Automation of Dormant Pruning in Specialty Crop Production: An Adaptive Framework for Automatic Reconstruction and Modeling of Apple Trees

Noha M. Elfiky, Shayan A. Akbar, Jianxin Sun, Johnny Park, Avinash Kak
Purdue University School of Electrical and Computer Engineering
Electrical Engineering Building, 475 Northwestern Ave, West Lafayette, IN 47907
nelfiky@purdue.edu, sakbar@purdue.edu,sun287@purdue.edu, jpark@purdue.edu, kak@purdue.edu

Abstract

Dormant pruning is one of the most costly and labor-intensive operations in specialty crop production. During winter, a large crew of trained seasonal workers has to carefully remove the branches from hundreds of trees using a set of pre-defined rules. The goal of automatic pruning is to reduce this dependence on a large workforce that is currently needed for the job. Automatically applying the pruning "rules" entails construction of $3D$ models of the trees in their dormant condition (that is, without foliage) and accurate estimation of the pruning points on the branches.

This paper investigates the use of Skeleton-based Geometric (SbG) features in a $3D$ reconstruction scheme. The results obtained demonstrate the effectiveness of the SbG features for automatic reconstruction using only two views — the front and the back. Our results show that our proposed scheme locates the pruning points on the tree branches with an accuracy of 96.0%. The algorithm that locates the pruning points is based on a new adaptive circle-based-layer-aware modeling scheme for the trunks and the primary branches "PBs" of the trees. Its three main steps are detection, segmentation, and modeling. Localization of the pruning points on the tree branches is a part of the modeling step. Both qualitative and quantitative evaluation are performed on a new challenging apple-trees dataset that is collected for the purpose of evaluating our approach.

1. Introduction

Dormant pruning, which involves cutting off certain primary branches (i.e., the branches which are connected directly with the trunk of a tree), is a critical component in the specialty crop production. Without pruning, fruit quality and the efficacy of pest and disease control decline precipitously, which ultimately results in significantly reduced orchard profitability. However, it is a very costly and labor-intensive practice. To mitigate the need for a large skilled seasonal workforce, there is currently much interest amongst specialty crop producers in mechanized and automatic pruning techniques.

The objective of automatic dormant pruning is to determine the locations of potential pruning points of the tree and measure relevant parameters of the PBs, such as the diameter, angle between the trunk and the PBs, etc. These parameters are used to decide whether each pruning candidate should be pruned indeed. Several factors, such as the complexity of the tree structures in modern high-density orchards, illumination variations, background clutter, and partial occlusions, complicate the task and make it a challenging computer vision problem. In this paper, we investigate to what extent the usage of the proposed reconstruction and modeling framework can alleviate some of these issues.

Most state-of-the-art work in automatic pruning relies on depth sensors or cameras [1, 16, 9, 10, 12, 5, 6] to capture the range images of a target tree from multiple views and reconstruct its $3D$ model. There exist two recent approaches to extract tree models from depth information. The first approach employs generative branching-structure models [12, 2, 7, 11], which provide hypotheses on the location of the branching points and extract key features from the measured data to confirm or reject these hypotheses. The problems with this approach arise when the tree structures are complex (i.e., a large number of hypotheses are required in order to maintain the same level of accuracy). The second approach, analyzes patterns of connecting points, finds either a minimum spanning tree [10] or a shortest path tree [17, 4], and applies cylindrical modeling on segmented point clouds of the trunk and PBs. However, incorrect segmentations are very difficult to avoid using these connection analysis methods and robustness is always an issue. In addition, it is hard to accurately model complex tree structures with sharp curvatures using those cylindrical models.

The work cited above for reconstructing a tree model from $2D$ and $3D$ sensory information does not lend itself well to the automatic calculation of pruning point. There are two reasons for this: the *first* is related to the lack of important detailed data [1, 16, 14, 15, 13], that is required for creating algorithmic implementations of the pruning heuristics used in manual pruning. The manual pruning heuristics are based on considerations such as how thick the branches are and/or what angle the PBs are making with respect to the trunk of the tree. The *second* issue is related to the fact that modern orchards tend to be dense, with very limited spacing between the trees. Therefore, it is not feasible to perform $3D$ reconstructions as in [1, 16] due to the high proximity of the trees.

The research reported in this paper is based on the sensory data collected with the KinectFusion (KF) software [8] that is provided with the Kinect 2 sensor in order to produce the $3D$ reconstruction of the scene/object in real-time. In our case, KF generates either the front or the back point cloud "PC" of the dormant apple tree, while simply moving the sensor around it. To obtain the final $3D$ reconstruction of the tree, we evaluate the use of geometric features in order to merge these front and back PCs *automatically*. Exploiting geometric attributes, which are robust to variations in illuminant, shadows, shading, and tree geometry, is a difficult challenge. Therefore, we investigate the use of geometric features for the automatic reconstruction task.

With regard to the specific contributions we report in this paper, we first extend the KF's reconstructions with geometric attributes that have yielded excellent $3D$ reconstruction results due to their discriminative power and robustness. The reconstruction scheme that comes with the KF sensor was found to be not optimal for high-density orchards. The KF reconstruction algorithm makes it impossible for the person who is capturing the images to go to the back side of the tree while capturing images on its front side, and vice versa. When sensory data is collected from two opposite viewpoints — as we do in our work — that invalidates the fundamental KF assumption, which assumes that the input images captured are extremely close to each other. To solve this problem, we record two separate point clouds, one for the front and the other for the back using the KF's reconstruction algorithm. Subsequently, we merge these PCs and find an initial registration based on the extracted geometric features. Finally, we refine the registration of the obtained PC using the Iterative Closest Point (ICP) algorithm in order to form the final $3D$ reconstruction of the tree. In addition, we propose an adaptive circle-based-layer-aware modeling technique, which accurately estimates the locations of potential pruning points as well as other relevant information of the tree model that is required for the automatic pruning task. We show that this technique allows for achieving efficient tree models on both synthetic and real data.

