more weight are thicker. For a parent node with  $n$  child branches, the radius of the  $i$ th child node can be estimated by the following formula:

$$R_{c_i} = R_p \times \left(\frac{L_{c_i}}{\sum_{i \in n} L_{c_i}}\right)^{0.5} \quad (2)$$

where  $R_{c_i}$  is the radius of the  $i$ th child node  $\{R_{c_i} | i \in n\}$ , and  $L_{c_i}$  is the total branch length supported by the  $i$ th child node. As shown in Figure 4, the radius of the node C1 is  $R_{C1} = R_p \times \left(\frac{L1+L2}{L1+L2+L3+L4}\right)^{0.5}$ .



**Figure 4.** Schematic diagram of the parent–child node radius ( $P$  is the parent node of the node  $C1$ ;  $L1$ ,  $L2$ ,  $L3$  and  $L4$  are the lengths of each skeleton line).

### 2.2.2. Update the Position and Radius of the Skeleton Node

Because the skeleton node obtained by the above method is generated based on surface points of the tree, which are not necessarily located at the center of the tree branches, reconstructing a tree model directly by these nodes will cause a positional deviation from the real tree. In order to improve the position accuracy of the skeleton node, this study uses a hierarchical clustering method to update the position of the skeleton node. The detailed procedure is as follows:

- (1) Update the level 0 skeleton line:

First, search for all skeleton nodes between the root node of the tree and the 0-level branch node. Taking the height  $Z_i$  of a node as a reference, we can get the points whose height is between  $(Z_i - \Delta h, Z_i + \Delta h)$  from the original tree point cloud; ( $\Delta h$  set to 0.1 m in our study).

Then, use the region grow method to cluster these points. Find the cluster closest to the current node, and take half of the maximum length of the cluster as the radius. If this radius and the current node radius satisfies Equation 3, then the radius and position of the skeleton node are updated using the cluster radius and cluster center position respectively; otherwise, no update occurs.

$$|R'_C - R_C| / \max(R'_C, R_C) < \varepsilon \quad (3)$$

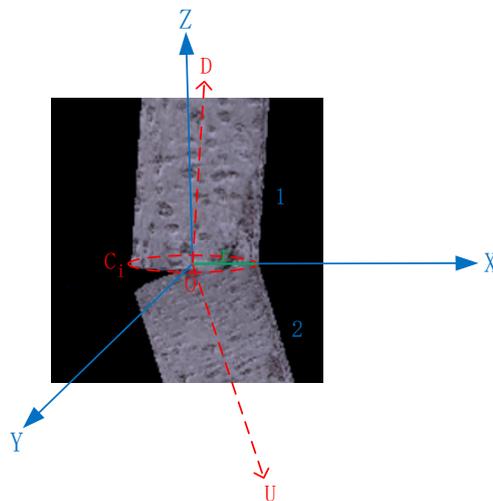
where  $R'_C$  is the radius of the point cluster,  $R_C$  is the estimated value of the node, and  $\varepsilon$  is the threshold ( $\varepsilon$  is set to 0.15). After all the skeleton nodes are processed, the skeleton line is updated.

#### (2) Update skeleton lines of other levels

Use the same method to update the radius and position of other level skeleton nodes until all skeleton lines have been updated.

### 2.2.3. Tree Branch Modeling

After obtaining the radius of the skeleton nodes, the tree model is generated by a flat cone fitting method. By drawing flat cones according to the skeleton line connection sequence, the model of the tree branch segment can be simulated. Different results can be generated by adjusting the radius of each flat cone. Although the upper and lower sections of the flat cone are supposed to be perpendicular to the axis of the flat cone, there would be a gap at the joint of the two flat cones (as shown in Figure 5) if the axial of the two sections is not in the same direction. To improve the modeling results, a local triangulation method is used to fill these gaps.



**Figure 5.** Schematic of local triangulation in the gap between segments of a tree trunk. The point of  $C_i$  needs to be interpolated around the circle of the flat cones.

The local triangulation method needs to calculate the coordinates of the points on the circumference of the bottom surface of the flat cone. As shown in Figure 5, the node  $O$  is the connection node between flat cone 1 and flat cone 2 with coordinates  $(x_o, y_o, z_o)$ , and  $r$  is the radius of the node. The center of the bottom of flat cone 1 and the top of flat cone 2 are  $D$  and  $U$ , respectively. The coordinates of circle point  $C_i$  can be calculated at equal intervals by the following formula:

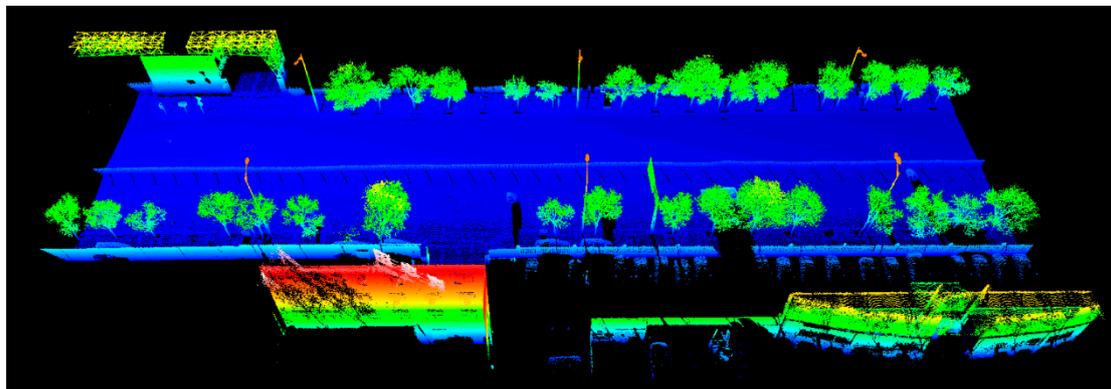
$$\begin{bmatrix} x_{c_i} \\ y_{c_i} \\ z_{c_i} \end{bmatrix} = \begin{bmatrix} x_o \\ y_o \\ z_o \end{bmatrix} + \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \alpha & \sin \alpha \\ 0 & -\sin \alpha & \cos \alpha \end{bmatrix} \times \begin{bmatrix} \cos \beta & 0 & \sin \beta \\ 0 & 1 & 0 \\ -\sin \beta & 0 & \cos \beta \end{bmatrix} \times \begin{bmatrix} r \cos \theta_i \\ r \sin \theta_i \\ 0 \end{bmatrix} \quad (4)$$

where  $\theta_i$  is the angle between the direction  $OC_i$  and coordinate axis  $Z$  ( $i = 0, 1, 2, \dots, n$ ),  $\alpha$  is the angle between the direction  $OD$  and coordinate axis  $Y$ ,  $\beta$  is the angle between the direction  $OD$  and coordinate axis  $X$ .

