Branch Diameter Measurement System

Ryan Schofield and Cindy Grimm

***Abstract*—Ground truth measurements are important horticultural tools, but current methods of taking these measurements are slow and/or inaccurate. This paper proposes a system that uses two time-of-flight sensors, an RGB camera, a load cell, an Arduino UNO, and a contact pole to take accurate, automatic ground truth measurements for branch diameters and locations. The system uses a 6 step algorithm, which involves aligning with the branch, using optical flow for image segmentation, finding the branch diameter in pixels, making physical contact with the branch, and finally getting a branch diameter in inches. This system was tested on one branch of a lab proxy tree. The diameters found were all within 0.1 inches of the caliper measured diameter.**

I. INTRODUCTION

Ground Truth measurements are an important horticultural tool. Limb to trunk ratios are an important factor in pruning rules [1] and trunk diameter can be linked to shaking parameters for pistachio tree harvesting [2]. Ground truth measurements can also be used to validate 3D reconstructions [3].

While ground truth measurements are important, current methods for taking them are slow and/or inaccurate. Manual methods, such as calipers, are time intensive and prone to human error. Methods involving point clouds are noisy, incomplete, and computationally expensive [4]. Computer vision methods tend to underestimate the cross-sectional area of trunks [5]. These measurements are further complicated with the inconsistency of branch growth (angle, diameter, length, etc.).

This paper proposes a system that takes accurate, automatic ground truth measurements of branch diameters. The system uses an end effector that attaches to a UR5 arm and houses all necessary sensors. Using two time-of-flight (TOF) sensors, the end-effector is automatically aligned perpendicular to the branch being measured. A video is taken at this start position while moving downward (negative z-direction). Two frames are taken from the video to use for optical flow, which provides image segmentation. The segmentation is processed and turned into a mask that is analyzed to find the branch diameter in pixels. The end effector moves forward, making physical contact with the branch. This contact provides accurate depth data to the system because the UR5 arm position is known. This depth data is used with the mask to calculate the pixel width and eventually transforming the diameter in pixels to the diameter in inches. This system is verified on a lab proxy tree.

II. METHODOLOGY

A. Hardware Design

In the custom end effector (Figure 1) there are five main components: a) two TOF sensors, b) RGB camera, c) load cell, d) Arduino UNO, and e) contact pole. The pair of TOF sensors are used to align the contact pole with these branch. The sensors flank either side of the contact pole (upper left figure 1) providing a single beam of depth data for each sensor (see figure 2 Step 4).

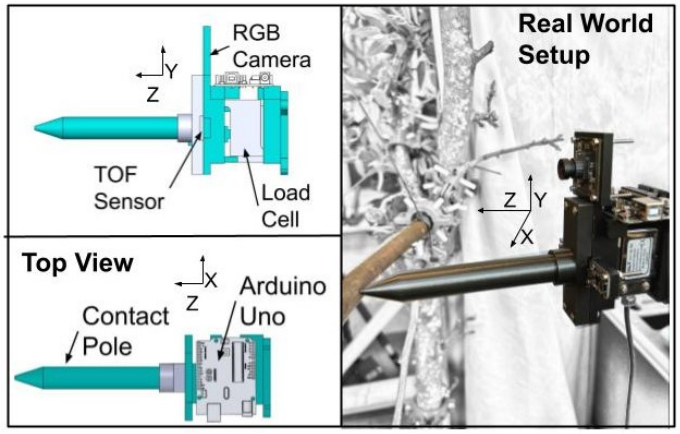

Fig. 1. Side view of end effector (upper left), top view of end effector (lower left), and real world setup (right), as well as the coordinate frame in each respective view.

The RGB camera provides video for optical flow and is placed in the center above (positive y-direction) the contact pole (upper left figure 1). After processing the optical flow mask, a diameter in pixels is returned.

The load cell is used to register contact with the branch and is placed centrally behind (negative z-direction) the contact pole (upper left figure 1). This provides accurate depth data to the system in the coordinate frame of the end effector.