2. The Data Acquisition System

This research started out with performing a comparative evaluation of two different sensor types, *LIDAR* and Kinect 2, for constructing $3D$ point clouds of dormant trees. Although the *LIDAR* sensors are expensive and heavy, they typically produce high quality data. On the other hand, the *Kinect* 2 sensor is inexpensive, lightweight, and the quality of its data is comparable to that of the *LIDAR* as we will show in this section. Given these advantages of the Kinect 2 sensor, we eventually settled on using the Kinect 2 sensor along with its Kinect Fusion (KF) software.

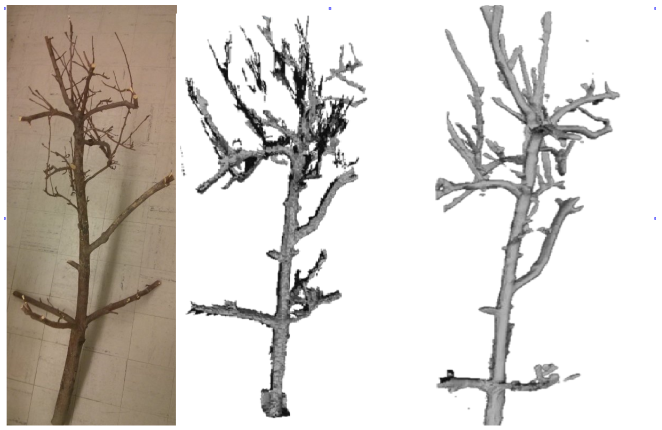


Figure 1. The ground-truth of tree $T1_{indoor}$ is shown on the left. The obtained reconstruction using LIDAR is shown on the middle. The Kinect 2 reconstruction is shown on the right.

To create a point cloud for an extended $3D$ object with the Kinect 2 sensor, holding the sensor by hand, one typically moves it around the object through overlapping sweeps. The KF software is smart enough to fuse together the points that are the same in the data collected from different but adjoining views. Through this process, the KF software can produce a very realistic looking $3D$ point cloud for a scene. This process works if there is sufficient overlap between successive data scans recorded by the sensor. Our main challenge with this approach to constructing a $3D$ point cloud for a tree is that it is NOT possible to walk around a tree since the trees in modern orchards are planted too close together.

In Figure 1, we show the $3D$ point cloud reconstructions obtained using the two aforementioned sensors. It is clearly demonstrated that the best results are obtained using the Kinect 2 (i.e., as shown in the upper part of the tree). In summary, inexpensive, lightweight Kinect 2 sensors provide accurate, high-resolution data — in contrast to using expensive, heavy-weighted LIDAR sensors. As a result, we base our approach on the Kinect 2 sensor with the KF software.

3. Reconstructing 3D Point Clouds using Skeleton-based Geometric Attributes

The goal here is to merge the data collected from the front and back of a tree into a single $3D$ point cloud. Note that the trees in modern orchards are usually planted much too close together and do not allow for collecting $3D$ data

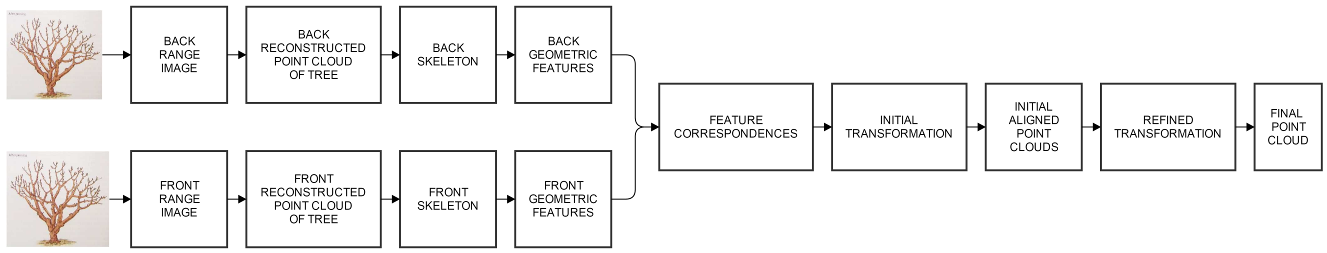


Figure 2. We show the block diagram of the proposed approach for constructing a $3D$ point cloud of a tree. The first and second rows show the processing of the back (B) and front (F) sides of the target tree T, respectively. The main steps involved are: (i) obtain the reconstruction of the F and B sides, separately, (ii) extract geometric features from the two views, (iii) perform a feature matching step to obtain the initial registration of T. Finally, we perform a final refinement step using ICP to get the desired $3D$ reconstrucion.

Algorithm 1 Geometric Feature Extraction for $3D$ reconstruction

Step A: Trunk Feature Extraction

1) Start from the root and for each candidate node, examine the no. of its connections in the connectivity matrix "C".

2) Check if $\sum_{j=1}^{M} C_{ij} > 2$, where M is the total number of skeleton nodes. Then, add the current node "i" to the trunk feature set TR$_{fp}$, and move towards the next candidate node that forms the least angle with respect to the z-axis. Otherwise, move to the upward node.

