Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC-PapersOnLine 49-16 (2016) 248–253

Development of a Mobile Robotic Phenotyping System for Growth Chamber-based Studies of Genotype x Environment Interactions

D. Shah*, L. Tang*, Jingyao Gai*, R. Putta-Venkata*

**Iowa State University, IA 50011*
USA (e-mail: dsshah@iastate.edu, lietang@iastate.edu)

Abstract: To increase understanding of the interaction between phenotype and genotype x environment to improve crop performance, large amounts of phenotypic data are needed. Studying plants of a given strain under multiple environments can greatly help to reveal their interactions. To collect the labor-intensive data required to perform experiments in this area, a Mecanum-wheeled, magnetic-tape-following indoor rover has been developed to accurately and autonomously move between and inside growth chambers. Integration of the motor controllers, a robot arm, and a Microsoft Kinect (v2) 3D sensor was achieved in a customized C++ program. Detecting and segmenting plants in a multi-plant environment is a challenging task, which can be aided by integration of depth data into these algorithms. Image-processing functions were implemented to filter the depth image to minimize noise and remove undesired surfaces, reducing the memory requirement and allowing the plant to be reconstructed at a higher resolution in real-time. Three-dimensional meshes representing plants inside the chamber were reconstructed using the Kinect SDK's KinectFusion. After transforming user-selected points in camera coordinates to robot-arm coordinates, the robot arm is used in conjunction with the rover to probe desired leaves, simulating the future use of sensors such as a fluorimeter and Raman spectrometer. This paper reports the system architecture and some preliminary results of the system.

Keywords: 3D Reconstruction, Phenotyping, Mobile robots, Robot arm, Robot probing, Growth chamber

1. INTRODUCTION

In a world of changing climate and increasing world population, there is a great need to understand the interaction between genotype and phenotype in order to produce enough crop yield. Organizations such as the Royal Society of London (2009) and the FAO (2011) suggest a need for at least a 50% increase in food supply in the next half-century. This will not be achieved without a drastic change in the way we grow our food. Sustainable intensification, involving increasing the productivity of existing farmland while reducing negative environmental impacts, is promoted as one of the best ways – and some would say the only way - to achieve this.

One of the main methods to increase crop yield without increasing chemical use involves plant breeding techniques. Effective plant breeding requires in-depth data on plants' health and growth patterns, which are part of their broader phenotype, or physical characteristics. Many traits relating to growth, performance, and yield are complex traits under polygenic control (Pieruschka et al., 2012). Studying these traits using plants' phenotypes to understand how each strain behaves under various growing conditions is an important step toward improving the characteristics of a crop stock. This paper presents a technical solution to several issues relating to current phenotyping techniques.

Due to the large amount of manual labor required in traditional by-hand phenotyping methods, numerous studies and experiments have been exploring phenotyping methods that are based on images, often either RGB (red, green, and blue channels) or RGB-D (red, green, blue, and depth channels). For instance, one phenotyping technique uses infrared and depth images acquired with a CamCube ToF camera (Alenyà, et al., 2013). The method involves taking a general view of the plant, segmenting to find leaves which are suitable for probing, and then moving the cameras closer to the suitable leaf using a Barret WAM arm. This leaf is then probed with a sample cutting tool. This points to the possibility for an application of a wide variety of sensors. For instance, fluorescence imaging sensors could be placed on the end of the robot arm for investigating the fluorochrome chlorophyll, which is involved in crop yield (Chaerle et al., 2001). Numerous other sensors, such as near-infrared spectroscopy, can be applied to extract phenotypic data (Montes et al., 2007).

Other image-based phenotyping techniques use infrared stereo image sequences to extract depth and then segment the resulting data to extract parameters such as leaf area and number of leaves (Aksoy, et al., 2015). Tobacco plants can be stereo-imaged periodically, using a KUKA robot arm. The image pairs are then run through an OpenCV block-matching algorithm to extract depth information. Next, the images are segmented to distinguish each leaf. Leaf area is found by ellipse-fitting each leaf, and the number of leaves is compared to the ground truth obtained via human measurement. Since these methods used fixed plant imaging positions, the methods are mainly suitable for stationary plants and a stationary robot arm, limiting the experiment to one growth environment. In a growing area larger than the reach of the robot arm, this approach also requires conveyance of the plants out of their growth environment, introducing other stress factors outside

