A MOBILE AND CLOUD-BASED FRAMEWORK FOR PLANT STRESS DETECTION FROM CROWDSOURCED VISUAL AND INFRARED IMAGERY

by

DANIEL ZUKOWSKI

B.A., Yale University, 2004

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirement for the degree of

Master of Science

Department of Computer Science

2016

SIGNATURE PAGE

This thesis entitled:
A mobile and cloud-based framework for plant stress detection from crowdsourced visual and infrared imagery
written by Daniel Zukowski
has been approved for the Department of Computer Science

_____
Nikolaus Correll

_____
Dirk Grunwald

_____
Qin Lv

Date_____

The final copy of this thesis has been examined by the signatories, and we find that both the content and the form meet acceptable presentation standards of scholarly work in the above mentioned discipline.

ABSTRACT

Zukowski, Daniel (M.S., Computer Science)

A mobile and cloud-based framework for plant stress detection from crowdsourced visual and infrared imagery

Thesis directed by Assistant Professor Nikolaus Correll

A cloud infrastructure and Android-based system were developed to enable amateurs and professionals to make use of laboratory techniques for remote plant disease detection. The system allows users to upload and analyze plant data as citizen scientists, helping to improve models for remote disease detection in horticultural settings by greatly increasing the quantity and diversity of data available for analysis by the community. Techniques used in research laboratories for remote disease detection are generally not available to home gardeners and small commercial farmers. Lab equipment is cost-prohibitive and experiments highly controlled, leading to models that are not necessarily transferable to the user's environment. Plant producers rely on expert knowledge from training, experience, and extension service professionals to accurately and reliably diagnose and quantify plant health. Techniques for disease detection using visible and infrared imagery have been proven in research studies and can now be made available to individuals due to advancements in smartphones and low-cost thermal imaging devices. The framework presented in this paper provides an internet-accessible data pipeline for image acquisition, preprocessing, stereo rectification, disparity mapping, registration, feature extraction, and machine learning, designed to support research efforts and to make plant stress detection technology readily available to the public. A system of this kind has the potential to benefit

both researchers and plant growers: producers can collectively create large labeled data sets which researchers can use to build and improve detection models, returning value to growers in the form of generalizable models that work in real-world horticultural settings. We demonstrate the components of the framework and show data from a water stress experiment on basil plants performed using the mobile app and cloud-based services.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Nikolaus Correll for his inspiration, advice, and assistance in producing this master thesis. It was largely a result of discovering our shared interest in automated agriculture that I decided to apply to the Computer Science graduate program at the University of Colorado. Furthermore, I would like to thank my other committee members Dirk Grunwald and Christine Lv for their appreciated feedback and encouragement. I would like to thank instructor Joseph Tanner and my fellow team members of the X-Hab projects in the Aerospace Department with whom I shared great experiences in planning and executing engineering projects. Also, I would like to thank Heather Hava, my research partner on many exciting projects in automated agriculture over the past several years, for her support and her passion for the subject. Additionally, I would like to thank Michael Hurowitz of AcroOptics for loaning an amazing artificial lighting fixture to our experiment. I would also like to thank my coworkers at Archethought who graciously picked up many of my responsibilities while I took leave to produce this work. Finally, I would like to thank my parents for their support and love during this effort, and for all they've given me throughout my life.

CONTENTS

CHAPTER 1

INTRODUCTION

It is widely accepted that a substantial increase in global agricultural output will be required in coming decades to support a rising world population. To support an estimated 9 billion people and their livestock in 2050 will require a 70% increase in global food, animal feed and biofuel crop production (FAO 2009). Adding to this challenge is the reality that crop yields are significantly limited due to the effects of pathogens, insects, weeds, and environmental influences. Fungi, bacteria, and viruses account for 18% of overall crop loss (Oerke et al. 2004), while abiotic environmental stressors account for an additional 50% of losses (Bray et al. 2000). In situations where these factors are identified early in their emergence and properly mitigated, crop yield and profit losses for producers can be significantly reduced (Roberts et al. 2006). Equipping growers with the ability to diagnose the presence of plant stressors during early onset is important for informing targeted plant protection actions aimed at maintaining high quality and quantity in crop yields. Thus, early crop stress detection systems deployed worldwide have the potential to improve the stability of the food supply needed for a rapidly growing population.