3) Repeat step 2 until no more nodes are found on the trunk. The output is the set of all nodes TR$_{fp}$ on the trunk.

Step B: Primary Branch (PB) Feature Extraction

1) Start from a node in TR$_{fp}$ and move inside the PB.

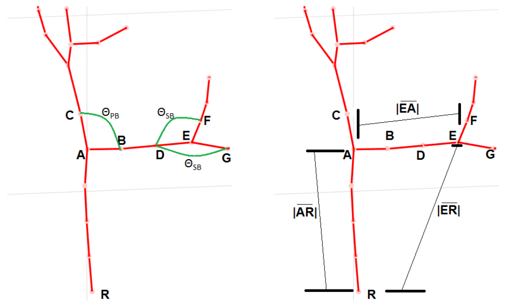
2) Check if $\{(\sum_{j=1}^{M} C_{ij} > 2) || (\sum_{j=1}^{M} C_{ij} = 1)\}$, then add node i to the PB feature set PB$_{fp}$, and go to step 3. Otherwise, move towards the next node along the PB. Note that, in case a PB does not have any secondary branch connections, we use its terminating point as a feature point.

3) Repeat steps 1 & 2 for all nodes in TR$_{fp}$. The output is the set of all nodes PB$_{fp}$ on the PB.

for a tree through a single sweep around the tree.

We scan and reconstruct each side of a tree (T) and then, using the algorithm presented in this section, we merge the two views into a single 3D point cloud.

For this purpose, we gather the depth images from the front (F) and back (B) sides of T, separately, and feed the data to the KF software to reconstruct the models separately for the F and B views. To merge F and B views and to build the final model of T, we extract geometric features from the skeleton of T, and then use these features to obtain the initial transformation (i.e., the rotation & translation matrices) between the F and the B views. Subsequently, trough the application of this transformation to the point cloud in one of the views, we merge it with the point cloud in the other view. Finally, we refine the overall 3D point cloud using the ICP algorithm. Figure 2 shows the block diagram of this scheme.



(a) (b)

Figure 3. We show an iIlustration of the feature points and feature descriptors used in our approach. In (a) we show the angle descriptors (θ_{PB} and θ_{SB}), while in (b) we show the distance descriptors ($|\overline{AR}|$, $|\overline{ER}|$, and $|\overline{EA}|$). θ_{PB} is the angle between a trunk feature point "TR$_{fp}$" and PB originating from it. θ_{SB} is the angle between a PB feature point "PB$_{fp}$" and the secondary branch "SB" originating from it. $|\overline{AR}|$, $|\overline{ER}|$, and $|\overline{EA}|$ represent the distances between (i) the TR$_{fp}$ and the root "R", (ii) the PB$_{fp}$ and R, and (iii) the PB$_{fp}$ and the TR$_{fp}$, respectively.

In order to obtain the geometric features, we first perform a Laplacian Smoothing function "f" on the entire PC as in [3], to extract the skeleton "S" of T. Then, we divide S into nodes connected with edges (i.e., denoted as "the connectivity matrix C " in Eq. 1). Next, we extract two main feature point nodes "fp" on S, namely, TR$_{fp}$ and PB$_{fp}$. TR$_{fp}$ are all the nodes on the trunk "TR" with PBs connected to them. On the other hand, PB$_{fp}$ are all the nodes on the PBs having secondary branches "SBs" connected to them. Algorithm 1 describes the steps for extracting TR$_{fp}$ and PB$_{fp}$ respectively.

$$[S, C] = f(pc), \tag{1}$$

where, f is the skeleton-extraction function and pc rep-

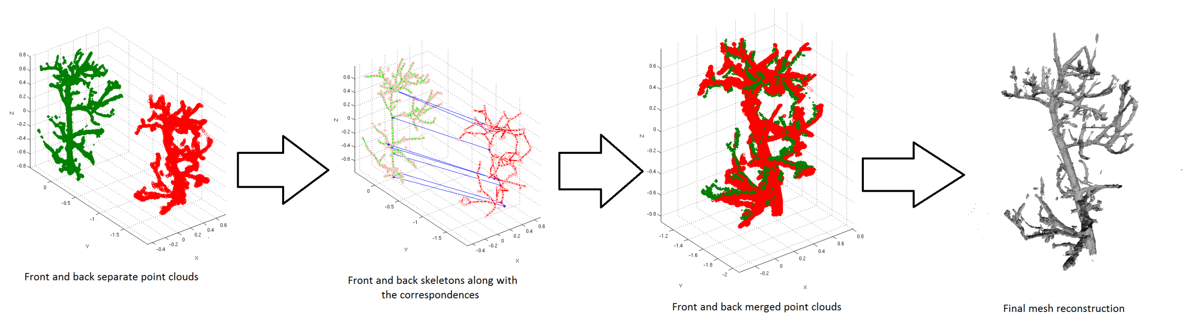

Front and back separate point clouds
Front and back skeletons along with the correspondences
Front and back merged point clouds
Final mesh reconstruction

Figure 4. We show the $3D$ reconstruction process. (1) We obtain the PCs of F and B. (2) We extract geometric features from the skeleton "S" of the tree and perform feature correspondences. (3) This results in the inital alignment of F and B, and (4) we apply ICP on the entire PC to form the final $3D$ reconstruction.

resents the input point cloud and is an $N \times 3$ matrix, where N is the total number of points in pc. S contains the coordinates of the skeleton nodes and is an $M \times 3$ matrix, where M is the total number of nodes in the skeleton S. C represents the connectivity matrix between the nodes in S and is an M×M matrix.

