Measuring and Modeling Apple Trees using Time-of-Flight Data for Automation of Dormant Pruning Applications

Somrita Chattopadhyay[1] Shayan A. Akbar[1] Noha M. Elfiky[1] Henry Medeiros [2] Avinash Kak[1]
[1]School of Electrical and Computer Engineering, Purdue University
[2]Department of Electrical and Computer Engineering, Marquette University

Abstract

Dormant pruning is one of the most expensive, labor-intensive, but, unavoidable procedure in the field of horticulture to ensure quality crop production. During winter, skilled farmers remove certain branches that are connected directly with the trunk of a tree carefully using a set of predefined rules. In order to reduce this dependence on a large manpower, our goal is to automate this pruning process by building 3D models of dormant apple trees, which eventually would be fed to an intelligent robotic system.

*In this paper, we present a semicircle fitting based robust 3D reconstruction scheme for modeling the trunk and primary branches of apple trees. The method involves estimating the diameter-error, creating semicircle fit model of the tree from a single depth image, and reconstructing the final 3D model of the tree by aligning a sequence of depth images. Analysis of the qualitative as well as the quantitative evaluations of our algorithm on five different dormant apple trees from our dataset under various indoor and outdoor environments demonstrate the effectiveness of the proposed framework for automatic 3D reconstruction. The results show that on an average, the proposed schemes provide a performance of **89.4%** for correctly estimating the diameters of the primary branches with a tolerance of **5 mm** and **100%** for correctly identifying the branches.*

1. Introduction

In recent years, automation of dormant pruning, which involves cutting off certain primary branches (i.e., branches that are connected directly to the trunk) of a tree for improving the yield as well as the quality of crop production, has emerged as an interesting research area. The process of pruning a tree is not only expensive, time consuming, and labor-intensive; but also requires a thorough knowledge of the particular tree, and location of potential pruning points. One of the major steps in automating this pruning process is reconstructing 3D models of the apple trees with accurately measured branch and trunk diameters. This issue is particularly challenging due to several factors such as background clutter, variable illumination conditions, complex tree structures, partial occlusions, close proximity of the consecutively planted trees in the orchards, etc.

In this research work, we propose a novel method for 3D reconstruction of the trunk and primary branches which exploits the fact that any cross-section of a dormant tree can be reliably approximated by a semicircle due to inherent cylindrical structure of the trunk and branches. The method extracts 2D skeleton from the depth image of an apple tree, creates cross-section around each vertex of the 2D skeletal structure, thus, dividing the depth images into several cross-sections, and fits a semicircle on each cross-section. Our algorithm repeats this process for several depth images of the apple tree, taken from multiple viewpoints, aligns their corresponding 3D point clouds, and employs incremental as well as global approaches in order to accumulate all these data for the final robust reconstruction. We also incorporate an empirical diameter correction model to make our proposed approach even more robust and accurate. The model empirically adjusts the calculated diameter of a branch based on a study that concludes that the branches that are present at greater depths from the sensor would contain more error in their measurements.

The rest of the paper is organized as follows. In Section 2, we review the current-state-of-art. In Section 3, we describe our dataset and important parameters of our algorithm. The preprocessing stages of our algorithm are then discussed in Section 4. We explain the proposed framework in Section 5. In Section 6, we evaluate the performance of the algorithm on both indoor and outdoor trees, respectively. Finally, Section 7 concludes the paper, and highlights potential future works.

2. Related Works

Existing literature on geometric reconstruction and 3D scene modeling is vast. In this section, we provide a brief overview of the state-of-the-art of different 3D reconstruc-

tion methods of trees, specifically in the field of automatic pruning. We also focus on the current state-of-art of Kinect Fusion based approaches, which make effective use of depth maps and point clouds for 3D reconstruction and modeling.

The research presented in this paper is based on the data collected with a Kinect2 sensor using its KinectFusion software [16]. Kinect has been chosen for its easy portability and low cost; while providing comparable quality depth images (at a very high frame rate of 30 fps) as of other expensive LASER sensors currently available in the market. However, the raw depth images contain a lot of noise near and on the edge pixels of the tree leading to inaccurate diameter estimation of the branches. Thus in order to make our approach more robust than the other traditional approaches [22, 25, 20, 21], we have used an empirical error-correction model for accurate diameter measurements.

A key part in modeling tree structure is the segmentation of the captured data into branches. The segmentation gives the topological tree structure, and the resulting segments (branches) can be then geometrically reconstructed. Most of the existing approaches convert the raw depth data to their corresponding 3D point clouds [5, 6, 7] for extracting the tree skeletal structure. Skeletons can also be defined using a neighborhood graph and checking the connected components of the level sets of the graph [29, 9], and also by voxelization and mathematical morphology [12, 1, 18]. We have skeletonized the tree structure directly from the 2D depth images using a Voronoi diagram based technique [24], thus improving overall complexity of our algorithm.

Another integral part of our algorithm is the registration of 3D point clouds to obtain the final richer reconstruction. Literature has addressed different methods to find the full 6DOF scan alignment, such as joint optimization combining visual features and shape matching [15, 14], overlap heuristic approach [31], semidefinite programming [8], probabilistic approach [10] that takes into account the uncertainty associated with displacement of the sensor and the measurement noises, and many more [28, 4]. The most important class of alignment algorithms has been based on the Iterative Closest Point Algorithm (ICP) [2], a nonlinear optimization problem in which correspondences between images are approximated using the closest pairs of points found between scans at the previous iterations. Though computational complexity is still a concern, we have started our registration process with this reliable alignment algorithm, and have achieved pretty good results.