The Arduino UNO is used to process the sensor data, such as depth data from the TOF sensors and force readings from the load cell. It is placed above (positive y-direction) the load cell (lower left figure 1).

Finally, the contact pole is placed in the center of the end effector (lower left figure 1) and makes contact with the branch when the camera is 6 inches from the branch. This known distance provides accurate depth information to convert image measurements to real world measurements.

B. Algorithm Overview

This algorithm has 6 main steps: 1) initial angle and position alignment, 2) correcting angle and position alignment, 3) perform optical flow, 4) determine diameter in pixels, 5) make contact with the branch, and 6) determine diameter in inches.

The goal of step 1 is to move to a position and rotate to an angle that shows the branch vertical in the image. As seen in

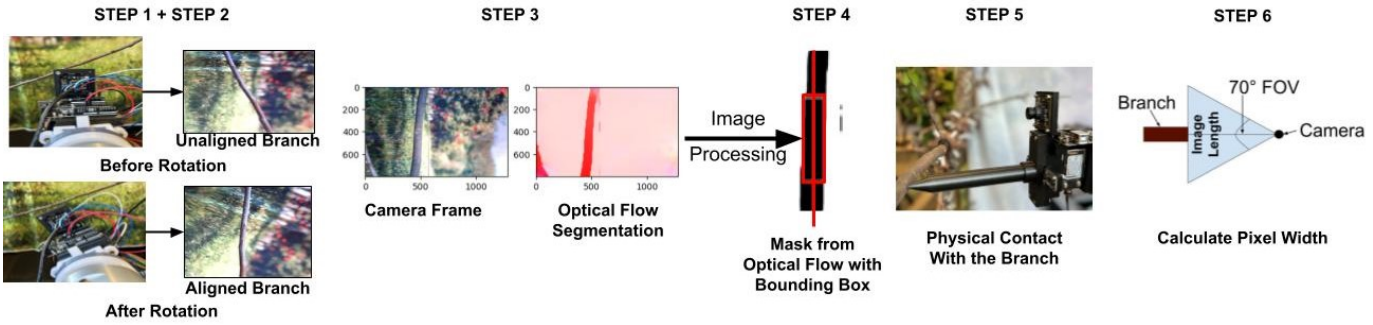

Fig. 2. The algorithm in images. Step 1 and 2 rotates the end effector so the branch is vertical in the image frame. Step 3 shows the image segmentation produced by optical flow. Step 4 shows the mask produced after processing the image segmentation and an example of a bounding box created. Step 5 shows the branch making contact with a branch. Finally, step 6 shows how the pixel image length is calculated with the known field of view (FOV)

the left-hand side of figure 2, before the end effector is rotated, the branch cuts diagonally across the image. After the rotation, the branch is vertical and centered. This is an important step to get an accurate diameter reading because when determining the diameter in pixels, the branch is assumed to be vertical. The goal of step 2 is to validate the branch angle and position. How the branch position and angle are found in each step is explained in the alignment section.

Once aligned, a 5-second video is taken as the arm moves downward (negative y-position). This video is used in step 3 to perform optical flow to get an image segmentation. The segmentation is processed into a mask that is used in step 4. Only the middle 25% of the mask is used. In step 4, the black pixels in each row are counted and the middle of the black section of each row is noted. This gives several diameter and middle line options. How these options are used are explained in the diameter measurement section.

In step 5, the contact pole is magnetically attached to the end effector and the arm moves forward until contact is registered by the load cell. This position is saved and used in step 6. The goal of step 6 is to calculate an accurate depth of the branch at the time the video was taken. Once a depth is known, the image length can be found because of the known field of view. This is used to calculate the diameter in inches, which is explained in the diameter measurement section.

The starting position of the UR5 arm is assumed to have the end effector at 0 degrees and in front of the branch of interest. The branch is assumed to have a 0 degree tilt in the plane.