In the same way, the circumferential points of the lower section of flat cone 2 are calculated, and then local triangles can be generated by using all these circumferential points to fill the gap between the two flat cones.

### 3. Tests and Results

The study area is a road section of approximate 127 m located in an urban business district in Beijing, China, which has more than 30 trees. This dataset was acquired using the SSW-IV [32] mobile mapping system with a Riegl laser scanner. The mean laser point density is 678 points/sq. m and the number of laser points in the scene totals 2 million. Figure 6a shows the point cloud of the dataset. There are street trees, buildings, cars, fences, and pedestrians in the test area. Figure 6b shows the point clouds of street trees obtained by the point cloud filtering and segmentation process. All of the experiments of this study have been carried out using a computer with an Intel core i7-6600U 2.81-GHz processor with an 8.0-GB RAM running the VS2010 C/C++ language.



(a) Point clouds of the study area colored by height.

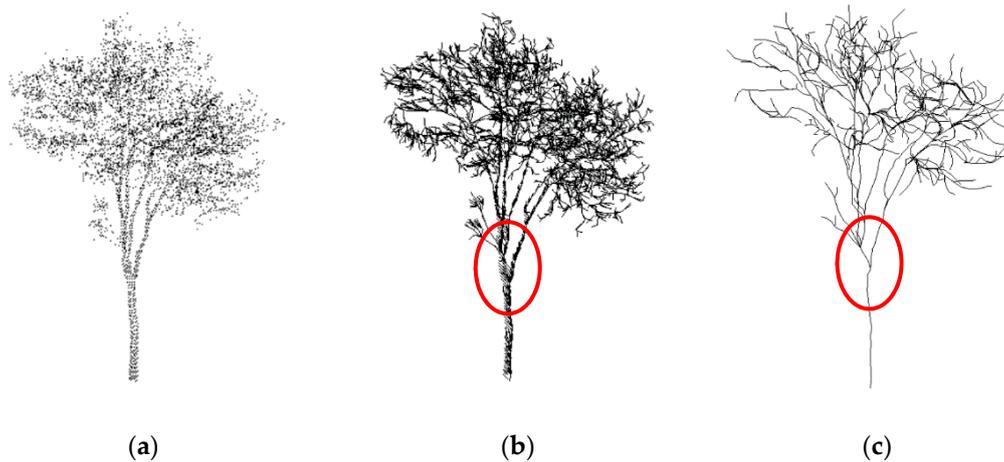


(b) Segmented street trees.

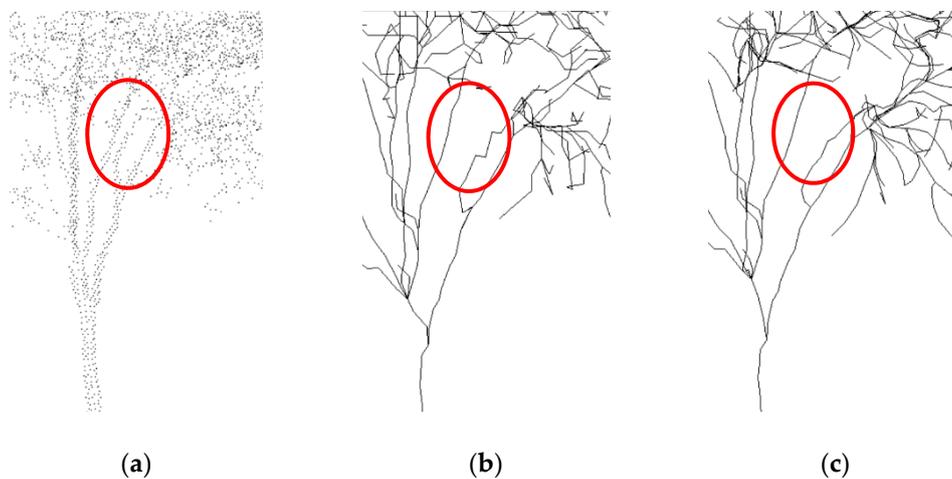
**Figure 6.** Point clouds of the study and segmented street trees.

For convenience of description in this paper, one street tree is shown in this paper to illustrate the procedure of tree modeling. Figure 7a is the original point cloud of a tree, while Figure 7b is the initial skeleton lines extracted from the original points by the Dijkstra algorithm. It is seen that the initial skeleton line roughly reflects the shape and structure of the tree, along with some redundant lines and connection errors in some parts of the tree (as shown in the red circle of the Figure 7b) that need to be optimized.

In addition to Figure 7c, the effects of skeleton optimization are also shown in Figure 8, where the partial skeleton lines are illustrated as a before and after comparison of optimization. As shown by the ovals in Figure 8, some angles between the skeleton segments are too large and seem to be unreasonable. This might be caused by the shortest distance rule. After the optimization process, the angle between the skeleton segments becomes smaller, and the transition is more natural.



**Figure 7.** Skeleton line extraction from tree point clouds: (a) Tree point clouds; (b) Initial tree skeleton; (c) Optimized tree skeleton.

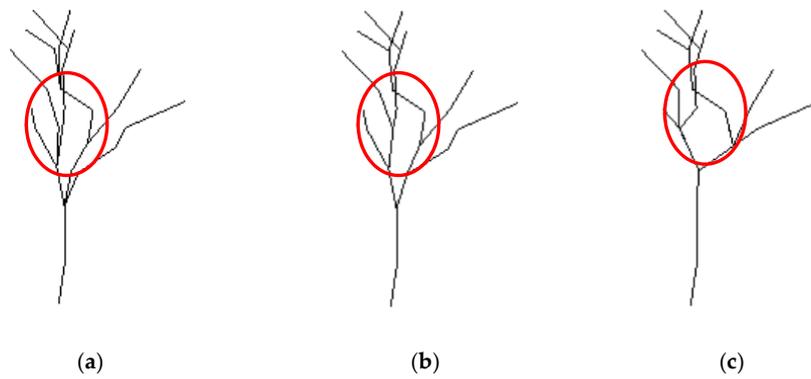


**Figure 8.** Comparison of skeleton branch before and after connection adjustment: (a) Tree points; (b) Before optimization; (c) After optimization.