In the past few decades, 3D scene reconstruction has received immense attention from the researchers. The current state-of-the-art [32, 23] as well as the KinectFusion based approaches attempt to solve this problem without imposing any constraint on the object in the scene, thus resulting in inaccurate modeling for challenging real-world applications — such as 3D reconstruction of trees for dormant pruning. Many of these methods [1, 18, 30] employ generative branching rules, which may result in inaccurate reconstruction for complex tree structures in the absence of large number of hypotheses. These methods do not account for the fact that the trees are predominantly cylindrical in structure, and using this information may lead to simplification of the tree structure. The other classical approaches [19, 11] estimate the tree model without taking into consideration important information — such as erroneous diameter-measurements of the cross-sections.

In [19], Livny et al. have utilized graph-structure based geometric attributes of the tree skeletal structure for automatic reconstruction of non-dormant trees, which itself is a difficult problem to solve for complex tree structures. Moreover, it mainly aims at reconstructing the major branches of the trees, and then sample the leaves randomly. In [11], the focus is on merging the point clouds corresponding to the front and the back sides of the tree, based on geometric attributes. Although the work in [11] also proposes a circle fitting based reconstruction scheme, it does not take into consideration a number of distortions present in the data acquisition system. The novelty of the work presented in this paper lies in proposing a framework that aims at 3D reconstruction of apple trees by taking into account the erroneous diameter measurements along with utilizing the predominantly cylindrical structure of trees. This paper also focuses on resolving a number of issues that cause distortions in the estimated parameters of the fruit tree. More importantly, although in this method we also extract the skeleton of the tree, and model the trunk and branches using circles, we do so using more robust approaches. Instead of fitting circles to the trunk and branches after merging the point clouds corresponding to both sides as in [11], we fit semicircles using separate views, and correct their diameters based on an empirical diameter calibration mechanism.

To this end, we highlight the novelties of this paper. Firstly, we propose a new semicircle fitting based framework for robustly reconstructing dormant apple trees in 3D space using sequence of depth images. Secondly, we exploit a new error correction model that accurately estimates the diameter of each cross-section of a tree — even when the data is noisy. Lastly, we publish a new dataset consisting of several dormant apple trees collected from various orchards.

3. Dataset and Implementation Details

In Section 3.1, we explain our dataset of dormant apple trees as well as the data acquisition procedure. In Section 3.2, we discuss about the important parameters of the proposed framework, and describe the metrics used to evaluate the performance of our algorithm in Section 3.3.

3.1. Dataset and Data Acquisition

We use five trees from our dataset[1] collected from different orchards for testing. Out of these trees, there is an indoor laboratory tree namely, Indoor 1; while the other four trees are obtained from actual fields, namely, Outdoor 1, Outdoor 2, Outdoor 3, and Outdoor 4. For training, we use a different tree from our dataset, namely, Indoor 2. For each tree, we collect three types of information, namely, **Depth images**, **Labeled images**, and **Groundtruth measurements**.

3.2. Parameter Tuning Scheme

The performance of our algorithm depends on three important parameters, namely, Depth Threshold (τ), Similarity Metric between a pair of 3D point clouds, and Distance Threshold. **Depth Threshold (τ)** is required to set a bounding box on the object that is being reconstructed (as discussed in Section 4.1). We use τ to define the maximum depth beyond which all the depth values in the depth image are set to zero. The value of τ should be altered depending upon the position of the object from the sensor. In our experiments, we set the value of τ between 1m and 2m for different trees. **Similarity Metric** (explained in Section 5.2) is used to measure the similarity between a pair of aligned point clouds by employing some distance measures. Only those depth images are used for the final reconstruction which satisfy the pre-defined similarity criteria. **Distance Threshold (α)** refers to the distance from the sensor center to the tree trunk. In our experiments, we use α equals to 1.5m for indoor, and 1m for outdoor trees, respectively.

3.3. Evaluation Metric

To quantitatively measure the performance of our approach, we use two metrics, namely, *Branch Identification Accuracy* and *Confidence Value*. **Branch Identification Accuracy (BIA)** indicates the accuracy of modeling the primary branches [18]. It is computed as the percentage of the detected branches verified to be true over the actual number of ground truth branches of the tree. **Confidence Value** shows the percentage of the primary branches whose *Estimation Error* - calculated based on the absolute difference between the groundtruth diameter (obtained using caliper) and the estimated diameter - are within a certain threshold value ϵ.

4. Preprocessing a Tree Using Its Depth Images

In this section, we describe the preprocessing functions required for the sensor's raw data in order to make them easier to analyze within the proposed $3D$ reconstruction framework. Hence, the main topics addressed in this section are as follows. Section 4.1 discusses the necessary noise filtering procedure required to enhance the quality of the raw images, and Section 4.2 describes the skeletonization of tree structure. Section 4.3 explains the procedure of estimating centers and radii information required for fitting semicircle models of the 3D point clouds of a tree. Finally, Section 4.4 describes an empirical approach that models the errors between the calculated diameters of the primary branches obtained from the sensor's data and their corresponding groundtruth measurements. Fig. 1 highlights the overall preprocessing framework.

4.1. Filtering Noise from Raw Depth Images

One major challenge in accurate 3D reconstruction of the dormant apple trees is the presence of noise in the raw depth images, especially around the edge pixels of the tree. Thus, preprocessing becomes a crucial stage of our proposed algorithm. These operations include lens distortion removal, background subtraction, and filling up the gaps present in the raw data. A clean depth image after preprocessing stage is shown in Fig. 1(c).

Removal of Lens Distortion: To remove lens distortion that affects the accuracy of the reconstruction [17, 27], we calculate its parameter vector **k** by calibrating the camera, as in [26]. Subsequently, the undistorted depth map can be obtained as follows:

$$\mathbf{X}_t = \begin{bmatrix} 2k_3xy + k_4(r^2 + 2x^2) \\ k_3(r^2 + 2y^2) + 2k_4xy \end{bmatrix} \tag{1}$$

$$\mathbf{X}_u = (1 + k_1r^2 + k_2r^4 + k_5r^6)\mathbf{X} + \mathbf{X}_t \tag{2}$$

Where $\mathbf{X} = (x, y)$ is the input depth map, $\mathbf{X}_t$ is the tangentially undistorted map, $\mathbf{X}_u$ is the radially and tangentially undistorted map, $r^2 = x^2 + y^2$, and $\mathbf{k} = (k_1, k_2, k_3, k_4, k_5)$ is the parameter vector, where k_1, k_2 and k_5 are the radial distortion parameters; while k_3 and k_4 are the tangential distortion parameters.