After obtaining the TR$_{fp}$ and PB$_{fp}$ feature points, the next step is to describe them as demonstrated in Figure 3. In particular, for each TR$_{fp}$ we assign the following descriptors, namely, θ_{PB} and $|\overline{AR}|$. θ_{PB} is the angle that the PB originating from the TR$_{fp}$ is making with TR and $|\overline{AR}|$ is the distance between TR$_{fp}$ and the root "R" of the tree. On the other hand, we describe each PB$_{fp}$ using θ_{SB}, $|\overline{ER}|$, and $|\overline{EA}|$. θ_{SB} is the angle that the SBs originating from the PB$_{fp}$ is making with the PB$_{fp}$, $|\overline{ER}|$ is the distance between the PB$_{fp}$ and R, and $|\overline{EA}|$ is the distance between PB$_{fp}$ and TR$_{fp}$. We use these descriptors to find the correspondences and the initial alignment between the F and B PCs. Lastly, we merge these initially aligned PCs using ICP to obtain the entire $3D$ reconstruction of T, as illustrated in Figure4.

3.1. A Stacked-Circle Algorithm for Fitting a 3D Model to a 3D Point Cloud

Our algorithm is based on fitting circles recursively to the point cloud for a tree. Fitting circles requires estimating the best locations for the centers of the circles and their radii. The algorithm starts by tracking the skeleton of a tree from base up and, at each point (i.e., layer) along the skeleton, fitting a circle to the points in the point cloud that are in a plane parallel to the ground plane. The algorithm is adaptive in the sense that that the candidate parameters chosen for the circle (the coordinates of the center and the radius) at each height above the ground plane depend on the parameters used for two adjoining circles. This lends smoothness to the $3D$ tree model that is fit to the data and, at the same time, allows the circles to change with the changing shape of the tree.

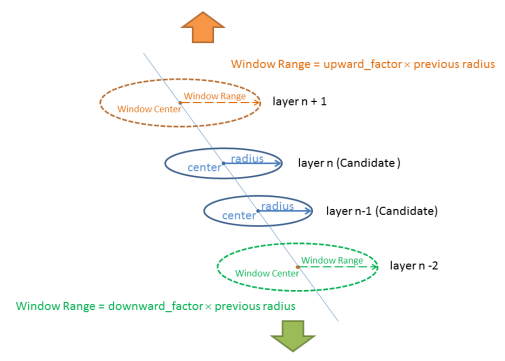
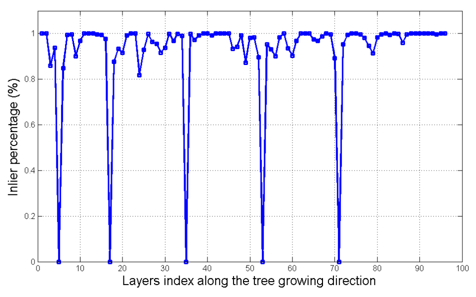


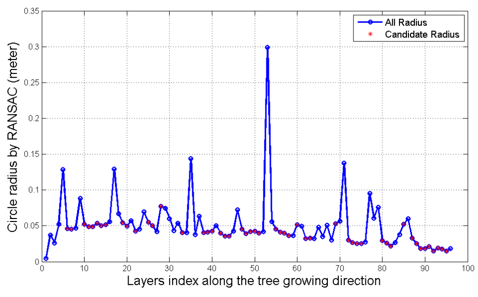
Figure 6. The construction of the bounding window "$bwin$". We use a circular-shaped bounding window (where its parameters "the center and radius" are calculated dynamically from previous layers that have been modeled) to limit the set of points within each layer of T to be used for modeling. The center$_{bwin}$ of a current layer is the linear interpolation of the centers of the previous two layers, while the radius$_{bwin}$ is adjusted based on the radius of the previous layer.

To present the details of the algorithm, it first divides the point cloud associated with the trunk into horizontal layers along the trunk's growing direction (the z direction). Subsequently, it projects the points in each layer on a plane parallel to the ground. The algorithm then fits a circle to the projected points using the Random Sample Consensus (RANSAC) algorithm. After such circles are estimated along the entire height of a trunk, their parameters (i.e., the coordinates of the center and the radii) are smoothed using a sequential algorithm that constructs a best estimate for the parameters at each height given the parameters in two adjoining layers along the height.

The algorithm is started by first locating the point on the trunk where the estimated parameters (the center coordinates and the radius) can be considered to be the most reliable according to following three criteria: (1) have high inlier percentages, (2) reside in the longest continu-



(a)



(b)

Figure 5. We initialze our CLAM approach with layers that have high inlier percentages, reside in the longest continuous sequence-of-layers with the minimum change in their radius, and are located at the lower region of the sequence. We show in the y-axis of (a) and (b) the inlier percentages and the radius within each layer (x-axis) on tree T1_{indoor}. The red markers indicate that the change within the radius in those layers is smaller than a certain threshold. We use layers 12 and 13 to initialize our CLAM, as they satisfy all the aforementioned conditions.

ous sequence-of-layers with the minimum change in their radius, and (3) are located at the lower region of the sequence. An illustrative example for tree T1_{indoor} is shown Figure 5, which demonstrates that layers 12 and 13 satisfy all the aforementioned conditions. Therefore, we use these two layers to initialize our CLAM approach for tree T$_{indoor}$.