10.1016/j.ifacol.2016.10.046

of the designed environment. Azzari et al. (2013) fused several point clouds, sometimes as many as 2000 point clouds per plant, to reconstruct the plant for extraction several pieces of information, including volume and allometric relationships. Point clouds were attained through manually moving their first-generation Kinect (v1). Chaivivatrakul et al. (2014) fused several point clouds into one 3d reconstruction and then extracted traits of corn plants such as leaf area, leaf length, and stem diameter. This method didn't consider the case where multiple plants are in view. Finally, phenotyping can also be done in the field (Klodt, 2015). By taking a pair of images of the same grapevine plant, on several different days, some phenotyping can be done automatically. Once image pairs are acquired, they can be rectified to extract depth information, and finally segmented to find leaf and stem areas. This method did not provide an automated image capturing technique, and did not provide a framework for integration with other sensors for monitoring plant growth.

The solution presented in this paper required a minimal amount of labor during run-time, and can be extended, by adapting established phenotyping and plant-breeding to the techniques, to track plants that are growing in multiple growth environments concurrently. This solution involved an autonomous rover equipped with a Universal Robots UR 10 (UR10) robot arm, an industrial computer, a Microsoft Kinect (v2) sensor, and a rover base. The system was self-powered and required no wires to the outside world. The rover was equipped with a 120V power supply capable of powering numerous auxiliary sensors. This system allows for attachment of plant monitoring equipment, as illustrated in a small proof-of-concept experiment. The rover autonomously moved to the region representing the desired growth chamber in a setup mimicking Iowa State University's future Enviratron plant growth facility. This facility will contain several growth chambers with a robot vestibule in front of each chamber door. Once at the destination, the system probed several plants in the chamber with a rigidly-mounted steel rod that simulates other sensors that need to be placed at a certain distance and with a specific orientation to plant leaves.

2. THE PHYSICAL SYSTEM

2.1 Background on Hardware and Design

The Enviratron Rover aims to be a tool which can be used to autonomously gather feedback on plants which are simultaneously growing in multiple environments, from sensors as varied as traditional cameras, Raman spectrometers, thermal imagers, and fluorescence monitoring systems. The final application consists of eight growth chambers, arranged in a grid pattern (Fig. 1). To reach these goals, the rover needs to be mobile, accurate at positioning sensors, and able to navigate the environment it is placed in (roughly 120 cm in the direction perpendicular to travel), all with extremely high reliability and repeatability. Additionally, making the rover autonomous has the potential to allow for data to be collected at precise intervals of time, with significantly lowered per-data labor requirements for the researchers involved. It is desired that the system can run without human input for a whole day, or 8 hours.

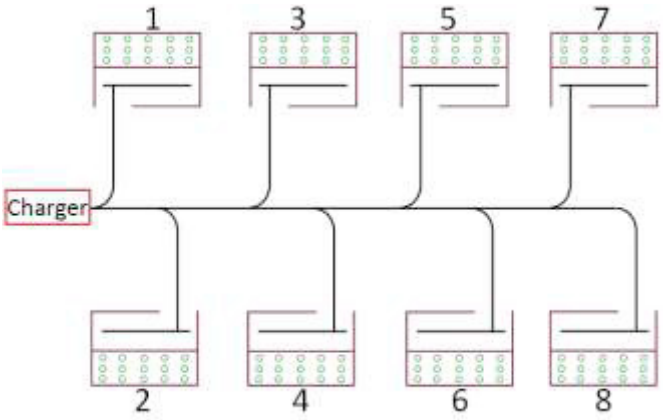

Fig. 1. The Chamber Arrangement.

The scope of this paper is to present the system in an intermediate stage where automation is achieved for each individual task. Our setup is equivalent to assuming the growth chamber vestibule is opened, and the material separating the vestibule from the growth area is already removed. Human input was only required to select, using a PC, a point on the plant to image, and the human told the rover when to shift sideways to view another plant. The procedures required to get Raman spectrometer readings share very similar sub-tasks, such as locating the plant and calculation of the surface normal.