Background Clutter Removal: Apart from lens distortion, raw depth images also contain a lot of clutter in the background that needs to be removed. For this purpose, we segment the foreground object from the background clutter by thresholding the pixel values of the depth image that are greater than a certain threshold (τ meters) to zero (see Section 3.2 for details).

Filling Up the Internal Holes: Finally, we apply morphological operations to fill up the small gaps present in the raw depth images, and eliminate the bays along the boundaries. We basically perform a closing operation, where the depth image A is first dilated, and then eroded with an all-one 3×3 structuring element B as follows:

$$A \bullet B = (A \oplus B) \ominus B \tag{3}$$

Where $\oplus$ and $\ominus$ denote the morphological operations, dilation and erosion, respectively.

[1]The dataset is now publicly available at `https://engineering.purdue.edu/RVL/WACV_Dataset`

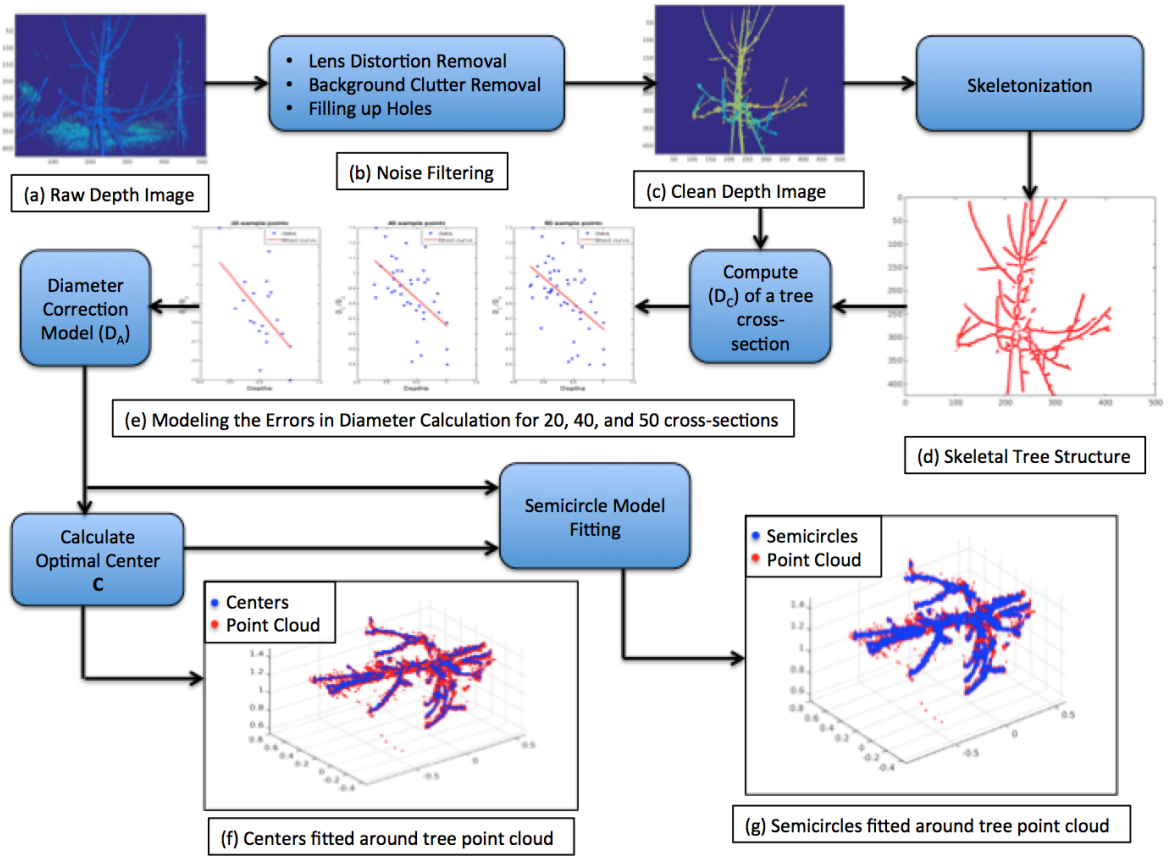


Figure 1: Preprocessing Framework

4.2. Extracting 2D Skeletons from Depth Images

After cleaning up the raw depth image, we extract its skeleton by employing a Voronoi Diagram technique, as in [24]. We provide the depth image **D** of a tree to the skeleton extraction algorithm F, and obtain the skeleton nodes S as well as their interconnections C (shown in Fig. 1(d)).

$$[S, C] = F(\mathbf{D}) \tag{4}$$

Finding the Edge Points of a Skeleton Node: Once we have extracted the 2D skeleton nodes and the interconnections between them, we aim at finding the edge points, indicated as p_1 and p_2 in Fig. 2 (a), that surround a skeleton node of the primary branch. In particular, for each pair of connected nodes in the skeleton forming a skeleton segment, we find the orientation of the skeleton segment by calculating the dominant direction **d** between a pair of skeleton nodes, s_1 and s_2, as follows:

$$\mathbf{d} = \frac{\langle x_1 - x_2,\ y_1 - y_2 \rangle}{\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}} \tag{5}$$

Where (x_1, y_1) and (x_2, y_2) are the pixel coordinates of a pair of skeleton nodes, s_1 and s_2, as shown in Fig. 2(a).