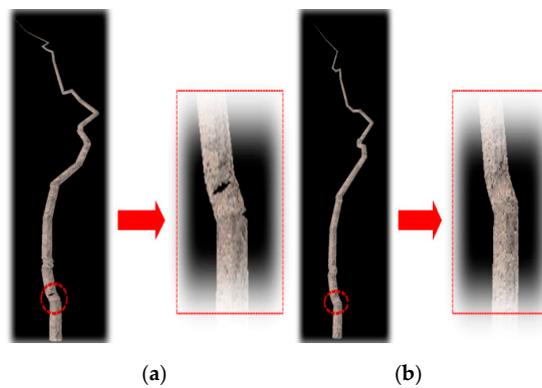
Figure 9 is a comparison of the branching node processing results, in which Figure 9a is a skeleton line before processing, Figure 9b is the skeleton process results at an angle threshold of 15 degrees, and Figure 9c is the results at an angle threshold of 30 degrees. It can be seen that the small branches were merged into the main branches. However, in Figure 9c some important branches that represent tree shapes also were merged due to the larger angle threshold. In order to keep the shape of the original tree and preserve the modeling accuracy, the angle threshold was set to 15 degrees after many repetitions in the experiment. The optimized results of the skeleton lines are shown in Figure 9c. Compared with Figure 9b, the optimized effect is apparent in the tree structure and the tree branch connection is smooth, which is in line with the actual shape of the tree.

Figure 10 shows the tree branch modeling results before and after using the local triangulation method. Through this optimization, the model becomes more realistic and natural.

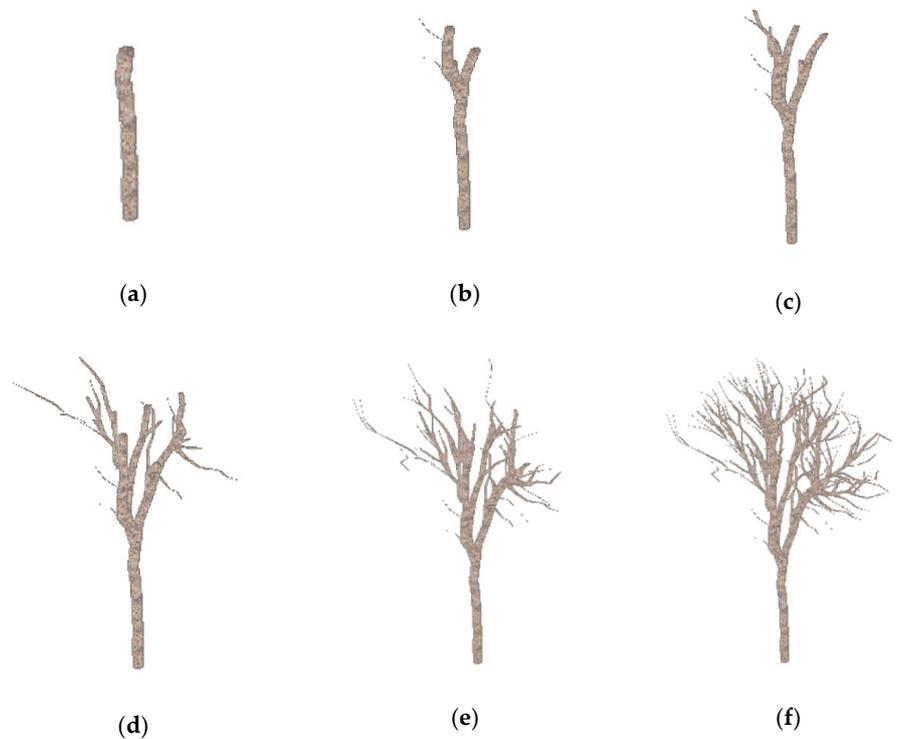
Figure 11 depicts the tree reconstruction results by different levels of detail from the tree point clouds. It can be seen that the model is able to restore the original tree shape realistically, and the tree branch connection also conforms to the natural tree growth rule. In addition, the tree is modeled hierarchically according to the levels of the tree branches, wherein (a)–(f) are the model results at branches level 0 to 5, respectively. The results show that the proposed method can hierarchically display the results to meet different rendering requirements in different scenarios.



**Figure 9.** Comparison of skeleton line processing by different angle condition: (a) is skeleton lines before processing; (b) is the branch node processing result by  $\Delta\theta = 15^\circ$ ; and (c) is the branch node processing result by  $\Delta\theta = 30^\circ$ , which has noticeable difference with the original tree shape.



**Figure 10.** Branch modeling results before and after using the local triangulation method: (a) branches model before gap filling; (b) branches model after gap filling.



**Figure 11.** Tree reconstruction results at different levels of detail: (a) level 0 branches; (b) level 1 branches; (c) level 2 branches; (d) level 3 branches; (e) level 4 branches; and (f) level 5 branches.

Figure 12 shows the modeling results of all of the street trees in the study area. Compared to tree point clouds in Figure 6a,b, the street tree models shown in Figure 12 are realistic.

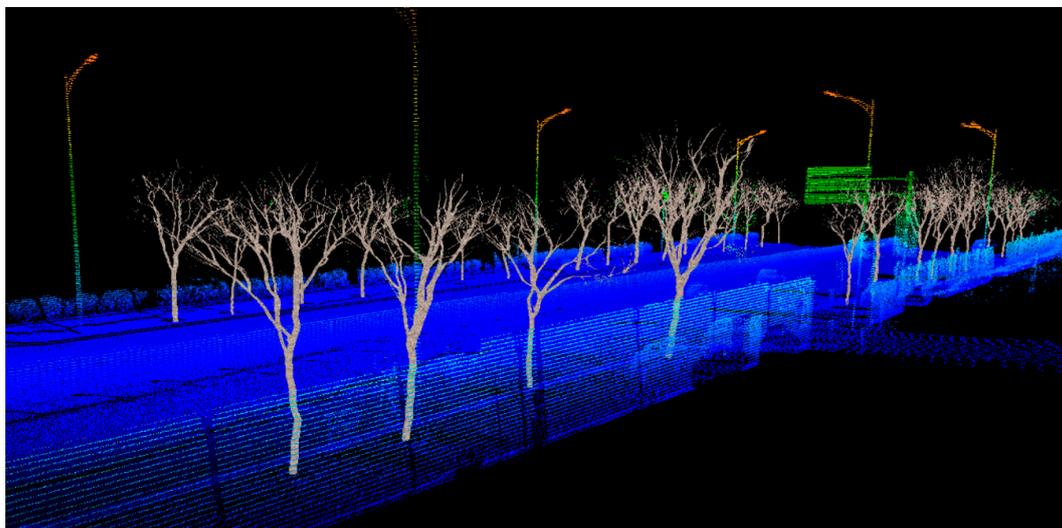


Figure 12. Modeling results of all street trees in the study area.

#### 4. Evaluation and Discussion

In order to quantitatively evaluate the accuracy of the reconstruction result, the original tree point clouds are overlaid on the tree models (Figure 13) to demonstrate the accuracy of the tree model results.