Several commercial systems could be repurposed and programmed for this task. However, each have significant drawbacks. For instance, the Segway 440 Flex Omni has omnidirectional control, can interface with external hardware, and is specified to carry payloads as high as 450 kg. However, it has minimal support for large external electronics such as the UR10 and the base system is quite costly. Another system, the Neobotix MMO-550, is omnidirectional and has support for the UR10, sensors and sensors such as SICK NAV350, and can be controlled using the open-sourced Robotic Operating System (ROS). However, its uptime is quoted at 3 hours, there is limited additional space for sensors and electronics, and is also quite costly. Finally, the Ridgeback, sold by Clearpath Robotics, fits the design requirements but is quite costly and is large for our application.

It should be noted, however, that, to the best of our knowledge, no use of a rover for autonomous imaging of plants in multiple environments has been reported in the literature. These previously-mentioned commercial systems are only collections of multi-purpose hardware which we merely claim *could be repurposed*, i.e. modified and put to use, for the current novel task.

2.2 System Hardware

The rover proposed in this paper (Fig. **2**) combines many features which are desirable in research applications similar to

this, and only a brief overview of these features will be explained in this section.

Fig. 2. The Enviratron Rover.

The main frame of the rover uses the T-slotted aluminum building system sold by 8020 Inc. (Columbia City, Indiana, USA). This makes the overall system architecture extremely modular and adaptive, and also provides natural ease of maintenance. The system was modeled entirely in SolidWorks prior to build. There was one Roboteq FBL2360 motor controller for each side (Left and Right) each controlling two Midwest Motion Products MMP BL58-412F-48V GRA60-032 brushless DC motors. Each motor drove one 6" Mecanum Wheel HD, purchased through AndyMark. Four dust and water-resistant Roboteq MG1600 magnetic tape sensors were used for guidance and simple localization. A Spektrum DX6i Transmitter was used for remote control during development. One lighted hard-wired emergency stop was placed on each side for safety. A Meanwell TS1000 DC/AC inverter powered external 120 V outlets as well as the three core components of the imaging and probing system: a Logic Supply ML400G-30 industrial computer, Microsoft Kinect for Windows V2, and Universal Robots UR 10 robot arm. Finally, the whole system is powered by a 100Ah battery from AA Portable Power Corp (also known as batteryspace.com). This battery was chosen for its high energy storage capacity and appropriate balance of features, safety, and pricing. The overall system, excluding the UR10 arm to allow comparison with previously-mentioned commercial systems, had roughly half the cost of the least-expensive commercial system.

The Mecanum wheels used are similar to those analyzed by Gfrerrer (2008). In the ideal Mecanum wheel, force between the wheel and the ground only occurs along a vector parallel to the axis of the single roller which is in contact with the ground at that instant. Each roller's axis is rotated 45 degrees from the motor's axis. The wheels make contact with the ground in a shape described as an "O", a configuration often termed "O from below" in the robotics community. If the wheels are installed such that this does not occur, the stability was found to be poor, especially for lateral motion.

3. SOFTWARE

3.1 Overview

Our task mainly involves: localization of the rover, control of the motion of the rover, building knowledge of the growth chamber's contents, and positioning the sensor. The way we integrated our hardware allowed compartmentalization of these tasks, all of which can be controlled by the "brain", which is the PC (Fig. **3**).

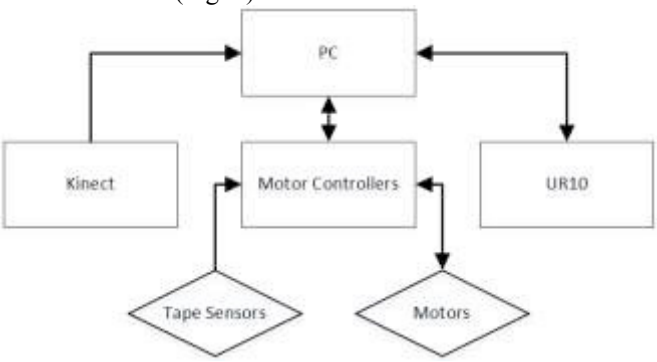

Fig. 3. Software Hierarchy. Line denotes a communication line, and arrows show direction packets are sent.

Various communication protocols and technologies were leveraged. Three tape-sensor data streams were sent to the motor controllers using Pulse Width Modulation (PWM), and one magnetic sensor used Roboteq's proprietary RoboMag protocol, allowing marker information to be transmitted easily. Motor position was implicitly sent to the motor controller via the built-in Hall-effect sensor. Roboteq's proprietary implementation of a PID closed-loop speed controller was leveraged by our custom scripts to control each motor during each scan cycle. The two motor controllers communicated using Roboteq's proprietary implementation of the Controller Area Network (CAN) protocol. Our custom PC code was all written in C++. The PC modifies and reads the motor controllers' internal variables using serial (RS232) communication. The PC gets current pose (position and orientation) information and sends desired poses to the UR10 via TCP/IP (Ethernet) communication. Kinect information is read using a USB 3.0 connection.