C. Alignment — Step 1 and 2

As stated above, the branch angle and position are found in steps 1 and 2 of the system. Each step has a set of movements to perform while collecting TOF data. For the initial guess (step 1), the system moves up (positive y-direction) for 5 seconds at 0.1 m/s (approximately 0.5 meters). Figure 3 shows the TOF data when passing an aligned branch and an unaligned branch (in the image plane), with the lowest data readings marked with a circle. At the lowest data reading, the y position of the end effector is saved. For the aligned branch, the low readings occur at the same time. However, for the unaligned branch, the low points are far apart on the t-axis indicating that the end-effector is not perpendicular to the branch. To make an initial guess at the branch position, both lowest positions are averaged. To find the branch angle, the difference in sensor readings and the distance between the lowest positions are used, as seen in equation 1.

$$\theta_{desired} = \tan^{-1}\left(\frac{dist_{readings}}{dist_{sensors}}\right) \tag{1}$$

Where:

- $dist_{readings}$: is the distance between the y positions of the lowest readings for TOF1 and TOF2
- $dist_{sensors}$: is a constant value of the distance x distance between the sensors
- $\theta_{desired}$: is the angle that is equivalent with the branch angle

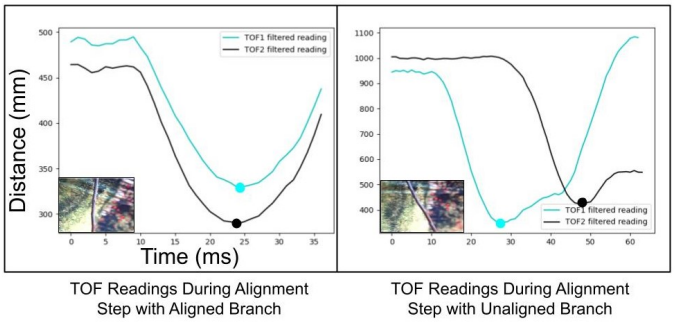

Fig. 3. The TOF data as the end effector passes an aligned branch (left) and an unaligned branch (right). The lowest point is marked as a circle for each time-of-flight sensor during each pass.

The validation of the branch angle and position (step 2), follows a different movement pattern than in the initial guess (step 1). The system rotates clockwise and counterclockwise in order to collect TOF data and end effector angle. Then the system moves up and down (y-direction) to collect TOF data and tool y position relative to the base. The movements described here provide similar data to that found in Figure

3. After the movements are complete, the rotation data is analyzed for the lowest TOF data and its corresponding angle. The position data is analyzed for the lowest TOF data and its corresponding y position. This gives two angles and two y positions. For the new branch angle and branch y-position, the angles and y positions are averaged respectively.

The goal of steps 1 and 2 is to find a position and angle that provides low positions that are aligned on the t-axis, similar to the left-hand graph in figure 3.

D. Diameter Measurement — Step 4 and 6

There are two frames used in the diameter measurements. The camera frame, which provides a diameter in pixels, and the world frame, which provides a diameter in inches. The pixel diameter is transformed into the inch diameter. To find the pixel diameter, the height of the mask provided by the optical flow is trimmed to the central 25% because this focuses on the branch section in front of the camera. The black pixels in each row are counted and considered the diameter for that row. The middle value of the black pixel section is considered the middle line for that row as seen in Figure 4. The 40% most seen diameters and middles are kept to eliminate any counts that come from an error. We propose 4 methods for calculating the diameter in pixels from these counts: 1) mean, 2) median, 3) W1, and 4) W2. These methods will later be compared to determine the most accurate method.

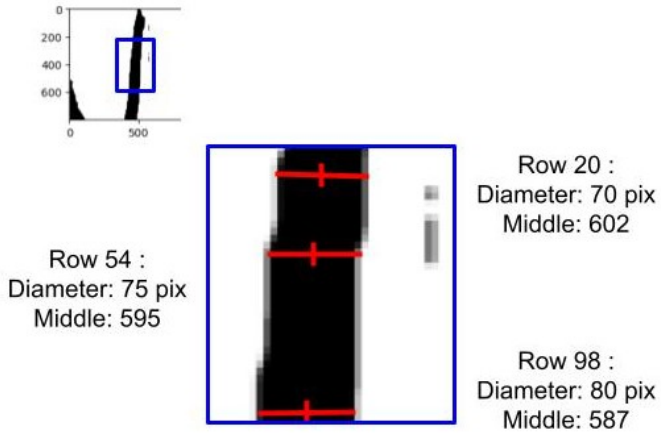

Fig. 4. Example branch mask with possible row counts turned into a diameter and middle line. All measurements are in pixels