Historically, crop health diagnostics have been performed by farmers relying on their own expertise and the assistance of professionals and laboratory services. In the field, diagrammatic scales are used to identify and estimate severity of disease. Extension service professionals often receive emailed images of sick plants from which they then use a variety of diagnostic guides and their own experience to determine the plant's ailment. In the laboratory, samples of plant tissue, water, and soil can be analyzed by skilled plant pathologists with specialized equipment. These traditional diagnostic methods have long

been applied in horticultural settings, but is it possible to approach the disease detection problem from a computer science perspective? In recent decades, experiments in autonomous diagnostics have been conducted with promising results using a variety of data sources and analysis techniques. Leveraging advancements in sensor hardware, computer vision, and machine learning, researchers have tested the utility of techniques from these fields by applying them towards the task of autonomous assessment of plant stressors using remotely acquired data. Studies have shown that such autonomous systems can indeed make significantly accurate diagnoses without physically interacting with the plant or its immediate environment. In some cases these methods can perform better than human diagnosticians by detecting diseased plants in visually pre-symptomatic states during pathogenesis (Lindenthal et al. 2005; Oerke et al. 2005). These techniques are made possible by the observation that biotic and abiotic stressors cause changes in plant color, structure, and temperature which alter the spectral properties of plant tissue, particularly in visible and infrared wavelengths, and can therefore be measured non-invasively by electromagnetic sensors including visible light cameras, infrared thermal imagers, and fluorescence and reflectance spectrometers (Mahlein 2016).

The general workflow of image-based remote diagnostic pipelines consists of four major steps: 1) imaging the plant in multiple wave bands, 2) pre-processing the imagery, 3) extracting features, and 4) using statistical methods and machine learning for diagnostic tasks related to what Singh et al. (2015) term ICQP tasks: identification, classification, quantification, and prediction of stress in plants. Experimental systems have been developed to perform one or more of the following tasks: identify the stressor affecting a plant, classify a plant into discrete classes of stress level (e.g. low, moderate, and high stress), estimate a stress/disease severity level (%), or predict the likelihood a plant will be experiencing stress in the future. Many studies aim to create discriminative models

through supervised machine learning by growing healthy and diseased plants in controlled environments from which labeled imagery is used to extract features. Section 1.3 mentions several studies that have investigated plant stress ICQP problems in this way.

For many years these techniques have been proven to work in controlled experimental settings with relatively small data sets, but we would like to test whether generalizable and practical models can be produced by crowdsourcing a much larger data set from real horticultural environments around the world. Large data sets aimed supporting plant stress ICQP research would allow, for example, further investigation into the use of powerful techniques such as deep learning (Schmidhuber 2015) which typically demand large quantities of training data. Deep learning methods have been applied successfully toward computer vision problems such as ImageNet object recognition by companies including Google (Szegedy et al. 2015) and Microsoft (Kaiming et al. 2015), or in the case of Baidu's performance leap in speech-to-text transformation from noisy environments (Amodei et al. 2015). Baidu's Andrew Ng describes the mathematical models and computing power underlying deep learning as powerful "rocket engines" that can be incredibly effective but require adequate "rocket fuel"—large heterogenous sets of training data—in order to reach their full potential (Ng 2015). In Baidu's case, record-breaking speech recognition performance was achieved on the data set by artificially increasing the size and variation of the data set by applying various transforms and background noise to the original training data—methods which may eventually be applied in the framework presented in this paper. It is in this spirit that we are producing a publicly accessible toolset for mass data collection and analysis—a system we hope may help generate a large and heterogenous data set, leading to plant stress detection models and tools that can benefit the wider plant-growing community. It is both cost- and time-prohibitive for individual research groups to build these large data sets that cover the range of subjects,

characteristics and conditions encountered in real horticultural settings. Similarly, it is cost-prohibitive for growers to employ the laboratory-grade techniques used by researchers to perform early remote detection due to the skills, equipment, and advanced computing required.

It is in pursuit of closing this gap that we have produced an open source mobile and cloud-based "big data" framework inspired by citizen science to serve as a locus for the advancement of remote disease detection research and application development. A mobile app allows users across the world to collect, label, and contribute data using their own smartphone with a low-cost thermal imaging attachment. Optical and thermal images are uploaded to the cloud and processed through a distributed data pipeline operating as a scalable microservices architecture. Processing modules deployed in Docker containers work together architecturally as a pipeline to perform preprocessing, feature extraction, and machine learning based on proven research methods for evaluating plant stress. As data accumulate, and models are developed and tested by researchers and other contributors, the community can benefit by utilizing these models in the field to perform plant stress ICQP tasks in real production settings. The modular, containerized architecture gives individual functional components of the framework the ability scale independently according to load, and allows the components of the system to be swapped with new ones as better disease assessment techniques emerge from the research community and become implemented into the framework.