To this end, we refer to the region, that surrounds a particular skeleton node, and is bounded by its edge pixels, as the *cross-section* of the skeleton node. It is important to note that in the extracted 2D skeleton images, it is extremely difficult to distinguish between the overlapping branches. Based on empirical calculations, any pair of skeleton nodes whose 3D coordinates are more than 3 cm apart are treated as disconnected.

4.3. Constructing Semicircle Model of the 3D Point Cloud of a Tree from its Depth Image

Towards this direction, next we estimate the diameter and the center parameters of a cross-section.

Estimating the Diameter of a Cross-Section: We look for the edge pixels along the normal direction $\mathbf{d}_{\mathbf{norm}}$ of the dominant direction **d**. We compute $\mathbf{d}_{\mathbf{norm}}$ as follows:

$$\mathbf{d} \cdot \mathbf{d}_{\mathbf{norm}} = \mathbf{0} \tag{6}$$

For each skeleton node, we obtain a pair of edge points. We describe the procedure for finding an edge pixel that corresponds to a skeleton node in Algorithm 1.

After obtaining the edge pixels corresponding to a skeleton node, we use Equ. (10) to calculate the diameter D_C of

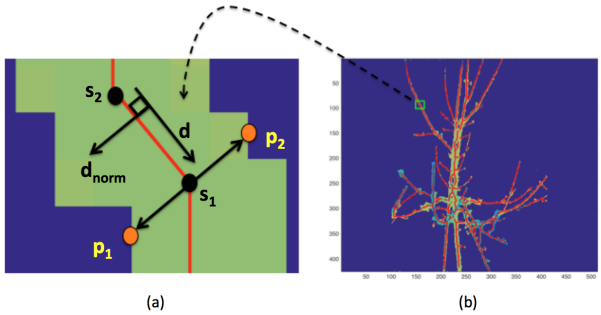


Figure 2: We show in (b) the depth image of a tree with its 2D skeleton superimposed, and a cross-section of the tree is highlighted with a green rectangle. The highlighted segment in green is magnified in (a) displaying edge pixel computation.

Algorithm 1 Edge Pixel Extraction algorithm

Step 1: Initialization

1) Input a single depth image, **D**.
2) Set a threshold = ϕ.
3) Initialize a counter, i = 1.
4) Compute dominant direction **d** between two skeleton nodes, (s_1, s_2), using Equ. 5.
5) Compute a direction, $\mathbf{d_{norm}}$, perpendicular to **d** using Equ. 6.

Step 2: Finding an Edge Pixel (p_1) of a skeleton node (s_1)

6) Compute $p_1 = p_1 + i * \mathbf{d_{norm}}$, where p_1 is the edge pixel computed by moving in a direction $\mathbf{d_{norm}}$ from s_1.
7) Compute Euclidean distance, **dist**, between the skeleton node $s_1 = (x_1, y_1)$ and edge pixel p_1.
8) Increment i.
9) Repeat Step 2 until $\mathbf{D}(p_1)$ is not zero and **dist** $\leq \phi$.

its cross-section. Next, we apply the "diameter-correction model" discussed in Section 4.4 on D_C in order to estimate the actual value of the diameter (D_A) at that particular node with depth value $d = \mu$ using (11). Finally, we obtain the radius of a cross section **r** as $\mathbf{r} = D_A/2$.

Estimating the Center of a Cross-Section C: We basically estimate **C** as follows:

$$\mathbf{C} = \mu + \mathbf{r}\nu \tag{7}$$

$$S_{3D}(n) = \mu K^{-1}\mathbf{S_n} \tag{8}$$

$$\nu = \frac{S_{3D}(n)}{|S_{3D}(n)|} \tag{9}$$

Where $\mathbf{S_n}$ is the n^{th} skeleton node with a depth value μ, K is the camera intrinsic matrix, $S_{3D}(n)$ is the 3D point corresponding to $\mathbf{S_n}$, and ν is the Kinect Observation Vector (KOV) at $\mathbf{S_n}$. Fig. 1 (f) displays the centers (shown in blue) fitted around the tree point cloud (shown in red).

Constructing Semicircle Model of the 3D Point Cloud of a Tree from its Depth Image: Once we estimate **C**, we use it along with the dominant direction **d**, the observation vector ν, and the radius **r** to obtain the optimized semicircle points (shown in blue in Fig. 1 (g)), as in [13], with an arc angle α that ranges from $-\pi/2$ to $\pi/2$. The final model provides the centers, radii, kinect observation vectors, and normals associated with each cross-section of the tree.

4.4. Diameter-Correction Model for Errors Obtained from Depth Images

Kinect sensors do not capture points that lie near the edge of the trunk and primary branches in the depth images. Consequently, there exists some errors between the groundtruth measurements of the diameters of the primary branches (D_A) and their corresponding diameter-measurements calculated from the depth image D_C. To measure D_A of a tree cross-section we use a caliper, and we calculate its corresponding D_C value from the depth image as follows:

$$D_C = \frac{d}{f}\|p_1 - p_2\| \tag{10}$$

Where d is the depth value at the center of a cross-section, f is the focal length of the sensor, and p_1 and p_2 are two edge points of the cross-section.

To model the error between D_C and D_A, we examine various cross-sections of a tree under different depth distances from the sensor. In particular, for each cross-section we record the depth value of its center pixel as well as the ratio between D_C and its corresponding D_A. In Fig. 1(e), we show the ratio values between D_C and its corresponding D_A (on the y-axis) under varying depth values (on the x-axis). It is observed that the ratio between D_C and D_A remains unchanged for varying number of samples; thus, confirming the robustness of the empirical model. From the figure, it can be inferred that as depth value increases, the error between D_C and D_A increases. To this end, we fit a linear model on the collected data, indicated in Fig. 1(e), as follows:

$$\frac{D_C}{D_A} = -0.95d + 1.6 \tag{11}$$

Next, we will use the obtained "Diameter-Correction model" to estimate the actual diameter value of a semicircle.