The first method (mean) is to find the mean of the diameters that are kept. Similarly, the second method (median) is to find the median of the diameters that are kept. Method 3 and 4 are more complicated. They require several bounding boxes to be made. A bounding box and middle line can be seen in red in Figure 5. The box spans the height of the trimmed mask and the width of the diameter tested, and is centered on the middle line. A bounding box is made with every diameter and every middle line that was kept.

$$w_{inbox} = cs[rm] - cs[lm - 1] \tag{2}$$

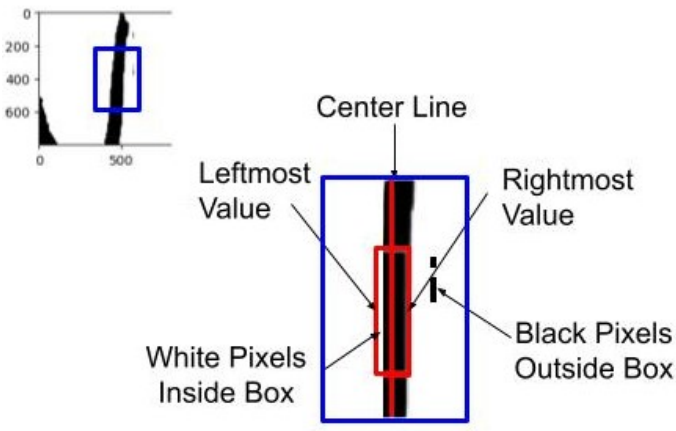

Fig. 5. A bounding box is seen in red with its center line. The vertical red line denotes the leftmost value and the vertical right line denotes the rightmost value.

$$b_{outbox} = ((c - (rm - lm + 1) * r) - cs[-1] - w_{inbox} \tag{3}$$

Where:

cs: is the cumulative sum
lm: is the furthest left column of the box being scored
rm: is the furthest right column of the box being scored
c: is the number of columns in the mask
r: is the number of rows being considered in the mask
w_{inbox}: is the number of white pixels inside the bounding box
b_{outbox}: is the number of black pixels outside the bounding box

The number of white pixels in each bounding box is calculated using equation 2. A cumulative sum of the columns in the trimmed mask is created and used in the equation. The cumulative sum holds the number of white pixels in the column to the left of the value that is being asked for. In this equation, it is the number of white pixels to the left of the rightmost value minus the number of white pixels to the left of the leftmost value. The cumulative sum is used again to calculate the number of black pixels outside the bounding box and can be seen in equation 3. In this equation, the number of pixels that exist outside the bounding box are calculated (in the parentheses) and the number of white pixels outside the box are subtracted.

$$score_{w1} = w_{inbox} + b_{outbox} \tag{4}$$

$$score_{w2} = 2 * w_{inbox} + b_{outbox} \tag{5}$$

For method 3 and 4, each bounding box is scored using the white values in the box and the black pixels outside the box. This is because the white pixels in the box and the black pixels outside the box are unwanted, and we are trying to minimize their presence. For method 3 (W1) the white pixels in the box and black pixels out of the box are weighted equally, as seen in equation 4. The diameter for this method is the diameter

from the lowest scoring bounding box. For method 4 (W2) the white pixels are weighted twice as much as the black pixels, as seen in equation 5. This is because it is more important to not have white pixels inside the box than having black pixels outside the box. Similarly, the diameter of this method comes from the diameter of the lowest scoring bounding box.

$$img_{inch} = 2 * dis_{video} * \tan(\theta_{camera}) \tag{6}$$

$$pix_{inch} = \frac{img_{inch}}{img_{pix}} \tag{7}$$

Where:

img_{inch}: is the width of the camera in inches
pix_{inch}: is the width of a pixel in inches
img_{pix}: is a constant value of the width of the images in pixels
θ_{camera}: is a constant value of horizontal field of view of the camera