1.1 Thesis Structure

Chapter 1 examines plant diagnostics, the spectral effects of stress on plants, related work in remote disease detection and citizen science, and new low-cost sensors that make it possible for non-researchers to participate in this project. Chapter 2 discusses the materials, methods and algorithms used by the framework in its data processing pipeline. Chapter 3 describes the mobile and cloud-based architecture of the framework for operation in a scalable production setting to support large-scale community involvement. Experimental results and conclusions are discussed in Chapter 4.

1.2 Conventional methods for plant disease diagnostics

Growers have traditionally relied on their own knowledge acquired from training and experience, or have consulted with diagnosticians and pathologists from private industry and public extension services to evaluate crop health. A common in situ method used by growers and diagnosticians consists of a visual examination aided by pictorial keys that guide the observer in assessing disease type and severity. This approach to visual assessment has long been investigated, and the convergence towards standardized approaches has led to increased accuracy and reliability of visual assessment methods (Bock et al. 2010). Figure 1 shows an example of a visual key designed by Sepúlveda-Chavera (2013) and used for assessment of powdery mildew in tomato caused by the bacterium *Solanum esculentum*. These keys provide standard scales that can be used by

trained professionals, but are subject to human error and are only useful for assessing disease severity after symptoms have become apparent to the human eye.

Figure 1: Diagrammatic scale for assessing the severity of powdery mildew in tomato (*Solanum esculentum*).

Whether symptoms are visible or not, diagnostics can be performed in the laboratory by testing plant tissues for the presence of pathogens. Bacterial infections can be diagnosed by placing pieces of infected plant tissue into nutrient media to incubate and grow the pathogen for identification (Schaad et al. 2001). Biotic pathogens and viruses can be identified using polymerase chain reaction (PCR) (Schaad et al. 2002) and enzyme-linked immunosorbent assay (ELISA) tests (Saettler et al. 1989; Hampton et al. 1990). Abiotic causes can be evaluated by analyzing nutrient content to check for deficiencies in micro- and macro-nutrients (Hajiboland 2012) which can produce symptoms similar to those caused by biotic agents (Flynn 2003). Tests of the surrounding environment (soil, water,

etc.) can be performed to evaluate the contribution of additional abiotic influences such as pH, salinity, pesticides and other environmental pollutants.

Laboratory testing is widely used to directly measure potential causal agents. However, these diagnostic methods demand specialized skills and equipment, require intrusive action to obtain a physical specimen, are expensive to perform, and must be repeated frequently. Days or weeks can pass before results are returned to the grower, during which the stressor will have had time to increase its potential for crop damage. To minimize losses and limit mediation expenditures, stress must be detected as early as possible so that the grower can take precise corrective action. Therefore, methods that are neither expensive nor invasive, can provide results quickly, and can be repeated often by non-experts are needed to improve the ability of growers to perform early detection of plant stress. Remote detection methods in particular have the potential to fill this gap, and are further explored in the following sections.

1.3 Effects of stress on spectral signature of plants

A plant's leaf surface, internal cellular structures, pigments, and water content influence its appearance in the visible and infrared spectra (Knipling 1970). Stress from biotic and abiotic factors can cause changes in these characteristics, altering the spectral properties of the plant, and thus providing the rationale for remote sensing applications in this area. For example, in visible wavelengths, leaves suffering from chlorosis will turn paler in color, becoming more reflective, whereas necrotic lesions may be detectable as dark spots on leaves. Water content influences the plant's emissivity in the thermal infrared band, causing plants affected by changes in transpiration rate or tissue health, for instance,

to exhibit features that can be detected with thermal imaging devices. Oerke et al. (2010) showed that scab in apple produces symptoms that become apparent and detectable in the thermal infrared 1-3 days before they are detectable in visible light due to a localized temperature decrease in the area where lesions later appear. Similarly, Allegre et al. (2006) showed that affected regions of leaves in Plasmopara viticola-infected grapevine exhibit decreased temperatures compared to non-affected regions. Several studies have evaluated the effectiveness of thermal infrared imagery in assessing water stress in crop canopies (Blum et al. 1982; Jones et al. 2009; Raza et al. 2014).