3.2 A Brief Note on Control Strategy

Although mathematical proof of stability for dynamic systems is quite common in the controls literature, that is not the focus of the present work. System modelling has been omitted for the time being and will likely be presented in a later report. Rather, this paper aims to present a working, intuitive, and adaptable framework for mobile manipulator implementation. The ideas presented could be applied to much wider classes of robots. Although it is dynamically-loaded, the system presented requires the controller to know relatively little about its system in order to function.

All inputs and real-time commands are fed into the leading motor controller which then commands the other "follower" motor controller. The motor controller's internal scripting has a large number of internal functions, but can only process Booleans and integers and has no ability to add libraries. This is generally not a limitation if clever coding practices are used. To provide the system with knowledge of its location, we have implemented a concept equivalent to tracking the rising edge of strategically-placed markers (any "upside-down" magnetic tape is interpreted as a marker by the MG1600's). A marker on one side of the track makes the controller "look" for rising edges on the other side of the track. The number of rising edges found during one "look" interval is stored as the last known location in the motor controller. To determine when the rover should turn off of the main track, we checked whether we counted an even or odd number of markers and compared with the current desired chamber. An example of this logic, for an odd chamber number, is presented in Fig. 4. In addition to information on the most recent marker count, representing the current magnetic-tape branch, the system can recognize the four-way intersection inside each individual chamber that corresponds to the leftmost plant. This provides requisite low-level localization. Further localization within the chamber will result from feedback from the Kinect sensor.

```
MarkerCopy = Marker
While MarkerCopy > 2
        MarkerCopy = MarkerCopy – 2
If MarkerCopy = 1 & Marker = DesiredChamber
        Follow Left Track
```

Fig. 4. Psuedocode for Track Choice.

The rover has two modes for guidance: one-sensor proportional control and two-sensor proportional control. The strategy can be boiled down to the following formulas.

1-sensor control:

$$V_{DOT} = V_{DEF} * F \quad (1)$$

$$F = \begin{cases} 0.5 \; if \; on \; turns \\ 0.75 \; if \; left \; tape \; detected \; (i.e. "look") \\ 1 \; else \end{cases} \quad (2)$$

$$V_{STE} = K_1 * T_1 \quad (3)$$

2-sensor control:

$$V_{DOT} = V_{DEF} * F \quad (4)$$

$$F = \begin{cases} 0.5 \; if \; on \; turns \\ 0.75 \; if \; left \; tape \; detected \; (i.e. "look") \\ 1 \; else \end{cases} \quad (5)$$

$$V_{STE} = K_2 * (T_1 + T_2) \quad (6)$$

$$V_{PERP} = K_2 * (-T_1 + T_2) \quad (7)$$

Where V_{DEF} is the default velocity, V_{DOT}, V_{STE}, V_{PERP} terms denote velocity in the direction of travel (DOT), steering, and perpendicular to the DOT, respectively. F is a scaling factor, K_1 and K_2 are proportional gains, T_1 is the reading from the tape sensor on the leading edge of the robot (i.e. the DOT), and T_2 is for the trailing edge. This is makes the robot correct the difference in the Tape readings by moving perpendicular to DOT; if the robot is moved forward both sensors will read opposite signs and strengthen the feedback. Similarly, an improperly-oriented robot, i.e. one that is not "pointing along the track", will have both sensors have the same sign and the steer command will be strengthened. The three velocity terms are summed appropriately for each wheel (c.f. AndyMark), and the results are sent to the individual motors.

3.3 Pre-Processing of Kinect Depth Data

The imaging and probing system has to be inherently robust to varying leaf size, stalk height, and plant type. Two approaches have been considered: 1) the trivial case of hard-coding robot-arm positions to several generally-desired poses, such as "front view" and "top view". 2) The general case of calculating desired pose based on knowledge of the sensor being used and the current arrangement of plants. For this study, we focused on the second, more general case.