To move from the pixel diameter to the inch diameter, the first step is to find the distance from the tree at the time of the optical flow. This is done using the tool z position at the time the optical flow video was taken and the tool z position at the contact point. It is known that at the contact point, the tool is 6 inches from the tree, so the difference of the two z positions plus 6 is the distance from the tree at the time of the optical flow. Using the distance from the tree at the time of the optical flow, the width of the image in inches can be calculated as seen in equation 6 and step 6 of Figure 2. The pixel width in inches can now be calculated as seen in equation 7. Using the four pixel diameters found in step 4, four diameters in inches can be calculated from the diameter in pixels and the pixel width in inches.

III. EXPERIMENTS

A. Lab Branch Experiment Setup

On a proxy tree in the lab, a branch set at 0 degrees was chosen. When measured using calipers, the branch's diameter was 0.71 inches. The arm was placed below the branch (negative y-direction) and 4 starting distances away from the branch (negative z-direction). Starting distances away from the branch were measured in inches from the camera and were 9, 8.75, 8.125, and 7.75. At each starting distance, 5 trials that gave diameters were run. Each trial gave 4 diameters, one for each of the 4 methods for finding diameters.

IV. RESULTS

A. Lab Branch Result

As seen in Table I, all the diameter methods produce an acceptable result. The maximum error that is seen across all trials is 0.086 inches (2.18mm). The best results come from the W2 method which has a maximum error of 0.035 inch (0.889 mm) and an average error of 0.0121 inch (0.31 mm).

Branch Diameter: 0.71				
	9in	8.75in	8.125in	7.75in
W1	0.056(0.030)	0.032(0.028)	0.071(0.015)	0.058(0.007)
W2	**0.003(0.007)**	**-0.014(0.004)**	**0.004(0.006)**	**0.024(0.007)**
Mean	0.032(0.027)	0.015(0.011)	0.047(0.013)	0.052(0.016)
Median	0.039(0.016)	0.015(0.018)	0.048(0.017)	0.050(0.013)

TABLE I
AVERAGE DIFFERENCE IN DIAMETER RESULTS IN INCHES (STANDARD DEVIATION) FOR EACH DISTANCE FROM THE TREE AND EACH DIAMETER METHOD.

REFERENCES

[1] J. R. Schupp, H. E. Winzeler, T.M. Kon, R. P. Marini, T. A. Baugher, L. F. Kime, and M. A. Schupp, A method for quantifying whole-tree pruning severity in mature tall spindle apple plantings. HortScience, 52(9), 1233–1240, 2017. https://doi.org/10.21273/hortsci12158-17

[2] T. Homayouni, A. Gholami, A. Toudeshki, L. Afsah-Hejri, and R. Ehsani, "Estimation of proper shaking parameters for pistachio trees based on their trunk size", Biosystems Engineering, Volume 216,2022,Pages 121-131,ISSN 1537-5110. https://doi.org/10.1016/j.biosystemseng.2022.02.008.

[3] S. A. Akbar, N. M. Elfiky, and A. Kak,"A novel framework for modeling dormant apple trees using single depth image for robotic pruning application", IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, pp. 5136-5142, 2016. doi: 10.1109/ICRA.2016.7487718.

[4] Y. Livny, F. Yan, M. Olson, B. Chen, H. Zhang, and J. El-Sana. "Automatic reconstruction of tree skeletal structures from pointclouds," ACM Trans. Graph., vol. 29, no. 6, pp. 151:1–151:8,Dec. 2010. http://doi.acm.org/10.1145/1882261.1866177

[5] L. Gonzalez Nieto, A. Wallis, J. Clements, M. Miranda Sazo, C. Kahlke, T. M. Kon, and T. L. Robinson, "Evaluation of Computer Vision Systems and applications to estimate trunk cross-sectional area, flower cluster number, thinning efficacy and yield of Apple", Horticulturae, 9(8), 880, 2023. https://doi.org/10.3390/horticulturae9080880