1.4 Related Work

1.4.1 Remote Disease Detection

Disease assessment using optical imagery has been performed on tobacco and apple (Wijekoon et al. 2008), grapefruit (Bock et al. 2008), and cotton (Camargo and Smith 2009). Neumann et al. (2014) used RGB data for detection of disease in sugar beet, and their work is particularly relevant to our framework for its use of smartphone cameras and mobile apps to address the classification problem in the field.

Infrared thermal imaging has been used by Oerke et al. (2011) to detect scab in apple, and by Gomez (2014) to detect downy mildew in rose. Presymptomatic detection of downy mildew in cucumber has been demonstrated by Lindenthal et al. (2005) and Oerke et al. (2005) using infrared thermography. Belasque et al. (2008) used fluorescence spectroscopy to detect citrus canker, and a combination of thermal and chlorophyll-

fluorescence imaging was used by Chaerle et al. (2004) to detect early-stage Cerscospora leaf spot in sugar beet.

A recurring theme found throughout these studies is the effectiveness of sensor fusion—combining data from multiple data sources—in creating stronger predictive models. For example, Raza et al. (2014) was able to detect water stress in spinach canopies with an accuracy of 97% by using both visible and thermal imagery and a classification model using a combination of Support Vector Machine (SVM) and Gaussian Process for Classification (GPC). Using color or thermal only, accuracy was lower at 68% and 92%, respectively. Raza et al. (2015) also produced a model to detect the presymptomatic presence of powdery mildew in tomato by fusing color, thermal infrared, and depth information. Adding the third dimension of depth via stereo imaging was shown to consistently improve the model's accuracy when compared to analyses performed using color only, thermal only, and color + thermal. This result has led us to include depth data extraction in our framework by implementing rectification and disparity modules for processing unconstrained stereo imagery from smartphone cameras.

Since diseases and other stressors vary in terms of how they affect the spectral signature of a plant, a robust and generalizable remote detection system will likely use a combination of sensors and processing methods in order to learn useful parameters for modeling diseases in various crop types. For this reason, we designed our framework as a configurable microservices pipeline so that techniques from past and future studies can continue to be applied to the framework in the form of swappable software modules, improving the system as new discoveries are made. Section 3.3 discusses this architecture in more detail.

1.4.2 Software applications

Disease assessment using optical imagery has been performed on tobacco and apple (Wijekoon et al. 2008), grapefruit (Bock et al. 2008), and cotton (Camargo and Smith 2009). Neumann et al. (2014) used RGB data for detection of disease in sugar beet, and their work is particularly relevant to our framework for its use of smartphone cameras and mobile apps to address the classification problem in the field. Infrared thermal imaging has been used by Oerke et al. (2011) to detect scab in apple, and by Gomez (2014) to detect downy mildew in rose. Pre-symptomatic detection of downy mildew in cucumber has been demonstrated by Lindenthal et al. (2005) and Oerke et al. (2005) using infrared thermography. Belasque et al. (2008) used fluorescence spectroscopy to detect citrus canker, and a combination of thermal and chlorophyll-fluorescence imaging was used by Chaerle et al. (2004) to detect early-stage Cerscospora leaf spot in sugar beet.

A recurring theme found throughout these studies is the effectiveness of sensor fusion—combining data from multiple data sources—in creating stronger predictive models. For example, Raza et al. (2014) was able to detect water stress in spinach canopies with an accuracy of 97% by using both visible and thermal imagery and a classification model using a combination of Support Vector Machine (SVM) and Gaussian Process for Classification (GPC). Using color or thermal only, accuracy was lower at 68% and 92%, respectively. Raza et al. (2015) also produced a model to detect the pre-symptomatic presence of powdery mildew in tomato by fusing color, thermal infrared, and depth information. Adding the third dimension of depth via stereo imaging was shown to consistently improve the model's accuracy when compared to analyses performed using color only, thermal only, and color + thermal. This result has led us to include depth data extraction in our framework by

implementing rectification and disparity modules for processing unconstrained stereo imagery from smartphone cameras.