Researchers have proposed using many 3D reconstruction methods on plants including structured light (Nguyen, 2015), time of flight (ToF) (Alenyá, 2013), and stereo reconstruction (Biskup, 2007). The Kinect V2 sensor was chosen for this study due to its affordable price, useful features and specifications (c.f., Butkiewicz 2014), and well-documented application program interface (API). An additional advantage of the Kinect V2 is that its Software Development Kit (SDK) provides reasonably accurate reconstruction sample code, termed KinectFusion that integrates easily into custom applications (Izadi et al., 2011, and Microsoft, 2016).

Before being able to use the Kinect's depth information to accurately position a robot arm, the mapping between the camera's coordinate system and the robot arm's end-effector's coordinate system must be determined. (An overview of camera calibration can be found in the seminal Tsai and Lenz paper (1989)). This mapping can be completely defined by a rotation followed by a translation, which can be resented by an affine transformation:

$$P_E = H_{CE} * P_C \quad (8)$$

Where P_E denotes a point in robot end-effector coordinates [m], P_C is a point in camera coordinates [m], and H_{CE} is the transformation matrix from camera coordinates to the end-effector coordinates.

This calibration matrix was found using Christian Wengert's add-on (Wengert) to the Camera Calibration Toolbox (Bouguet, 2015) for MATLAB. The H_{CE} for our setup was approximately

$$H_{CE} = \begin{bmatrix} 1 & 0 & 0 & 0.0540 \\ 0 & -1 & 0 & 0.1026 \\ 0 & 0 & -1 & 0.0825 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (9)$$

Feeding an unfiltered depth image to the KinectFusion algorithm was found to lead to a gradual erosion of the leaves, stalk, and stems of the plant, resulting in unusable meshes. However, this off-the-shelf algorithm was found to be very effective if an appropriately-filtered depth image was instead passed to the algorithm. The number of voxels that are tracked are limited, and unstable voxels are filtered out, to allow the algorithm to run in real time. In our application the table and

walls surrounding the plant are far more stable than the pixels corresponding to the thin-stemmed, thin-leaved plants.

We wanted an algorithm that was computationally efficient enough to allow quick reconstructions, and knew the Kinect V2 API has an accurate mapping between the color camera and the depth camera. However, not every depth pixel has a corresponding color pixel due to physical properties of the sensors. We set depth pixels without an RGB counterpart to zero. Incoming images from the Kinect were put into a data structure that is effectively an RGB image. This representation is known to be sensitive to lighting changes. There are many methods for dealing with this issue. Yang and Waibel, for instance, found that human faces were clustered in what they term chromatic color space (Yang 1997). We are mainly interested in plants, which are generally diffuse green and brown objects, so we chose to convert our RGB input image into HSV. After conversion to HSV, an experimentally-determined threshold was applied to each depth pixel. Let D_{ij} (T_{ij}) represent the pixel in row i row and column j of the R x C depth image (thresholded depth image), corresponding to a region of the HSV image as determined by the Kinect API's mapping. We have, for $1 \le i \le R \; and \; 1 \le j \le C$,

$$T_{ij} = \begin{cases} D_{ij} \; if \; V_{ij} < 140 \\ 0 \; else \end{cases} \quad (10)$$

The value of 140 conservatively thresholds out many extremely bright pixels, such as the walls of the growth chamber. Next, a 2x2 rectangular structuring element was used to morphologically dilate the image (Jain, 1995), conservatively removing many noisy elements. This resulting thresholded and dilated depth image, which contains the plant pixels plus some other pixels, is passed to the KinectFusion algorithm with default settings except for expanded voxels in the camera's left-right direction, which it labels as X. The goal is to probe leaves close to the front of the plant, so Y and Z are currently not very important. Most settings were found to have minimal effect on the results. An example fusion image is shown in Fig. 5. Finally, the mesh is processed by the mesh probing algorithm.