5. 3D Reconstruction of a Tree from its 3D Point Clouds

In this section, we provide a detailed explanation of the proposed framework for obtaining the 3D reconstruction of dormant apple trees. In particular, Section 5.1 describes the process of obtaining and aligning the 3D point clouds of a

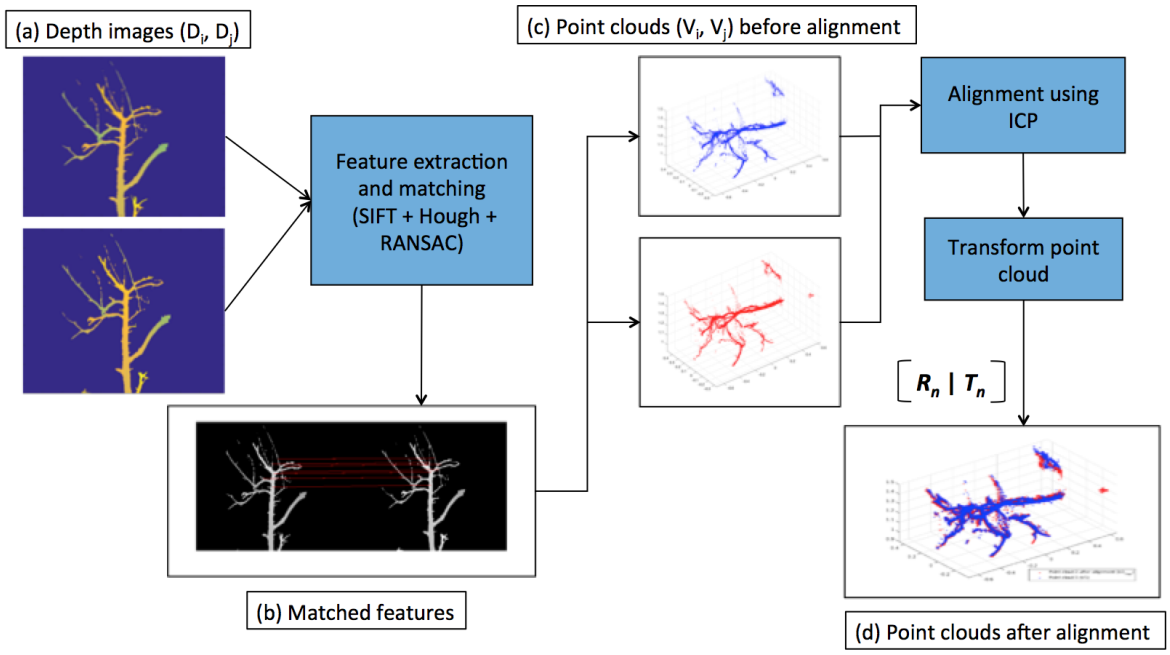


Figure 3: Alignment framework of a point cloud pair for the tree 'Indoor 1'

tree from its depth images. In Section 5.2, we discuss the proposed scheme for merging the aligned point clouds of a tree to obtain the final 3D reconstruction.

5.1. Alignment of 3D Point Clouds

The first step in the alignment process is to obtain the 3D point clouds (Fig. 3(c)) from the depth images (Fig. 3(a)) of a tree. Towards this direction, we first extract relevant feature points from the depth images of the tree, and then find reliable and robust corresponding feature points between consecutive pair of depth images. In particular, we use a combination of Scale Invariant Feature Transform (SIFT), Hough Transform, and RANSAC to extract the features, and find the corresponding matching points in a pair of depth images, as in [3]. An example illustration is shown in Fig. 3(b). Next, we convert the pair of depth images together with their matched features to obtain the corresponding 3D point clouds, as in [16], based on the following equation:

$$\mathbf{V}(\mathbf{u}) = D(\mathbf{u})K^{-1}\mathbf{u} \tag{12}$$

Where $\mathbf{u} = (x, y, 1)$ is an image pixel in the homogeneous coordinate system with a depth value $D(\mathbf{u})$, K is the camera intrinsic matrix, and $\mathbf{V}$ is the output 3D point cloud.

The goal now is to align the 3D point clouds in order to obtain the final 3D reconstruction of the tree. To align a pair of 3D point clouds, we use the Iterative Closest Point (ICP) algorithm [33] which takes a pair of point clouds $\mathbf{V_i}$ and $\mathbf{V_j}$ as input, and returns the rigid transformation that best aligns them, as follows:

$$[R, T] = ICP(\mathbf{V_i}, \mathbf{V_j}) \tag{13}$$

Where R and T are the rotation and translation matrices.

An initial alignment between the two tree point clouds, $\mathbf{V_i}$ and $\mathbf{V_j}$, is obtained by applying ICP on the point clouds of only the matched pair of points, $\mathbf{M_i}$ and $\mathbf{M_j}$. ICP generates a homography between the matched points, $\mathbf{M_i}$ and $\mathbf{M_j}$, giving us an initial estimate of the transformation, $[R_{init}|T_{init}]$, between $\mathbf{V_i}$ and $\mathbf{V_j}$.

Once we have an initial alignment between the point cloud pair, we rotate and translate $\mathbf{V_i}$ by R_{init} and T_{init} to obtain a transformed $\mathbf{V_i}$, referred to as $\mathbf{V_i^t}$. Next, we reapply ICP on $\mathbf{V_i^t}$ and $\mathbf{V_j}$ to get a more robust alignment, $[R_f|T_f]$. The final alignment, $\widehat{\mathbf{V_i}}$, can then be obtained by transforming $\mathbf{V_i^t}$ by R_f and T_f, and get its mapping in the space of $\mathbf{V_j}$, as shown in Fig. 3(d).