Since diseases and other stressors vary in terms of how they affect the spectral signature of a plant, a robust and generalizable remote detection system will likely use a combination of sensors and processing methods in order to learn useful parameters for modeling diseases in various crop types. For this reason, we designed our framework as a configurable microservices pipeline so that techniques from past and future studies can continue to be applied to the framework in the form of swappable software modules, improving the system as new discoveries are made. Section 3.3 discusses this architecture in more detail.

1.4.3 Citizen science

The driving motivation for developing and providing this framework is the goal of enabling a citizen science collaboration with the global community of plant growers in order to accelerate and improve research and applications in early plant stress detection technology. Whitelaw et al. (2003) defined citizen science as "a process where concerned citizens, government agencies, industry, academia, community groups, and local institutions collaborate to monitor, track and respond to issues of common community [environmental] concern." While not generally covered in the scientific peer-reviewed literature, several citizen science projects involving community-based monitoring (CBM) are covered in Conrad & Hilchey's (2009) review. Notable citizen science projects have addressed topics such as radiation and air quality measurements (Safecast) [60], taxonomical identification and tracking of wildlife populations (BugGuide and iSpot) [11,

32], climate change through observations of bud burst (BudBurst) [55], and galaxy classification (GalaxyZoo) [25].

In recent years, citizen science projects have emerged in the plant science domain specifically. The Photosynq project [50] offers a handheld spectrometer and mobile app for collecting plant, soil, and water quality data and has led to hundreds of projects by its community of more than 1,000 users. Goeau et al (2011) developed a framework for taxonomical identification of plant species from smartphone camera photographs. Their system includes a mobile app that allows the public to take pictures of plant leaves and flowers, perform identification estimation against stored models, and add to the collective database for improving identification models. Several apps designed to assist with taxonomical identification and plant symptom evaluation can be found in Google Play and the Apple App Store [1, 52, 67].

1.5 Low-cost consumer spectral devices

In 2015, FLIR Systems, Inc. released the second generation of their FLIR One thermal imager for Android. As mentioned earlier, thermal imagery has been shown to be an effective data source in remote plant stress diagnostics, and thus is likely to be an important component of robust disease detection systems. We chose the FLIR One sensor to be used with our framework due to its wide availability and relatively low cost (the sensor retails for approximately $250). This makes it affordable to individuals who would be participants on the citizen science side of the deployed framework, and to researchers who may not have budgets for laboratory-grade thermal imaging equipment. According to FLIR, the thermal resolution of the device is 0.1 C. Compared to the Cedip Titanium, used by

Raza (2015), for example, the FLIR One is no doubt a consumer-grade tool—it has lower resolution and sensitivity, and lacks a self-cooling system. One goal of our mobile app and cloud framework is to evaluate the utility of this device for plant stress detection tasks. We also looked at the $250 Seek Thermal smartphone-compatible thermal imaging device, however decided not to use it because it currently requires the user to perform manual focusing. Support for additional devices such as Seek can be added in the future by motivated contributors, as the mobile application code is open source and therefore open to improvements and extensions by the community.

CHAPTER 2

MATERIALS & METHODS

2.1 Hardware & Software Overview

An Android application for acquiring and uploading plant imagery and metadata was developed and tested on a Galaxy Note 5. The app was designed to integrate with the FLIR One thermal imaging camera (Figure 2) using the Software Development Kit (SDK) for Android provided by FLIR Systems, Inc. [22]. Imagery and metadata are uploaded by the mobile app to a cluster of Ubuntu 14.04 cloud servers on Amazon's Elastic Compute Cloud (EC2). Imagery is stored in Amazon's Simple Storage Solution (S3) while metadata and extracted features are stored in Postgres [53] and Elasticsearch [17] databases. Optical and thermal images are received by an HTTPS API endpoint developed with Ruby on Rails [31]. Via a RabbitMQ [56] message bus, data are passed along to multiple microservice processing modules in a pipeline configuration deployed in Docker [16] containers. The microservice modules perform calibration, preprocessing, registration, rectification, feature extraction, and machine learning tasks designed specifically for plant stress detection. Functions provided by open-source libraries including OpenCV [9], numpy [45] and scikit-learn [63] were used in developing several of the processing and learning modules. At the time of this writing, a web site is being developed to provide researchers with tools to configure, use, and extend the pipeline for plant stress ICQP tasks. An initial pipeline was designed and programmed with modules that implement various techniques published in the literature, however this debut pipeline represents only one of any number of pipelines that can be developed and configured according to the needs of the experimenter. The

initial pipeline was tested and verified in the basil water stress experiment described in Section 4.1.