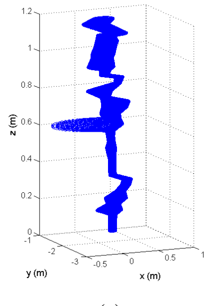
Subsequently, as described above, starting with the most-reliable layers, the algorithm marches in both directions and upgrades the estimated parameters at each layer along the trunk. This idea is best illustrated with the depiction in Figure 6. The two blue ovals in the figure are the locations of the points along the trunk that yielded the more reliable parameters. As the algorithm marches up the tree trunk, the red oval has its parameters updated using the parameter values in the two blue ovals. By the same token, as the algorithm marches down the tree trunk, the green oval updates the parameter values using the estimates in the blue ovals. In Figure 7, we show example results obtained for modeling the trunk of tree, with and without applying the proposed Circle-based-Layer-Aware Modeling (CLAM) approach. As shown, using CLAM results in a significant boost in the model accuracy.

3.2. Primary Branch Modeling

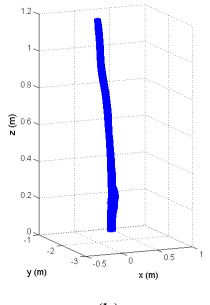
Modeling the PBs, where the main objective is to estimate the required pruning parameters, involves the following three steps. First, we segment the PC of the PBs (i.e., denoted as "PC$_{pbs}$") from the tree's PC. For this, we use the trunk model (i.e., introduced in Section 3.1) to extract the PC$_{pbs}$. In particular, we depend upon two trunk models, namely, "PC$_{tr}$" (i.e., the blue circle in Figure 8) and "PC$_{en}$" (i.e., the red circle in Figure 8), in order to segment the PC$_{pbs}$ — as it lies between PC$_{tr}$ and PC$_{en}$, then we remove the PC$_{tr}$ from PC$_{en}$. The second step is to cluster PC$_{pbs}$ into individual PBs, as illustrated in Figure 9. Lastly, we apply CLAM to model each individual PC$_{pb}$.

Algorithm 2 Modeling the Trunk and Primary Branches (PBs)

```
Step 1: Trunk Modeling
i) Divide the trunk PC to layers across the $z$ direction
ii) Find the "good initialization" layers using RANSAC
iii) Refine the point set of each layer using layer-awareness
iv)Apply circle-based modeling "CLAM" on the refined set
Step 2: Primary Branch (PB) Modeling
1) Segment PC of the PBs "PC$_{pbs}$" from the tree's PC
2) Cluster PC$_{pbs}$ into individual PBs
3) Filter out small candidate PB clusters (# points$<$ 400)
4) Calculate the growing direction of each PB ("PB$_{Vg}$")
5) Rotate PB$_{Vg}$ towards the $z$-axis and project PC$_{pb}$
6) Model PC$_{pb}$ using CLAM, as described in Step A
```



(a)



(b)

Figure 7. We show the trunk model of tree *T1$_{indoor}$* in (a) without Layer-aware Refinement. While, in (b) we show the outcome after applying the Layer-aware Refinement.

To model an individual PB (i.e., denoted as PB$_i$) using CLAM, we need to calculate its growing direction (i.e., denoted as PB$_{Vg(i)}$). We use RANSAC to fit a cylinder model on PC$_{pb(i)}$ and acquire its direction vector (i.e, denoted as

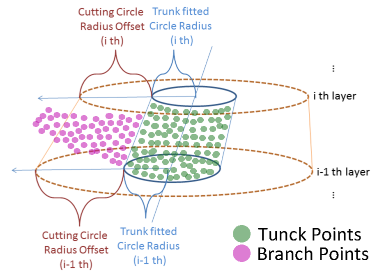

Cutting Circle Radius Offset (i th)
Trunk fitted Circle Radius (i th)
i th layer
i-1 th layer
Cutting Circle Radius Offset (i-1 th)
Trunk fitted Circle Radius (i-1 th)
Tunck Points
Branch Points

Figure 8. The illustration of the segmentation process to extract the point clouds of the trunk (shown in green dots) and the primary branches (shown in pink dots) from the original PC of the target tree.

PB$_{Vcyl(i)}$). We verify the correctness of PB$_{Vcyl(i)}$ (i.e., whether it is headed towards the growing direction of PB$_i$ or in the opposite direction) based on the direction of a reference vector "PB$_{Vref(i)}$". To calculate PB$_{Vref(i)}$, we first construct a new coordinate system (where its first and the second coordinates are the point's z value and the distance between the point and the trunk center, respectively). We compute the Euclidean Distances (ED) between each point in PC$_{pb(i)}$ and the world frame (0,0). We use the two points with the shortest and longest EDs to calculate PB$_{Vref(i)}$ based on the differences between them. We obtain PB$_{Vg(i)}$ by computing the dot product of PB$_{Vref(i)}$ and PB$_{Vcyl(i)}$ using Eq. 2.

$$PB_{Vg(i)} = \begin{cases} PB_{Vcyl(i)} & \text{if } (PB_{Vcyl(i)} \cdot PB_{ref(i)}) \geq 0, \\ -PB_{Vcyl(i)} & \text{if } (PB_{Vcyl(i)} \cdot PB_{ref(i)}) < 0, \end{cases} \quad (2)$$

Once the $PB_{Vg(i)}$ direction is calculated , we rotate it towards the z-axis of the world frame, project the points of PC$_{pb(i)}$ on it, and apply CLAM, as discussed in Sec. 3.1. Algorithm 2 summarizes the main steps for modeling the trunk and PBs based on the proposed CLAM approach.