5.2. 3D Reconstruction of a Tree from its Aligned 3D PointClouds

To obtain the final 3D reconstruction of a tree, we merge a pair of aligned point clouds based on two alternative techniques, namely, **Incremental Model (IM)** and **Global Model (GM)**. In both approaches, we start first with a pair of 3D point cloud pair ($\mathbf{V_i}$, $\mathbf{V_j}$), then we transform $\mathbf{V_i}$ to the frame of $\mathbf{V_j}$, and measure the similarity and diversity between the pair of aligned point clouds based on distance metrics. In our framework, we use the Jaccard Similarity Index (**J**) as our similarity metric as it varies over a

(Tree ID, Number of Depth Images)	Incremental Approach				Global Median Approach				Global Mean Approach				KinectFusion	
	ϵ = 3 mm	ϵ = 5 mm	ϵ = 7 mm	BIA (%)	ϵ = 3 mm	ϵ = 5 mm	ϵ = 7 mm	BIA (%)	ϵ = 3 mm	ϵ = 5 mm	ϵ = 7 mm	BIA (%)	ϵ = 7 mm	BIA (%)
(Indoor 1, 30)	55.5	77.8	77.8	100.0	33.3	55.5	88.9	100.0	66.7	78.8	88.9	100.0	0.0	100.0
(Outdoor 1, 32)	85.7	85.7	100.0	100.0	85.7	100.0	100.0	100.0	85.7	100.0	100.0	100.0	0.0	100.0
(Outdoor 2, 33)	66.7	83.3	83.3	100.0	83.3	83.3	100.0	100.0	83.3	83.3	83.3	100.0	0.0	83.3
(Outdoor 3, 29)	50.0	100.0	100.0	100.0	25.0	50.0	75.0	100.0	25.0	25.0	75.0	100.0	25.0	100.0
(Outdoor 4, 51)	60.0	100.0	100.0	100.0	60.0	100.0	100.0	100.0	60.0	100.0	100.0	100.0	0.0	100.0
Mean	63.6	**89.4**	**92.2**	100.0	57.5	**77.8**	**92.8**	100.0	64.1	77.4	89.4	100.0	5.0	96.6

Table 1: This table displays the confidence values (%) in measuring the diameters with estimation error less than ϵ. For each tree, the branch identification accuracy (**BIA %**) using all the evaluated approaches is also provided.

Algorithm 2 3D Reconstruction of a tree using Incremental and Global approaches

Step 1: Initialization

1) Set $i = 1, j = 2$; where i and j are the indices for the previous and current point clouds as well as depth images.

2) Initialize $\psi_{inc} = \psi_1$, $\psi_{global} = \psi_1$. ψ_{inc}, the final 3D model using **IM**, is initialized to the semicircle fitted model of the first depth image. ψ_{global} contains the semicircle fitted models of all the similar 3D point clouds in the sequence. ψ_{global} contains only ψ_1 in the beginning.

Step 2: Basic Operations

1) Convert D_i and D_j to V_i and V_j. D_i and D_j represent 2D depth images. V_i and V_j are the corresponding point clouds.

2) Transform V_i to the frame of V_j.

3) Measure similarity between transformed V_i and V_j.

4) If similar = TRUE, build model ψ_j. If **Incremental**, go to Step 3, and if **Global**, go to Step 4.

5)If similar = FALSE, discard D_j, and increment j.

Step 3: Incremental Approach

1)Transform ψ_{inc} to the frame of V_j.

2) Save ψ_{inc} = Transformed ψ_{inc}.

3) Update ψ_{inc} = average (ψ_{inc}, ψ_j), and increment i, j.

4) Repeat Steps 2 & 3 until all the depth images in the sequence are visited.

Step 4: Global Approach

1) Save model ψ_j, $\psi_{global} = \{\psi_{global}, \psi_j\}$.

2) Save transformation, and increment i, j.

3) Repeat Steps 2 & 4 until all the depth images in the sequence are visited.

4) Find m = length(ψ_{global}), and transform $\psi_{global}(1)$, $\psi_{global}(2)$,....., $\psi_{global}(m-1)$ to the frame of $\psi_{global}(m)$.

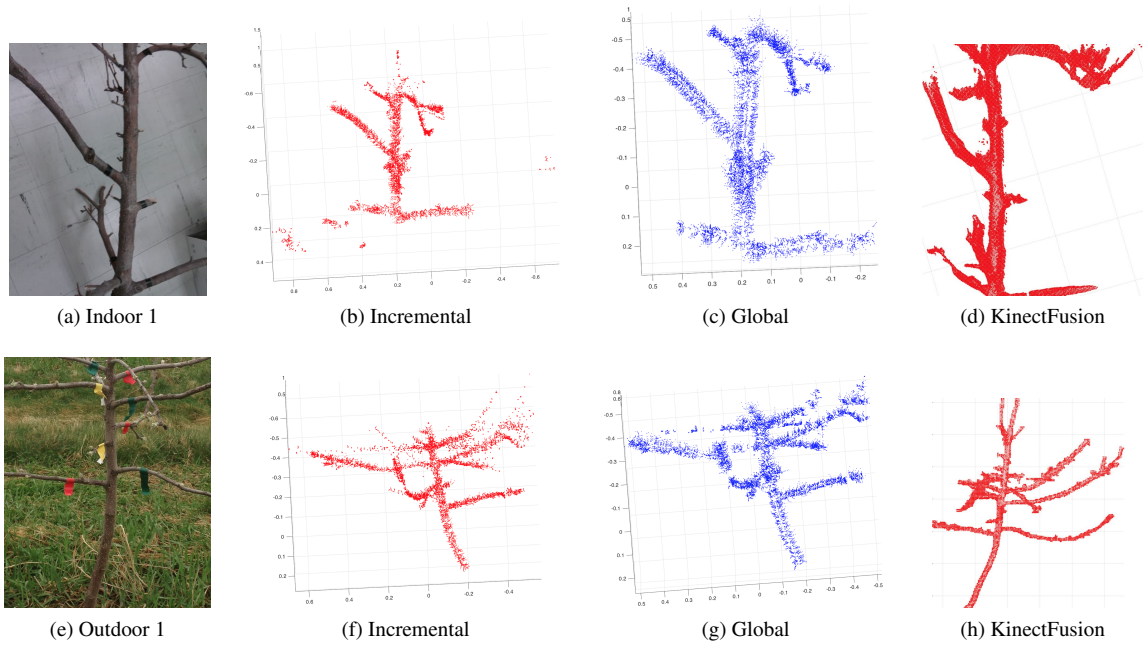
5) Create the final model by taking mean (GM_{avg}) or median (GM_{med}) of all the transformed point clouds.