Figure 2: FLIR One thermal imager for Android, approx. $250.

2.2 Pipeline

A block diagram of the cloud framework is shown in Figure 3. A complete list of microservice modules, their purpose, and algorithms are described in Table 1.

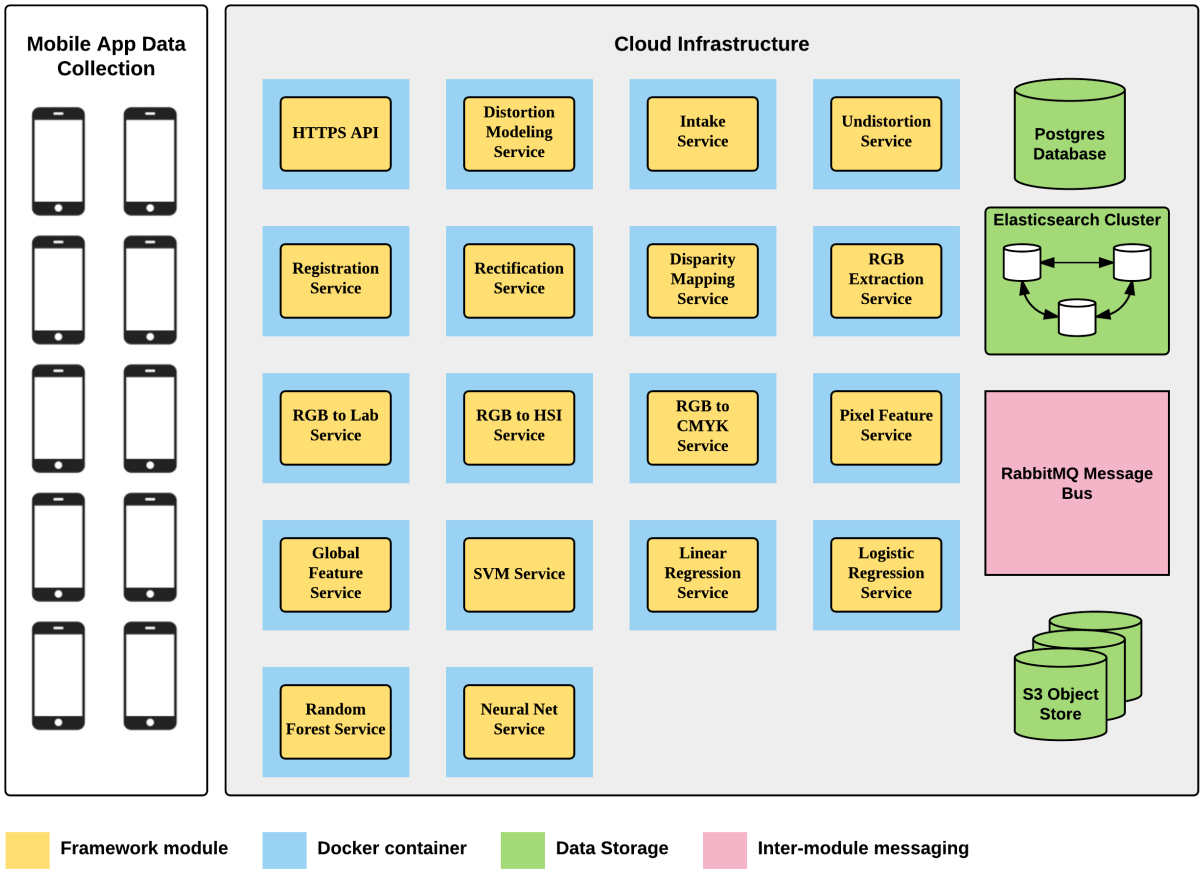

Figure 3: System architecture block diagram.