4. Evaluation of the Framework with Experiments

Here we present the experimental setup (see Section 4.1) and the results of our experiments. Firstly, we evaluate the proposed $3D$ reconstruction scheme using the SbG features in Section 4.2. Secondly, we evaluate the performance of our adaptive CLAM approach for constructing a $3D$ model of a tree in Section 4.3. Thirdly, we perform an evaluation of the accuracy of the pruning parameters estimated from a $3D$ model of a tree in Section 4.4.

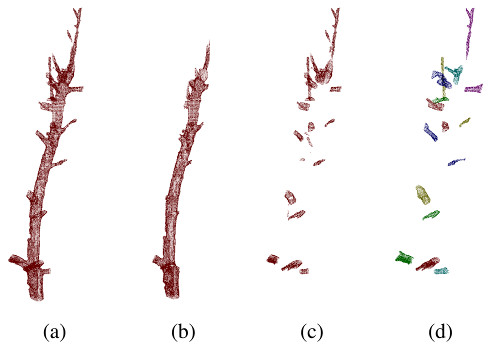


Figure 9. We show in (a) the original point cloud of tree *T1*$_{indoor}$, in (b) the segmented point cloud of the trunk, in (c) the segmented point cloud of the PB, and in (d) the individual clustered point clouds for each PB labeled in different colors.

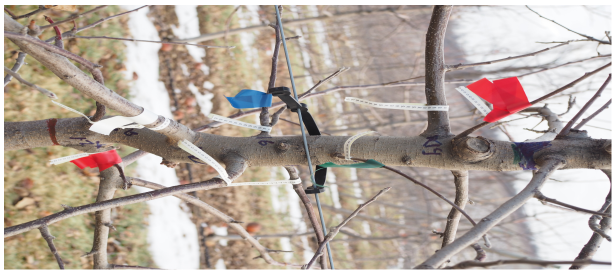


Figure 11. Each Tree is divided into sections starting from the tree-root. Each section is annotated based on the following color-coded pattern (i.e., red, blue, yellow, white, and green).

4.1. The Data, the Ground Truth, and the Evaluatoin Metrics

Dataset: Our evaluation dataset consists of six trees, with two constructed synthetically, two as stand-alone trees in our laboratory, and two consisting of trees in an apple orchard. These are shown in Figure 10. The trees *T1*$_{indoor}$ and *T2*$_{indoor}$ in the figure are stand-alone trees in our laboratory. The trees *T3*$_{outdoor}$ and *T4*$_{outdoor}$ are real live trees in an actual apple orchard. Finally, the trees *T5*$_{syn}$ and *T6*$_{syn}$ are generated synthetically by using Xfrog software. As the reader would expect, the data collection for the trees *T1*$_{indoor}$ and *T2*$_{indoor}$ is in an indoor environment. On the other hand, the data for trees *T4*$_{outdoor}$ and *T5*$_{outdoor}$ is collected in an outdoor environment. The data is collected using a Kinect 2 sensor. Each tree is scanned just front and back to yield two views. The depth data thus collected is then processed using our framework. Finally, the ground-truth is obtained by dividing each tree into several sections staring from the root. Subsequently, each primary branch is labeled with a color-coded pattern, as shown in Figure 11.
Evaluation Methodology: To validate the performance of our approach, we follow the protocol used in [9]. The results are presented using the Branch Identification Accuracy *(BIA)* metric, which is computed as the percentage of the

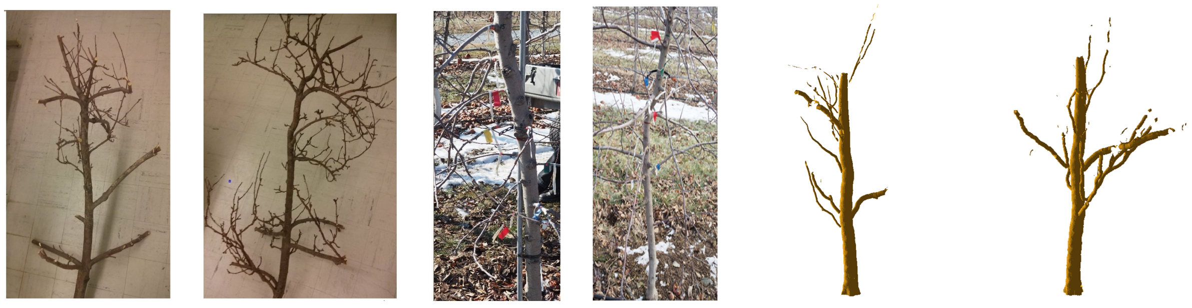

(a) $T1_{indoor}$ (b) $T2_{indoor}$ (c) $T3_{outdoor}$ (d) $T4_{outdoor}$ (e) $T5_{syn}$ (f) $T6_{syn}$

Figure 10. The real trees in the dataset are shown in (a), (b), (c), and (d). While, the synthetic trees are shown in (e) and (f).