larger range making it easier to discriminate the aligned point clouds. In case $\mathbf{V_j}$ and $\mathbf{V_{j+1}}$ are not similar, we discard $\mathbf{V_{j+1}}$, and examine the next point cloud in sequence. On the other hand, if a pair of aligned point clouds is similar, in **IM** we fit a semicircle model (ψ_i, ψ_j) of the point cloud pair, and compute their average model ψ_{inc}. Then, we align $\mathbf{V_j}$ with the next point cloud in sequence, $\mathbf{V_{j+1}}$, in an incremental manner, whenever similarity criteria are satisfied. In particular, we measure the similarity between the transformed $\mathbf{V_j}$ and $\mathbf{V_{j+1}}$, map ψ_{inc} to the frame of $\mathbf{V_{j+1}}$, compute the average of transformed ψ_{inc} and ψ_{j+1}, and update the previous average model. The incremental approach is described in Step 3 of Algorithm 2. In **GM**, we fit the semicircle model (ψ_i, ψ_j), and store it in a holistic model (denoted as ψ_{global}). We repeat the same process for (ψ_j, ψ_{j+1}), and store ψ_{j+1} in ψ_{global}. To this end, ψ_{global} contains the semicircle models of all the point clouds that satisfy the similarity criteria. To obtain the final reconstruction, we map the obtained models to the frame of last point cloud following a daisy chaining approach. In particular, we form the final 3D model of a tree either by computing the median of all the transformed point clouds (denoted as GM_{med}) or by averaging the models of the aligned point clouds (denoted as GM_{avg}). We highlight the entire process in Step 4 of Algorithm 2.

6. Experimental Results

In this section, we evaluate the proposed 3D reconstruction framework on each of the 5 test trees, mentioned in Section 3.1. We also compare our results with those obtained using the KinectFusion approach, as a baseline. In first row of Fig. 4, we demonstrate a groundtruth image of the tree 'Indoor 1', and the final 3D reconstructions using **IM**, **GM**, and KinectFusion based approaches. Same results for the tree 'Outdoor 1' have been provided in the second row of Fig. 4. Moreover, Table 1 shows the quantitative results of the proposed approaches. In particular, it is observed that the BIA metric for both the incremental and the global approaches provide a **100%** for reconstructing branches of all the trees in the dataset. On an average, the proposed approaches allow us to successfully detect about **100%** of the original tree branches. This indicates that in most scenarios, we can reconstruct all branches of the trees.

Moreover, quantitative evaluations of the diameter estimation accuracies show that the global and the incremental approaches significantly outperform the KinectFusion approach under different tolerances. On an average, we can estimate the diameters of the primary branches of all trees with confidence values of **77.8%** and **92.8%** (using the global median approach), and **89.4%** and **92.2%** (using the



(a) Indoor 1 (b) Incremental (c) Global (d) KinectFusion

(e) Outdoor 1 (f) Incremental (g) Global (h) KinectFusion

Figure 4: The first row shows the ground truth images of the tree 'Indoor 1', its 3D reconstruction using Incremental Model (IM), Global Model (GM) and KinectFusion (KF). The second row displays the same for the tree 'Outdoor 1'.

incremental approach) for tolerances of **5 mm** and **7 mm**, respectively. Taking all criteria into consideration, we conclude that the incremental approach is the best choice as it outperforms the other approaches for reconstructing all the trees in our dataset.

7. Conclusion and Future Work

In this work, we present a new method that aims at accurate 3D reconstruction of the trunk and primary branches of dormant apple trees based on an efficient semicircle fitting algorithm using a sequence of depth images. The proposed approach also allows us to estimate the diameter of a primary branch using a novel empirical model. The method has been tested on several trees under various indoor and outdoor environments for both qualitative and quantitative evaluations, and the results look promising. Our experiments show that the proposed method gives a better estimation of the diameters when compared with the results of KinectFusion. On an average, the results allow us to accurately estimate the diameters of **89.4%** of primary branches within an ϵ value of **5 mm**. We can also identify the branches of the trees in our dataset with **100%** accuracy.

In the future, we plan to extend the empirical error model in order to account for possible variations in both illumination and primary branch thickness. For that purpose, we would be including depth as well as color information of the images. We would also work towards designing a more robust evaluation metric which can handle the presence of false positives and false negatives in our final reconstruction results. We also plan to improve the computation time of our algorithm by employing parallelization.

7.1. Acknowledgment

We would like to acknowledge the United State Department of Agriculture (USDA) that provides the funding of this research. We would also like to thank Professor Peter Hirst's group from the Department of Horticulture and Landscape Architecture at Purdue University, Professor James Schupp's group from the Department of Horticulture at Pennsylvania State University, as well as the Bear Mountain Orchards at Aspers, PA, for their help in the tree scanning process and the development of the dataset. We would also like to acknowledge Mr. Bret Wallach from Vision Robotics Corporation for his valuable advises in the course of this research.

References

[1] B. Adhikari and M. Karee. 3d reconstruction of appletrees for mechanical pruning. 2012.

[2] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Robotics-DL tentative*, pages 586–606. International Society for Optics and Photonics, 1992.