Considering that different levels of tree branches have different levels of importance, the tree model's accuracy is calculated according to the different branch levels. The accuracy of level 0–6 branches was calculated in the experiments. The detailed process is as follows: find all of the neighboring points of the current node, calculate the distance from each neighboring point to the current node, and take the difference between the average distance of all neighboring points and the current node radius as the error, which is shown below:

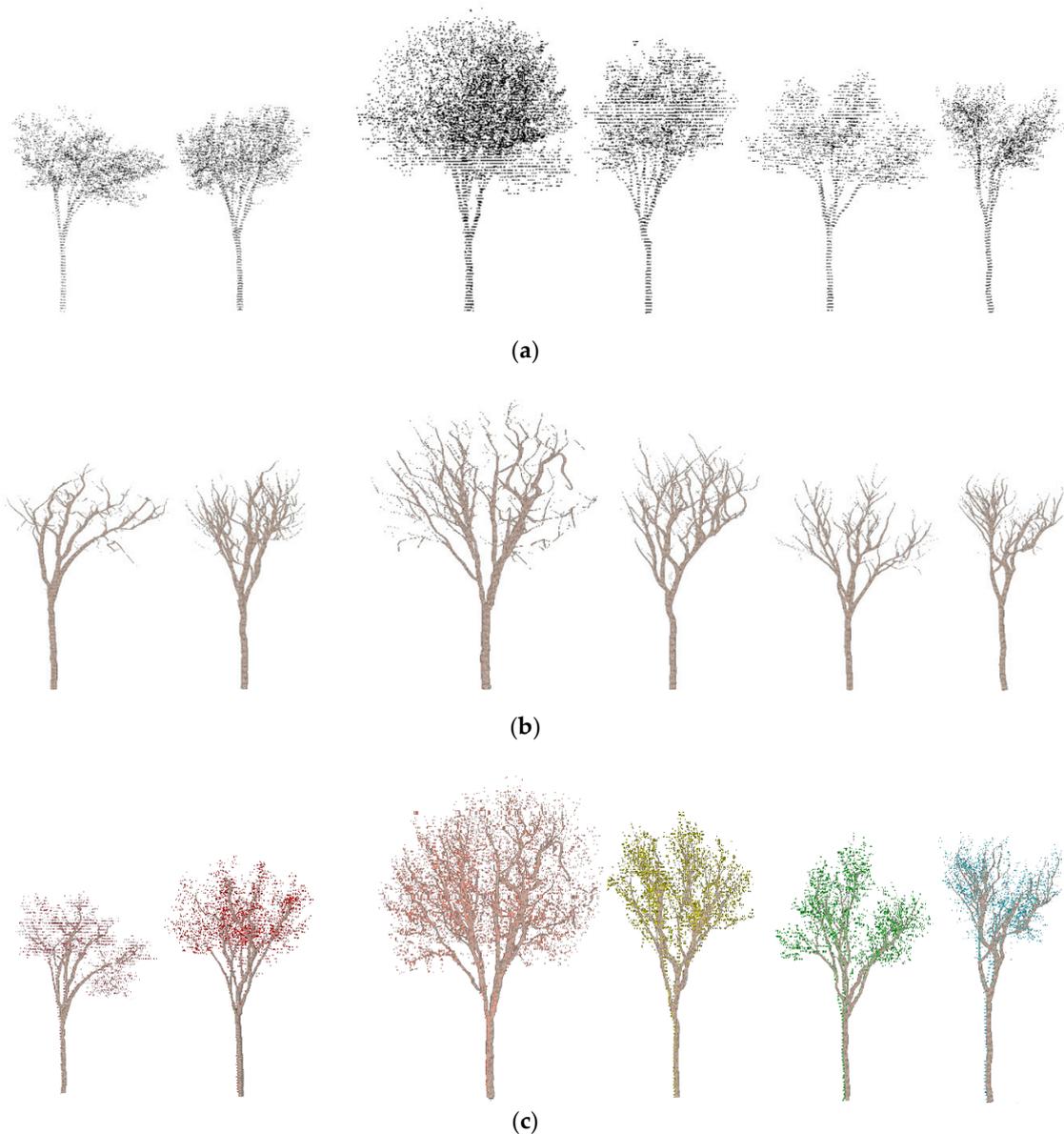
$$e_j = \frac{1}{n_j} \sum_{i=1}^{n_j} |d_i - r_j|, \quad i = 1, 2, 3, \dots, n_j; \quad j = 1, 2, 3, \dots, m \quad (5)$$

where  $d_i$  is the distance of neighboring point  $i$  to the current node  $j$ ,  $r_j$  is the branch radius of the current node  $j$ .

The mean error of all the skeleton nodes on the stem is taken as the total branch error. Table 1 shows information of tree point spacing, computation time and modeling errors in different levels of detail. It can be seen from Table 1 that from level 0 to level 6, all of the tree branch errors are within centimeter level. Tree branches at level 0 have the highest accuracy, and the level 0 branch of tree No. 2 have the smallest modeling error (8.1 mm). As the branch level increased, i.e., with more details, the model error increases. The error at level 0 branch is the smallest, and the error increases rapidly as the branch level increases. The error in level 1 branch is larger than at level 0, and the same pattern repeats and is observed between level 4 and level 5. When the branch levels reach 5 and 6, the tree error tends to be stable. It is consistent with the actual point cloud distribution characteristics: tree trunks usually occupy more point clouds and can be represented in details. Therefore, tree trunks have the highest accuracy. Tree branches usually lack details due to occlusion so the modeling results vary by the actual locations of the tree branches. In addition, the modeling errors of tree No. 1, No. 2, No. 4, No. 5, and No. 6 stabilize or decrease when level 5 is reached while the errors of tree No. 3 continues to increase after level 5 is reached. This result shows that the structure of tree No. 3 is more complicated than the others. It is therefore necessary to preserve a higher level of branches to achieve

the same accuracy when modeling. Likewise, detailed point clouds are needed to ensure modeling accuracy for trees with a complex structure.

Terrestrial laser scanning provides us a new opportunity to precisely model the trees with high level of details. However, due to the sample randomness of laser scanning and the characteristics of tree structures, tree branches with different diameters may be covered with point clouds of different detail levels. Therefore, branches of the same tree can have a different modeling quality. It can be seen from Table 1 that the reconstruction accuracy of the tree model gradually decreases with the increase of the branching level. However, low-level branches are not sufficient for describing a complete tree model. As many branches as necessary must be reconstructed to form the entire tree model. Therefore, this section further analyzes the relationship between tree branch diameter, point cloud spacing, and the modeling accuracy.