Fig. 5. A sample of fused image

3.4 Mesh Probe Algorithm

Given an initial pose of the robot, defined by position $P = [p_x, p_y, p_z]^T$ and rotation vector $r = [r_x, r_y, r_z]$ in the robot base coordinates, we desire the transformation matrix from the end effector to the base. This is given by

$$H_{EB} = \begin{bmatrix} R & P \\ \mathbf{0} & 1 \end{bmatrix} \quad (11)$$

The rotation matrix R is defined as

$$R = [A_x, A_y, A_z] = \begin{bmatrix} c + r_x^2 v & r_x r_y v - r_z s & r_x r_z v + r_y s \\ r_y r_x v + r_z s & c + r_y^2 v & r_y r_z v - r_x s \\ r_z r_x v - r_y s & r_z r_y v + r_x s & c + r_z^2 v \end{bmatrix} \quad (12)$$

Where $c = \cos(\theta)$, $s = \sin(\theta)$, $and \; v = 1 - \cos(\theta)$. See Craig (2009) for discussion.

and $\mathbf{0} = [0, 0, 0]$. To convert user-selected points P_{Ci}, $1 \le i \le 3$, from camera coordinates into base coordinates P_{Bi}, $1 \le i \le 3$, we used

$$P_{Bi} = H_{EB} * H_{CE} * P_{Ci} \quad (12)$$

Next, we determined the normal to three user-selected points (the normalized cross product between [P_{C2}-P_{C1}] and [P_{C3}-P_{C1}]) in order to calculate the desired end-effector coordinate system axes A_X, A_Y, A_Z. Two logical constraints were added to our system. Az should align with the surface normal N and the Kinect should be level with the ground. Since our probe end-effector is orthogonal to our end-effector's XY plane, i.e., it extends in the Az direction, we solved the following equations for A_X, Ay and A_Z:

$A_z = -N$, $A_x \cdot N = 0$, $A_{xz} = 0$, and

$$A_y = A_z \times A_x \quad (13a,b,c)$$

The position the end-effector moved to, arbitrarily chosen as P_{C1} from the 3 user-defined points, is a translation from the actual leaf position. Our probe stick was offset from the tool center point in the direction of the tool's x-axis, and was orthogonal to the tool's XY plane. Thus, the end-effector position, in robot base coordinates, was defined by:

$$P = P_{C1} + L * N - r * A_X \quad (14)$$

Where L is the length of the probe stick and r = radius from the tool center point to the thin probe. Next, we determined the rotation vector to send to the arm by solving equation 12 for r_x, r_y, and r_z. Finally, the calculated coordinate values were sent to the UR10 to probe the leaf.

4. EXPERIMENT

This section presents an experiment demonstrating current effectiveness of the proposed system. The UR 10 arm was commanded to probe two plants (one artificial Silk Dracaena plant - VCK8023, from artificialplantsandtrees.com - and one real Ficus plant), on three separate leaves, five times each (i.e. 30 separate probings). Both plants were placed on a 73 cm-high table (roughly the height that the UR base is at), around 1m apart. During normal conditions, the UR arm would extend roughly 80 centimeters in its Y direction for its end-effector to hit a desired leaf. One small piece of blue painters' tape (~5 mm square) was placed in the middle of each desired leaf.

Before each trial, the rover was set up at the edge of a track with complete magnetic tape as would be in a setup with one chamber. The rover entered the "chamber" and stopped at the intersection. A user clicked a button on the PC's custom user interface (UI), commanding the rover to shift sideways. There was only one issue with the tape-following navigation. After trial 3, one sensor reported magnetic tape in an un-taped portion of concrete. The sensors were calibrated with the Roboteq utility and the experiment proceeded as planned. As the lab currently lacks access to a 2D robot-tracking setup, the precision of this tape-following is omitted for this report.

Once the desired plant was in view, the user initiated our filtered Kinect Fusion algorithm. The UR10 moved to view the desired leaves, and the mesh was saved. Three points near the bottom-left corner of the tape markers were located in the MeshLab software, and stored in .txt file. The user clicked on our UI to probe the plant, the coordinates were sent through our "mesh probe algorithm", and the UR 10 approached the leaf with the probe. Using the UR 10's touchscreen user interface, we found ground truth by translating the end-effector until the probe hit the bottom-left corner of the tape markers. A summary of results is presented below.

Table 1. Euclidian Distance Errors [mm].