Tree Number	Number of actual branches	Branch reconstruction		Branch Modeling		
		Reconstructed branches	BIA%	Modeled branches	BIA%	PPRA%
$T1_{indoor}$	16	15	93.75	15	93.75	73.00
$T2_{indoor}$	20	19	95.00	18	90.00	78.00
$T3_{outdoor}$	11	11	100.00	11	100.00	100.00
$T4_{outdoor}$	20	19	95.00	17	85.00	60.00
Mean (BIA%)			96.00		92.20	78.00

Table 1. The first column indicates the examined trees. We show in second column the number of ground-truthed primary branches in each tree. In the "Branch Reconstrucion" & "Branch Modeling" columns, we show the number of branches successfully reconstructed and modeled, respectively. Using the criterion whose values are displayed in the last column, we show the percentages of the detected primary branches with correct radius estimations over the total number of detected primary branches. We refer to this criterion as the *Primary Point Radius Accuracy (PPRA)*. The last row displays the mean values for the entries in the columns.

detected branches verified to be true over the actual number of ground-truth branches of the tree.

4.2. Experiment 1: Reconstruction of 3D Point Clouds using Skeleton-based Geometric Attributes

This experiment shows the efficacy of the proposed Skeleton-based Geometric (SbG) attributes for the reconstruction of a $3D$ point cloud from the front and the back scans of a tree. The top row of Figure 12 shows qualitatively the reconstructed point clouds. For a quantitative assessment of the quality of the reconstructed point clouds, the third column of Table 1 shows the value of the *BIA* metric defined previously in Section 4.1. On average, our reconstruction scheme allowed us to successfully detect 96.0% of the original tree branches. As stated previously, the ground truth for this accuracy measurement consists of the photographs of the trees after we have attached colored tags to the primary branches.

4.3. Experiment 2: Fitting 3D Tree Models to the Point Clouds

We now present validation experiments that evaluate the quality of the 3D tree models that are fit to the $3D$ point clouds using the approach introduced in Section 3.1. As stated earlier, the end-goal of this model fitting is the localization of the primary branches emanating from the trunk of the tree and estimation of the parameters of these branches in the vicinity of their junctions with the trunk. See the bottom row of Figure 12 for a qualitative assessment of the 3D models that are fit to the $3D$ point clouds for each tree. These results demonstrate that we get accurate and smooth $3D$ model on both synthetic and real data. For a more quantitative assessment, we go back to the values of the BIA metric in the sixth column of Table 1.

4.4. Experiment 3: Measuring the Accuracy of the Estimated Branch Radius

Since the radius of a primary branch is the main parameter used for locating a pruning point on a primary branch emanating from the trunk of a tree, it is important to evaluate the accuracy with which such radii are calculated by our framework. We measure these accuracies by comparing the estimated radii with their corresponding ground-truth values. The ground truth was obtained by asking orchard managers to locate the pruning points on the trees and by physically measuring branch diameters at those locations. These accuracy measurements are expressed through the parameter named Pruning Point Radius Accuracy (PPRA). It is the ratio of the total number of detected primary branches with

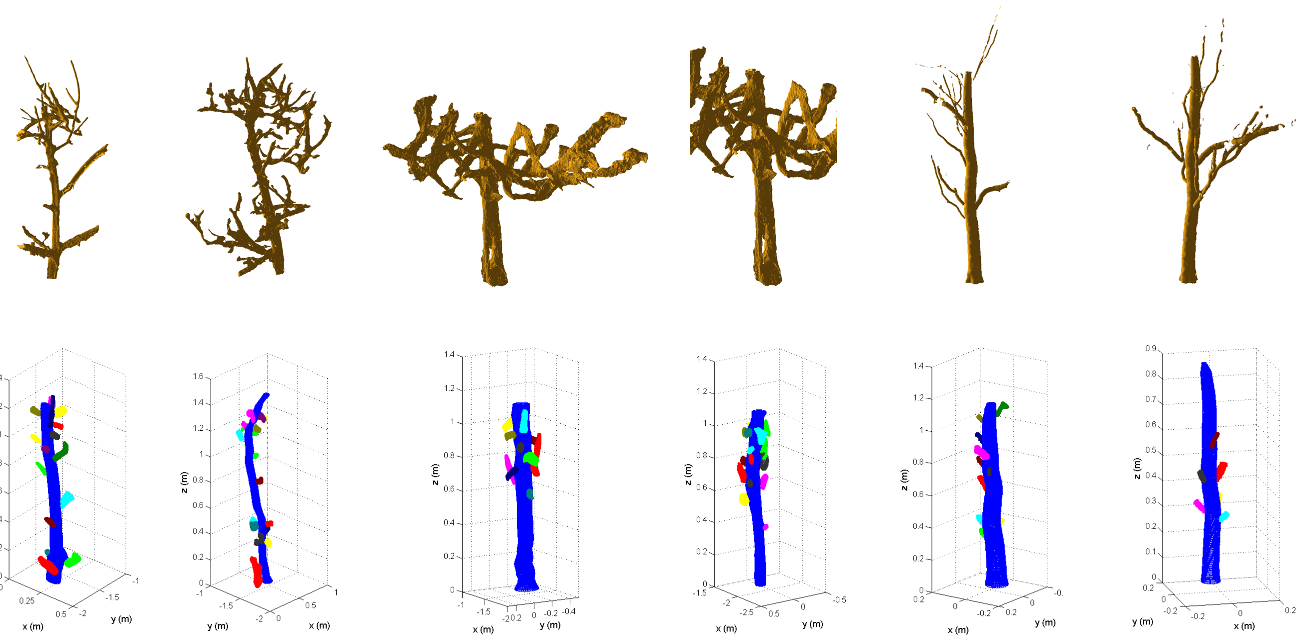


Figure 12. The first row shows the $3D$ point clouds reconstructed for each of the six trees of Figure 10. The second row shows the CLAM-based $3D$ tree model fit to the relevant portions of the point clouds. The colors show the locations of the different primary branches attached to the main trunk.

correctly estimated radii over the total number of successfully detected branches. By "correctly estimating" a primary branch radius, we mean estimating it within a tolerance of 2 millimeters. The last column of Table 1 shows the value of the PPRA parameter for the validation experiments. As shown, our approach achieves an average PPRA based accuracy of 78.0% on real trees for which the data was collected outdoors and on the trees for which the data was collected indoors, and, as the reader would expect, an accuracy of 100.0% on synthetic trees.