**Figure 13.** Street tree point clouds overlaid on a row of selected tree models: (a) tree point clouds; (b) tree model results; (c) tree points overlaid on models.

**Table 1.** Information of tree point spacing, modeling time and modeling errors (Level 0–6).

Tree ID	Point Spacing (mm)	Computing Time (s)	Errors in Different Levels of Detail (mm)						
			L0	L1	L2	L3	L4	L5	L6
1 (dark red)	57.8	16.7	8.7	19.2	26.3	28.4	30.9	30.8	29.5
2 (red)	46.9	9.1	8.1	18.1	25.6	28.1	31.4	31.8	28.5
3 (orange)	45.9	8.1	10.2	18.3	28.4	29.7	33.9	37.6	39.5
4 (yellow)	50.4	10.6	10.5	19.8	27.3	29.9	32.2	34.1	28.4
5 (green)	57.5	16.2	11.4	21.4	26.9	28.5	31.6	33.9	29.8
6 (blue)	56.2	15.8	11.8	20.1	31.5	35.7	36.3	35.7	32.9

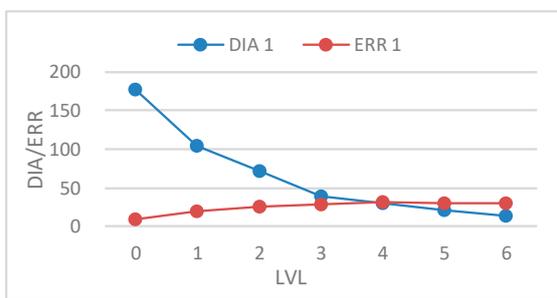
Different point cloud density (point spacing) has a different effect on target modeling [33–35]. Even for trees in the same scene, due to the difference in tree structure, there would be some differences in point cloud density among trees. Therefore, it is necessary to further analyze the relationship between the tree model accuracy and the point spacing, and the model error and the branch diameter. Table 1 also shows the average point spacing of the point clouds for six trees. It can be seen from the table that although the tree point clouds are collected under the same conditions, the average point distance of the tree point clouds still has a certain degree of difference due to the difference in tree structure and growth period. The average point distancing of tree 3 is relatively smaller than the others, indicating a higher density. Figure 14 shows the branch diameter and modeling error trend of each tree. LVL represents the branch level of the tree, DIA represents the diameter of the tree branch, and ERR represents the error of the tree models. It can be seen that, as the detail level of the tree model increases, the diameter of the tree branches gradually becomes smaller and smaller, and the modeling accuracy gradually declines. As such, the branches of different grades of the same tree have different reconstruction accuracy, whereas for trees with similar structures, branches of the same level have similar reconstruction accuracy. Although branch error in the same level fluctuates due to the effect of varying point cloud density, errors of different trees in the same level have similar magnitude. The average error for level 0 is 10.1 cm, and 19.5 cm for level 1, 27.7 cm for level 2, etc. Eventually, most branch errors will stabilize or decrease in level 6 because the end node does not change position in the process of skeleton optimization. It should be noted that the maximum error of all tree models is less than the average point spacing, about 0.61 times the average distance of the point cloud.

Table 2 shows the accuracy of tree models, where the reference is measured manually for node diameter and trunk length. Since high-level branches are difficult to measure effectively, Table 2 only counts the diameter error of the root node (abbreviated as R-node) and the 0-level branch node (abbreviated as 0-node), as well as the length error of the level 0 branch. It can be seen that the node diameter error and the branch length error are both within centimeters. Among them, the diameter error of the root node is greater than that of the 0-level branch node. This indicates that the root of the tree model has a large bias from the measurement, which may be caused by an incomplete point cloud at the root of the tree.

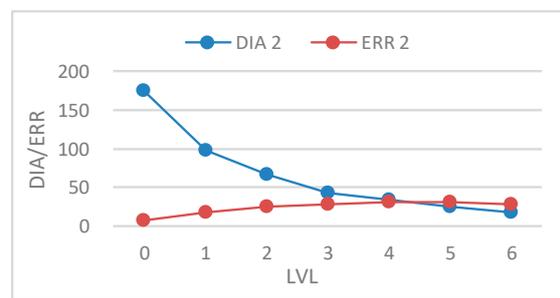
In order to further analyze the sensitivity of the method to the density of tree point clouds, the tree point clouds are randomly sampled, which are then used to generate the tree models. Figure 15 shows the original tree point clouds, thinned point clouds and their modeling results respectively. It can be seen that the tree structure is still well defined in the point clouds that are thinned up to 50%. The reconstructed tree model is very close to the one from the original point clouds. When the point cloud is thinned by 75%, the reconstructed tree model becomes much more generalized; some curved branches are straightened, and the result is quite different from the actual tree. Some branches are even not detected, while some connections of the tree bifurcation are no longer consistent with the actual tree structure. It shows that the proposed method needs tree point clouds with a clear structure to achieve realistic results.

**Table 2.** Errors of tree models (level 0) (mm).

Tree ID	Model			Errors (Reference-Model)		
	Diameter		Length	Diameter		Length
	R-Node	0-Node		R-Node	0-Node	
1	192	164	2406	−23	−16	22
2	214	194	2799	−12	21	19
3	278	266	2499	27	5	−8
4	188	182	2494	−4	−2	22
5	206	194	2516	−8	−16	−48
6	198	180	2729	−16	4	98
Mean				15	11	36
Root Mean Square Error (RMSE):				17	13	47



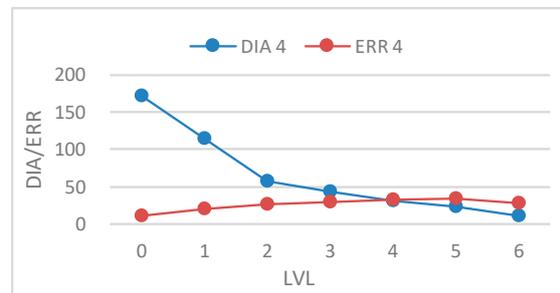
(a)



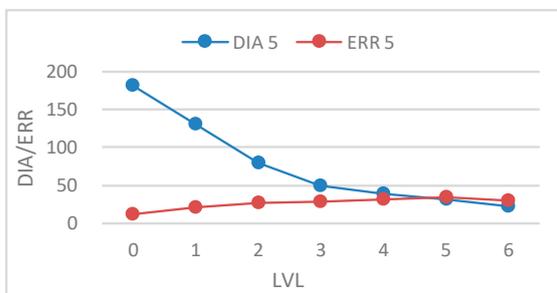
(b)