		Trial					
Plant	Leaf	1	2	3	4	5	Avg.
1	1	29.7	27.7	27.4	27.9	26.6	27.9
1	2	22.9	23.5	22.0	24.5	26.9	24.0
1	3	20.7	33.9	20.9	25.7	30.3	26.3
2	1	18.6	16.9	8.5	24.0	19.6	17.5
2	2	27.4	34.3	29.5	30.3	32.6	30.8
2	3	34.7	33.0	27.1	35.1	31.9	32.4
						Avg.	26.5

5. CONCLUSIONS

For some precise applications, the results indicate that further refinement would be needed. Future work includes reducing these errors. The system may be highly sensitive to camera calibration, requiring a calibration after even a slight change in camera orientation relative to the end-effector. Otherwise, improvements include more sophisticated pre-processing of the depth data or refining the KinectFusion routines. Use of another highly accurate distance sensor to augment the rough, real-time Kinect algorithm is one alternate solution. Finally, we aim to make the system fully autonomous.

REFERENCES

Aksoy, E. E., Abramov, A., Wörgötter, F., Scharr, H., and Fischbach, A.. (2015). Modeling leaf growth of rosette plants using infrared stereo image sequences. *Computers and Electronics in Agriculture,* 110, 78-90.

Alenyà, G., Dellen, B., Foix, S., Torras, C.. (2013). Robotized Plant Probing: Leaf Segmentation Utilizing Time-of-Flight Data. *Robotics & Automation Magazine, IEEE* 20(3), 50-59.

AndyMark http://files.andymark.com/MecanumWheelSpecSheet.pdf

Azzari, G., Goulden, M.L., Rusu, R.B. (2013). Rapid Characterization of Vegetation Structure with a Microsoft Kinect Sensor. *Sensors*, 13, 2384-2398.

Biskup, B., Scharr, H., Schurr, U., Rascher, U. (2007). A Stereo Imaging of Plant Canopies. *Plant, Cell and Environment*, 30, 1299-1308.

Bouguet, J.-Y. (2015). Camera Calibration Toolbox for Matlab, http://www.vision.caltech.edu/bouguetj/calib_doc/

Butkiewicz, T. (2014). Low-cost coastal mapping using Kinect v2 time-of-flight cameras. *Oceans - St. John's*, 2014, St. John's, NL, pp. 1-9.

Chaerle, L., Van Der Straeten, D. (2001). Seeing is believing: imaging techniques to monitor plant health. *Biochimica et Biophysica Acta,* 1519, 153-166.

Chaivivatrakul, S., Tang, L., Dailey, M., Nakarmi, A. (2014). Automatic morphological trait characterization for corn plants via 3D holographic reconstruction. *Computers and Electronics in Agriculture.* 109, 109-123.

Craig, J. (2009). *Introduction to Robotics: Mechanics and Control*. Pearson, New York.

Food and Agriculture Organization of the United Nations. (2011). Save and Grow: A policymaker's guide to the sustainable intensification of smallholder crop production. FAO, Rome.

Gfrerrer, A. (2008). Geometry and kinematics of the Mecanum wheel. *Computer Aided Geometric Design,* 25, 784-791.

Izadi, S.; Kim, D.; Hilliges, O. (2011). 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, Santa Barbara, CA, USA, 16–19 October 2011.

Jain, R., Kasturi, R., and Schunck, B. (1995). *Machine Vision*, Chapter 2: Binary Image Processing, McGraw-Hill, New York.

Klodt, M., Herzog, K. , Töpfer, R., and Cremers, D.. (2015). Field phenotyping of grapevine growth using dense stereo reconstruction. *BMC Bioinformatics* 16(143).

Microsoft, (2016). Kinect Fusion: Kinect for Windows 1.7, 1.8. Microsoft.com. https://msdn.microsoft.com/en-us/library/dn188670.aspx

Montes, J., Melchinger, A., Reif, J.. (2007). Novel throughput phenotyping platforms in plant genetic studies. *Trends in Plant Science*, 12(10).

Nguyen, T., Slaughter, D., Max, N., Maloof, J., Sinha, N. (2015). Structured Light-Based 3D Reconstruction System for Plants. *Sensors*, 15, 18587-18612.

Pieruschka, R., Poorter, H.. (2012). Phenotyping plants: genes, phenes, and machines. *Functional Plant Biology,* 39, 813-820.

Royal Society of London. (2009). Reaping the Benefits: Science and the Sustainable Intensification of Global Agriculture. Royal Society, London.

Wengert, C. calib_toolbox_addon (Calibration Software) http://www2.vision.ee.ethz.ch/software/calibration_toolbox//calibration_toolbox.php

Yang, Jie, Lu, W., and Waibel, A. (1997). Skin-color modeling and adaptation'. CMU-CS-97-146.