[3] J. Blaber. A simple sift implementation with pose estimation. `http://www.mathworks.com/matlabcentral/fileexchange/45187-a-simple-sift-`

implementation-with-pose-estimation, 2014 (accessed October, 2015).

[4] M. Brophy, A. Chaudhury, S. S. Beauchemin, and J. L. Barron. A method for global non-rigid registration of multiple thin structures. In *Computer and Robot Vision (CRV), 2015 12th Conference on*, pages 214–221. IEEE, 2015.

[5] A. Bucksch and R. Lindenbergh. Campinoa skeletonization method for point cloud processing. *ISPRS journal of photogrammetry and remote sensing*, 63(1):115–127, 2008.

[6] A. Bucksch, R. Lindenbergh, and M. Menenti. Skeltre. *The Visual Computer*, 26(10):1283–1300, 2010.

[7] J. Cao, A. Tagliasacchi, M. Olson, H. Zhang, and Z. Su. Point cloud skeletons via laplacian based contraction. In *Shape Modeling International Conference (SMI), 2010*, pages 187–197. IEEE, 2010.

[8] K. N. Chaudhury, Y. Khoo, and A. Singer. Global registration of multiple point clouds using semidefinite programming. *SIAM Journal on Optimization*, 25(1):468–501, 2015.

[9] J.-F. Côté, J.-L. Widlowski, R. A. Fournier, and M. M. Verstraete. The structural and radiative consistency of three-dimensional tree reconstructions from terrestrial lidar. *Remote Sensing of Environment*, 113(5):1067–1081, 2009.

[10] Y. Cui, S. Schuon, S. Thrun, D. Stricker, and C. Theobalt. Algorithms for 3d shape scanning with a depth camera. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(5):1039–1050, 2013.

[11] N. Elfiky, S. Akbar, J. Sun, J. Park, and A. Kak. Automation of dormant pruning in specialty crop production: An adaptive framework for automatic reconstruction and modeling of apple trees. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 65–73, 2015.

[12] B. Gorte and N. Pfeifer. Structuring laser-scanned trees using 3d mathematical morphology. *International Archives of Photogrammetry and Remote Sensing*, 35(B5):929–933, 2004.

[13] I. Hameduddin. Rotate vector(s) about axis. http://www.mathworks.com/matlabcentral/fileexchange/34426-rotate-vectors-about-axis/content/rodrigues_rot.m, 2012 (accessed October, 2015).

[14] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, 2012.

[15] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox. Rgb-d mapping: Using depth cameras for dense 3d modeling of indoor environments. In *Experimental robotics*, pages 477–491. Springer, 2014.

[16] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011.

[17] S. Jin, L. Fan, Q. Liu, and R. Lu. Novel calibration and lens distortion correction of 3d reconstruction systems. In *Journal of Physics: Conference Series*, volume 48, page 359. IOP Publishing, 2006.

[18] 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*, 103:127–135, 2014.

[19] Y. Livny, F. Yan, M. Olson, B. Chen, H. Zhang, and J. El-Sana. Automatic reconstruction of tree skeletal structures from point clouds. *ACM Transactions on Graphics (TOG)*, 29(6):151, 2010.

[20] T. Mallick, P. P. Das, and A. K. Majumdar. Characterizations of noise in kinect depth images: a review. *Sensors Journal, IEEE*, 14(6):1731–1740, 2014.

[21] S. Meister, S. Izadi, P. Kohli, M. Hämmerle, C. Rother, and D. Kondermann. When can we use kinectfusion for ground truth acquisition? In *Proc. Workshop on Color-Depth Camera Fusion in Robotics*, volume 2, 2012.

[22] C. V. Nguyen, S. Izadi, and D. Lovell. Modeling kinect sensor noise for improved 3d reconstruction and tracking. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 524–530. IEEE, 2012.

[23] H. Roth and M. Vona. Moving volume kinectfusion. In *BMVC*, pages 1–11, 2012.

[24] Y. B. Sinai. Skeletonization using voronoi. http://www.mathworks.com/matlabcentral/fileexchange/27543-skeletonization-using-voronoi, 2010 (accessed October, 2015).

[25] J. Smisek, M. Jancosek, and T. Pajdla. 3d with kinect. In *Consumer Depth Cameras for Computer Vision*, pages 3–25. Springer, 2013.

[26] A. Staranowicz, G. R. Brown, F. Morbidi, and G. L. Mariottini. Easy-to-use and accurate calibration of rgb-d cameras from spheres. In *Image and Video Technology*, pages 265–278. Springer, 2014.

[27] R. Szeliski. Image alignment and stitching: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 2(1):1–104, 2006.

[28] G. K. Tam, Z.-Q. Cheng, Y.-K. Lai, F. C. Langbein, Y. Liu, D. Marshall, R. R. Martin, X.-F. Sun, and P. L. Rosin. Registration of 3d point clouds and meshes: a survey from rigid to nonrigid. *Visualization and Computer Graphics, IEEE Transactions on*, 19(7):1199–1217, 2013.

[29] A. Verroust and F. Lazarus. Extracting skeletal curves from 3d scattered data. *The Visual Computer*, 16(1):15–25, 2000.

[30] Q. Wang and Q. Zhang. Three-dimensional reconstruction of a dormant tree using rgb-d cameras. *ASABE Paper*, (131593521), 2013.

[31] T. Weber, R. Hänsch, and O. Hellwich. Automatic registration of unordered point clouds acquired by kinect sensors using an overlap heuristic. *ISPRS Journal of Photogrammetry and Remote Sensing*, 102:96–109, 2015.

[32] T. Whelan, M. Kaess, M. Fallon, H. Johannsson, J. Leonard, and J. McDonald. Kintinuous: Spatially extended kinectfusion. 2012.

[33] J. Wilm. Iterative closest point. http://www.mathworks.com/matlabcentral/fileexchange/27804-iterative-closest-point, 2013 (accessed October, 2015).