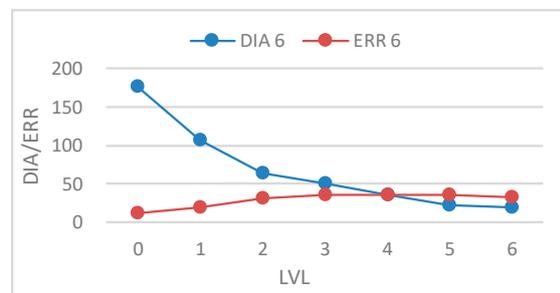
(c)



(d)

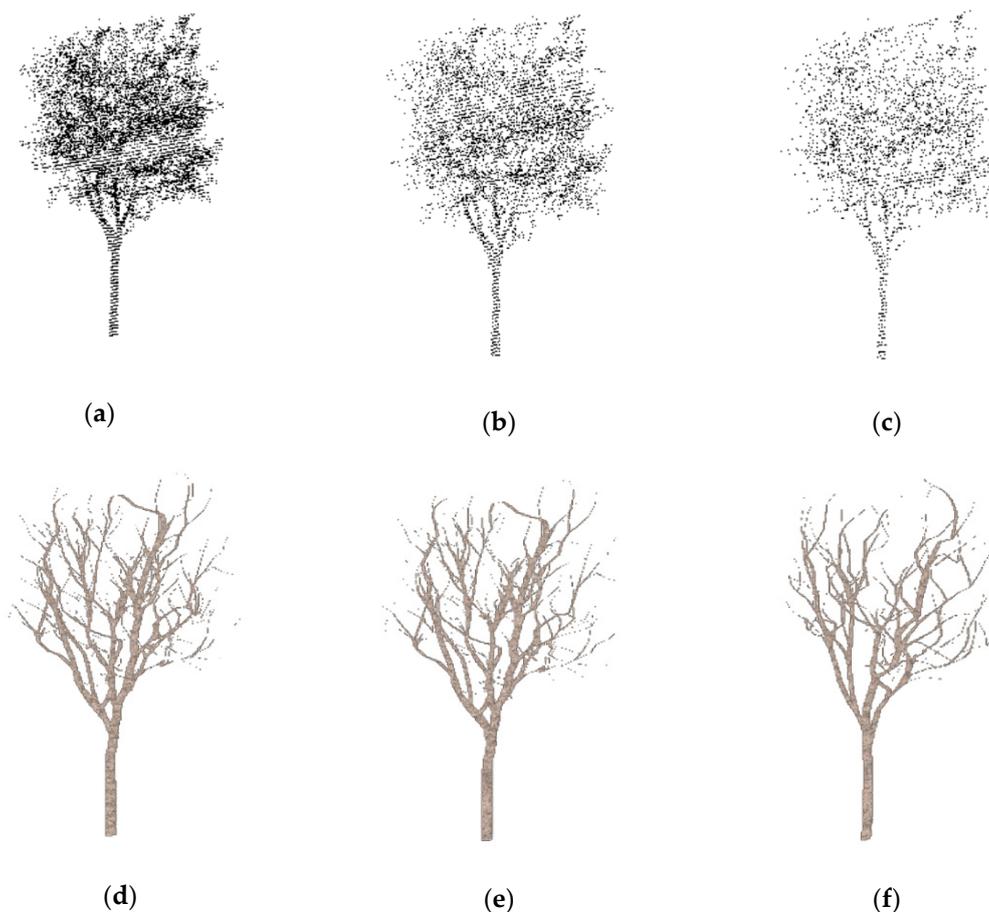


(e)



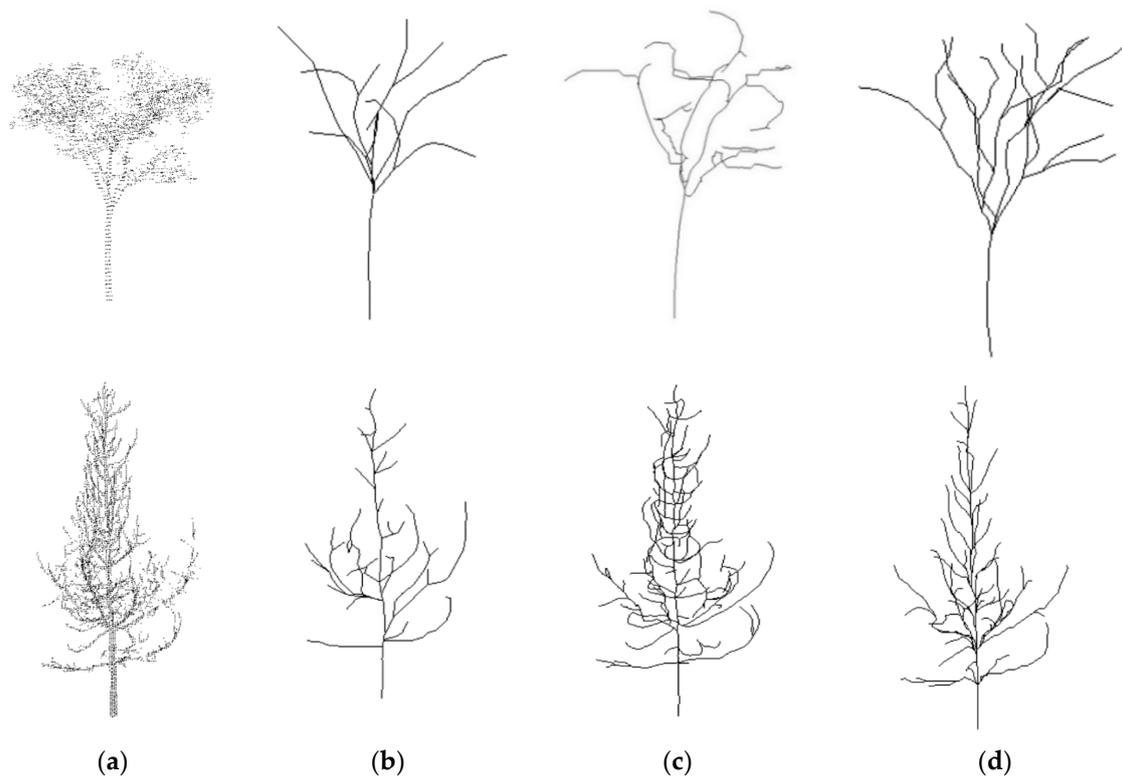
(f)

**Figure 14.** Tree branch diameter and modeling error (in mm) with respect to the level of details for six trees ((a–f) for tree 1–6 respectively).



**Figure 15.** Comparison of modeling results with respect to different point densities: (a) original tree point clouds; (b) tree points thinned by 50%; (c) tree points thinned by 75%; (d) model from the original tree point clouds; (e) model from the tree points thinned by 50%; (f) model from the tree points thinned by 75%.