5. Conclusion and Future Work

In this work, we presented a complete system that first constructs a 3D model of an apple tree in the field, while the tree is in its dormant condition. We showed how such a model can be used to estimate important parameters that are relevant for pruning. The system uses Kinect 2 sensor for scanning the apple trees from just two viewpoints — front and back. Subsequently, the model of a tree is constructed using the SbG features presented in this paper. Our results demonstrate the effectiveness of the SbG feature based framework for model construction. Our results show that we can achieve an accuracy of 96.0% with regard to locating the pruning points on the limbs. The ground truth that was used for this accuracy measurement consisted of human tagged pruning points on a set of trees. This paper also presented an adaptive circle-based-layer-aware modeling "CLAM" scheme for the trunk and the primary branches of a tree. Our algorithms are able to detected correctly the primary branches with an accuracy of 92.2%.

Our plans for extending the research reported here are as follows: At the moment, when the adjacent trees are too close together, we must manually edit the scanned data for separating the tree of interest from its neighbors. We would like to automate this process in the future. We also wish to adaptively determine the best values to use for various thresholds used by our system. As a case in point, the parameter that is used to discard the branches that are too small (and, therefore, not deserving of further attention) is set manually for the data collected in the field for each tree.

Acknowlodegments

We would like to thank Professor Peter Hirst's group from the Department of Horticulture and Landscape Architecture at Purdue University and Professor James Schupp's group from the Department of Horticulture at Pennsylvania State University for helping us with the data collection process that is necessary for our expirements. In addition, we would also like to acknowlegde the USDA grant that supports the funding of the conducted research.

References

[1] B. Adhikari and M. Karkee. 3d reconstruction of apple trees for mechanical pruning. In *2011 ASABE International Meeting*, August 2011. 1

[2] J. Binney and G. Sukhatme. 3d tree reconstruction from laser range data. In *IEEE International Conference on Robotics and Automation*, pages 1321–1326, May 2009. 1

[3] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su. Point cloud skeletons via laplacian-based contraction. In *Proc. of IEEE Conf. on Shape Modeling and Applications*, pages 187–197, 2010. 3

[4] Z. Cheng, X. Zhang, and B. Chen. Simple reconstruction of tree branches from a single range image. *Journal of Computer Science and Technology*, 22(6), November 2007. 1

[5] M. Dassot, A. Colin, P. Santenoise, M. Fournier, and T. Constant. Terrestrial laser scanning for measuring the solid wood volume, including branches, of adult standing trees in the forest environment. *Computers and Electronics in Agriculture*, 89:86–93, November 2012. 1

[6] S. Delagrange and P. Rochon1. Reconstruction and analysis of a deciduous sapling using digital photographs or terrestrial-lidar technology. *Annals of botany*, 108(6):991–1000, April 2011. 1

[7] S. Friedman and I. Stamos. Automatic procedural modeling of tree structures in point clouds using wavelets. In *3D Vision - 3DV 2013, 2013 International Conference on*, pages 215–222, June 2013. 1

[8] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, and A. Fitzgibbon. Kinectfusion: Realtime 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 559–568, New York, NY, USA, 2011. ACM. 1

[9] M. Karkee, B. Adhikari, S. Amatya, and Q. Zhang. Identification of pruning branches in tall spindle apple trees for automated pruning. *Computers and Electronics in Agriculture*, 103(0):127 – 135, 2014. 1, 4.1

[10] Y. Livny, F. Yan, M. Olson, B. Chen, H. Zhang, and J. El-Sana. Automatic reconstruction of tree skeletal structures from point clouds. *ACM Trans. Graph.*, 29(6):151:1–151:8, December 2010. 1

[11] P. Prusinkiewicz and A. Runions. Computational models of plant development and form. In *New Phytologist*, pages 549–569, 2012. 1

[12] F. Schöler and V. Steinhage. Towards an automated 3d reconstruction of plant architecture. In *Applications of Graph Transformations with Industrial Relevance*, pages 51–64. Springer Berlin Heidelberg, 2012. 1

[13] R. Sun, J. Jia, H. Li, and M. Jaeger. Image-based lightweight tree modeling. In *Proceedings of the 8th International Conference on Virtual Reality Continuum and Its Applications in Industry*, VRCAI '09, pages 17–22, New York, NY, USA, 2009. ACM. 1

[14] P. Tan, G. Zeng, J. Wang, S. Kang, and L. Quan. Image-based tree modeling. *ACM Trans. Graph.*, 26(3), July 2007. 1

[15] C. Teng and Y. Chen. Image-based tree modeling from a few images with very narrow viewing range. *The Visual Computer*, 25, April 2009. 1

[16] Q. Wang and Q. Zhang. Three-dimensional reconstruction of a dormant tree using rgb-d cameras. In *2013 ASABE International Meeting*, July 2013. 1

[17] H. Xu, N. Gossett, and B. Chen. Knowledge and heuristic-based modeling of laser-scanned trees. *ACM Trans. Graph.*, 26(4), October 2007. 1