Finally, we compare our method with the ones in [15,22], respectively. The method in [15] aims to use structural knowledge of trees to model them. This method uses a single-source shortest path algorithm to extract the short path of each point to the root of the tree. Then the lengths of the shortest paths are quantized and the points are clustered into bins. Finally, the centroid of the points in each bin is calculated to generate the main skeleton of the tree. The method in [22] aims to automatically reconstruct the tree's geometric structure. It also intends to recover the tree from incomplete data. It uses a distance minimum spanning tree and a robust structure-aware global optimization method to extract tree skeletons from terrestrial laser scanning data. Figure 16 top row presents a tree point cloud and the extracted skeleton lines from the method in [15] (implemented by ourselves), [22] and this paper, respectively. It can be seen that the skeleton lines generated by these two methods in [15,22] are similar to ours in general. Compared with our method, the main skeleton generated by [15] is basically consistent with the shape of the tree, while missing some details because of the clustering process. Since it uses the Laplacian smoothing operator for branch processing, results in Figure 16c top from [22] show smaller or shorter branches that we do not have. Nevertheless, it does not completely follow the tree structure in straight branches, yielding unnaturally connected, sharp artifact branches. Our optimization method is able to adjust the skeleton line by the consistency of the connection direction between the parent node and the child node. It can be seen from Figure 16d that the shape of the tree is better retained by the proposed method.



**Figure 16.** Two trees for comparison of tree skeleton line extraction results: (a) Tree point clouds; (b) skeleton lines by the method [15]; (c) skeleton lines by the method [22]; (d) skeleton lines by our method.

Figure 16 bottom shows the skeleton line extraction result of another tree. It can be seen from the figure that skeleton lines obtained by the three methods are very similar. Although the method in [15] can generate a clear and smooth main skeleton, compared with the other two methods, the main skeleton obtained by this method misses more details. The skeleton lines obtained by the method in [22] are generally smoother than the results of our method. However, both methods can extend skeleton lines to the end of tree branches. In comparison, the length of skeleton lines obtained by our method is a little constricted. The main reason is that our method regards the endpoint of the branch as a leaf node, which is not included in the skeleton. In general, the skeleton lines obtained by our method is able to present more details and better retain the tree structure.

## 5. Conclusions

The emergence of laser scanning technology provides a novel and effective means for reconstructing tree models with realistic structures. In this paper, we presented a hierarchical modeling method for street trees that can classify tree skeletons into different levels using a branch ranking method. This hierarchical structure of street trees can then be modeled by flat cones based on tree skeleton lines. Because the tree skeleton lines are extracted by the Dijkstra algorithm and optimized by a branch ranking method, the resultant street tree models appear natural and realistic. Multi-street trees were used to evaluate the effectiveness of the proposed method. Test results show that the hierarchical strategy can reconstruct tree models at a high level of detail directly from tree point clouds. Errors in our tree models were shown to be larger for a higher level of the tree branches, which means that the accuracy of the tree models was related to the degree of detail provided in the tree point clouds. The maximum error of all tree models is less than the average point spacing, approximately 0.61 times the average distance of the point cloud. Compared with two recently reported methods, our method can better retain the characteristics of the input point clouds. For our method to reconstruct

tree branches in different levels successfully, they must be sufficiently sampled in the point clouds. Furthermore, because the proposed method focuses on reconstructing tree branches, significant returns from tree leaves and fruits in the point clouds will be regarded as noise. As such, these returns need to be filtered out before applying our method to assure reliable modeling performance. Future research will focus on how to reconstruct tree models from sparse or incomplete point clouds and to model fine details such as leaves.

**Author Contributions:** J.X. was primarily responsible for conceiving the method and writing the source code and the manuscript; J.S. provided critical technical discussions and contributed to the manuscript writing; and G.W. performed the experiments and participated in the analysis of the results. The work was in part carried out when J.X. was visiting Purdue University. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research was funded by the National Natural Science Foundation of China (41671450 and 41801394), the National Key Research and Development Program (2018YFD1100405), and the China Scholarships Council (201806275006).

**Acknowledgments:** Special thanks go to anonymous reviewers who provided constructive comments that substantially improved the quality of the paper. Tree skeletons extracted by the method of [22] in this paper are implemented and provided by Dong Chen at the College of Civil Engineering, Nanjing Forestry University, to whom we are very grateful.

**Conflicts of Interest:** The authors declare that they have no conflict of interest.

## References

1. Lindenmayer, A. Mathematical models for cellular interaction in development parts I and II. *J. Theor. Biol.* **1968**, *18*, 280–315. [[CrossRef](#)]
2. Prusinkiewicz, P.; Lindenmayer, A.; Hanan, J. Development models of herbaceous plants for computer imagery purposes. *ACM SIGGRAPH Comput. Graph.* **1988**, *22*, 141–150. [[CrossRef](#)]
3. Wither, J.; Boudon, F.; Cani, M.-P.; Godin, C. Structure from silhouettes: A new paradigm for fast sketch-based design of trees. *Comput. Graph. Forum* **2009**, *28*, 541–550. [[CrossRef](#)]
4. Makoto, O.; Shigeru, O.; Takeo, I. Interactive design of botanical trees using freehand sketches and example-based editing. *Comput. Graphics Forum* **2005**, *24*, 487–496.
5. Teng, C.-H.; Chen, Y.-S.; Hsu, W.-H. Constructing a 3D trunk model from two images. *Graph. Model* **2007**, *69*, 33–56. [[CrossRef](#)]
6. Quan, L.; Tan, P.; Zeng, G.; Yuan, L.; Wang, J.; Kang, S.B. Image-based plant modeling. *ACM Trans. Graph.* **2006**, *25*, 599–604. [[CrossRef](#)]
7. Zhu, C.; Zhang, X.; Hu, B.-G.; Jaeger, M. Reconstruction of tree crown shape from scanned data. In Proceedings of the Technologies for E-Learning and Digital Entertainment, Third International Conference, Nanjing, China, 25–27 June 2008; Volume 5093, pp. 745–756.
8. Rutzinger, M.; Pratihast, A.K.; Elberink, S.O.; Vosselman, G. Tree modelling from mobile laser scanning data-sets. *Photogramm. Rec.* **2011**, *26*, 361–372. [[CrossRef](#)]
9. Bucksch, A.; Lindenbergh, R.C. CAMPINO—A skeletonization method for point cloud processing. *ISPRS J. Photogramm. Remote. Sens.* **2008**, *63*, 115–127. [[CrossRef](#)]
10. Bucksch, A.; Lindenbergh, R.; Menenti, M. SkelTre: Robust skeleton extraction from imperfect point clouds. *Vis. Comput.* **2010**, *26*, 13–20. [[CrossRef](#)]
11. Su, Z.; Zhao, Y.; Zhao, C.; Guo, X.; Li, Z. Skeleton extraction for tree models. *Math. Comput. Model.* **2011**, *54*, 1115–1120. [[CrossRef](#)]
12. Zhang, D.; Yun, T.; Xue, L.; Luo, Y. Reconstruction algorithm with complex topology of tree branches. *J. Nanjing Norm. Univ. (Nat. Sci. Ed.)* **2015**, *38*, 128–136. (In Chinese)
13. Cao, J.; Tagliasacchi, A.; Olson, M.; Zhang, H.; Su, Z. Point Cloud Skeletons via Laplacian Based Contraction. In Proceedings of the IEEE International Conference on Shape Modeling and Applications, Aix-en-Provence, France, 21–23 June 2010; pp. 187–197. [[CrossRef](#)]
14. Verroust, A.; Lazarus, F. Extracting skeletal curves from 3D scattered data. *Vis. Comput.* **2000**, *16*, 15–25. [[CrossRef](#)]
15. Xu, H.; Gossett, N.; Chen, B. Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Trans. Graph.* **2007**, *26*, 19. [[CrossRef](#)]

16. Huang, H.; Tang, L.; Chen, C. A 3D individual tree modeling technique based on terrestrial LiDAR point cloud data. In Proceedings of the 2015 2nd IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM), Fuzhou, China, 8–10 July 2015; pp. 152–156.
17. Borchert, R.; Slade, N.A. Bifurcation Ratios and the Adaptive Geometry of Trees. *Int. J. Plant Sci.* **1981**, *142*, 394–401. [[CrossRef](#)]
18. Yan, D.-M.; Wintz, J.; Mourrain, B.; Wang, W.; Boudon, F.; Godin, C. Efficient and robust reconstruction of botanical branching structure from laser scanned points. In Proceedings of the 2009 11th IEEE International Conference on Computer-Aided Design and Computer Graphics (CAD/CG 09), Huangshan, China, 19–21 August 2009; pp. 572–575.
19. Livny, Y.; Yan, F.L.; Olson, M.; Chen, B.Q.; Zhang, H.; El-Sana, J. Automatic reconstruction of tree skeleton from point clouds. *ACM Trans. Graph.* **2010**, *29*, 1–8. [[CrossRef](#)]
20. Balsa-Barreiro, J.; Fritsch, D. Generation of visually aesthetic and detailed 3D models of historical cities by using laser scanning and digital photogrammetry. *Digit. Appl. Archaeol. Cult. Herit.* **2018**, *8*, 57–64. [[CrossRef](#)]
21. Chen, L.C.; Teo, T.; Shao, Y.; Lai, Y. Fusion of LIDAR data and optical imagery for building modeling. *Int. Arch. Photogramm. Remote Sens.* **2004**, *35*, 732–737.
22. Wang, Z.; Zhang, L.; Fang, T.; Mathiopoulos, P.T.; Qu, H.; Chen, N.; Wang, Y. A structure-aware global optimization method for reconstructing 3-d tree models from terrestrial laser scanning data. *IEEE Trans. Geosci. Remote Sens.* **2014**, *52*, 5653–5669. [[CrossRef](#)]
23. Wang, Z.; Zhang, L.; Fang, T.; Tong, X.; Mathiopoulos, P.T.; Zhang, L.; Mei, J. A local structure and direction-aware optimization approach for three-dimensional tree modeling. *IEEE Trans. Geosci. Remote Sens.* **2016**, *54*, 4749–4757. [[CrossRef](#)]
24. Mei, J.; Zhang, L.; Wu, S.; Wang, Z.; Zhang, L. 3D tree modeling from incomplete point clouds via optimization and L1-MST. *Int. J. Geogr. Inf. Sci.* **2016**, *31*, 1–23. [[CrossRef](#)]
25. Axelsson, P. DEM generation from laser scanner data using adaptive TIN models. *ISPRS J. Photogramm. Remote Sens.* **2000**, *33*, 110–117.
26. Chen, M.; Wan, Y.; Wang, M.; Xu, J. Automatic Stem Detection in Terrestrial Laser Scanning Data with Distance-Adaptive Search Radius. *IEEE Trans. Geosci. Remote Sens.* **2018**, *56*, 2968–2979. [[CrossRef](#)]
27. Zhong, R.; Wei, J.; Su, W.; Chen, Y.F. A method for extracting trees from vehicle-borne laser scanning data. *Math. Comput. Model.* **2013**, *58*, 733–742. [[CrossRef](#)]
28. Kaasalainen, S.; Kukko, A.; Lindroos, T.; Ahokas, E.; Litkey, P.; Kaartinen, H.; Hyypä, J. Brightness Measurements and Calibration with Airborne and Terrestrial Laser Scanners. *IEEE Trans. Geosci. Remote Sens.* **2008**, *46*, 528–534. [[CrossRef](#)]
29. Kaasalainen, S.; Jaakkola, A.; Kaasalainen, M.; Krooks, A.; Kukko, A. Analysis of Incidence Angle and Distance Effects on Terrestrial Laser Scanner Intensity: Search for Correction Methods. *Remote. Sens.* **2011**, *3*, 2207–2221. [[CrossRef](#)]
30. West, G.B.; Brown, J.H.; Enquist, B.J. A general model for the structure and allometry of plant vascular systems. *Nature* **1999**, *400*, 664–667. [[CrossRef](#)]
31. Murray, C.D. A relationship between circumference and weight in trees and its bearing on branching angles. *J. Gen. Physiol.* **1927**, *10*, 725–729. [[CrossRef](#)]
32. Yang, M.; Wan, Y.; Liu, X.; Xu, J.; Wei, Z.; Chen, M.; Sheng, P. Laser data based automatic recognition and maintenance of road markings from MLS system. *Opt. Laser Technol.* **2018**, *107*, 192–203. [[CrossRef](#)]
33. Balsa-Barreiro, J.; Lerma, J.L. Empirical study of variation in lidar point density over different land covers. *Int. J. Remote. Sens.* **2014**, *35*, 3372–3383. [[CrossRef](#)]
34. Lari, Z.; Habib, A. New Approaches for Estimating the Local Point Density and its Impact on Lidar Data Segmentation. *Photogramm. Eng. Remote. Sens.* **2013**, *79*, 195–207. [[CrossRef](#)]
35. Guo, Q.; Li, W.; Yu, H.; Alvarez, O. Effects of Topographic Variability and Lidar Sampling Density on Several DEM Interpolation Methods. *Photogramm. Eng. Remote. Sens.* **2010**, *76*, 701–712. [[CrossRef](#)]