Module	Purpose	Subsystem	Algorithms
CalibrationCollection	Calibration (image acquisition)	Android App	-
DistortionModelingService	Calibration (lens modeling)	Cloud	Checkerboard calibration
ImageAcquisiiton	Acquisition (plant data)	Android App	-
IntakeService	Store uploaded images in Amazon S3	Cloud	-
UndistortionService	Undistortion	Cloud	Checkerboard calibration
RegistrationService	Rectification	Cloud	Oriented FAST Rotated BRIED (ORB), 8-point algorithm
DisparityMappingService	Depth representation from stereo imagery	Cloud	Block Matching Stereo (BMS)
RGBExtractionService	RGB extraction	Cloud	-
RGB2LabService	RGB to Lab transform	Cloud	-
RGB2HSIService	RGB to HSI transform	Cloud	-
RGB2CMYKService	RGB to CMYK transform	Cloud	-
PixelFeatureService	Pixel-based feature engineering	Cloud	-
GlobalFeatureService	Global feature engineering	Cloud	-
SVMService	SVM model train/test	Cloud	Support Vector Machine (SVM)
LinRegService	Linear Regression train/test	Cloud	Linear Regression
LogRegService	Logistic Regression train/test	Cloud	Logistic Regression
RandomForestService	RandomForest train/test	Cloud	Random Forest
NeuralNetService	Neural Network train/test	Cloud	N-Layer Neural Network with Backpropagation and Dropout

Table 1: List of framework modules, subsystem in which they are implemented, and algorithms used.

2.3 Data Acquisition

Plant stress detection experiments using stereo imagery have often employed two or more cameras secured in place with mounting hardware to help maintain mechanical alignment between imaging elements (Biskup et al. 2007; Sabine et al. 2011; Raza 2015) . When cameras are aligned and calibrated to be coplanar, the stereo rectification process is simplified by having prior knowledge of the cameras' spatial relationship. This type of setup requires at least two cameras and specialized knowledge for calibrating such camera configurations. However, because our framework is intended to be used with a single camera-equipped smartphone in conditions where a mounting device may not be available, the system must facilitate handheld stereo imaging in an environment of unknown geometry. Therefore, the framework must perform stereo image rectification in the uncalibrated case, which has been solved a number of ways as shown by Papadimitriou and Dennis (1996), Fusiello and Irsara (2008), and Kumar et al. (2010). Also, because of differences in the optical properties of camera lenses across smartphones, modeling lens distortion for each user's device is a necessary prerequisite prior to acquiring imagery for the purpose of undistorting submitted images prior to rectification. The following sections describe the initial modules, processes, and algorithms we have implemented in the framework for performing calibration, acquisition, undistortion, rectification, registration, feature extraction, and modeling from user-generated optical and thermal imagery.

2.3.1 One-time calibration

The intrinsic properties of most camera lenses cause the appearance of straight lines to bend to some degree either outward from the center of the image, known as barrel distortion, or inward towards the center of the image resulting in pincushion distortion (Figure 5). Effective stereo image rectification requires undistorted input images which can be created by applying a transformation to raw images using information from the lens model. For our purposes, it would be difficult to provide pre-generated models for every smartphone camera lens that may be used by the community for undistorting user-submitted imagery. Thus, we recommend that users perform a calibration step to give the processing pipeline necessary information to correct for their specific camera's lens distortion prior to performing rectification and extracting 3D information from the stereo pair.

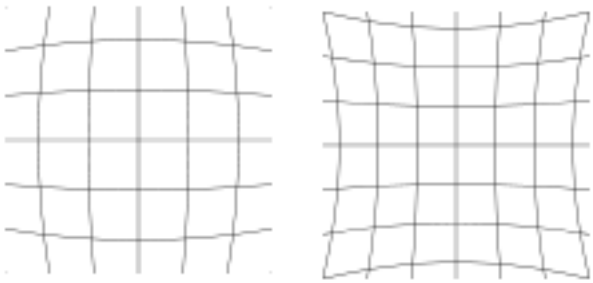

Figure 5: Barrel distortion (left) and pincushion distortion (right).

The lens calibration module implements the checkerboard technique as described by Zhang (2000). A checkerboard pattern is printed out and images are taken from various perspectives. Using a set of 10 images (Figure 6) , the calibration algorithm detects the

checkerboard and measures the distortion in the pattern, estimating camera and distortion matrices which are then applied in preprocessing for undistorting images.

2.3.2 Image acquisition

When initiating data collection, the app first instructs the user to find an area of interest (AOI) and center it in the field of view at a distance that allows the subject to appear prominently in the image. Following on-screen instructions, the user positions the device slightly to the left of the AOI, rotates the camera to center the view on the AOI, and captures the left image, repeating this process for the right image. Thermal images are also captured at this time. The left, right, and thermal images are securely uploaded to the cloud via an HTTPS API, providing the raw data for processing through the detection pipeline.

2.3.3 Metadata

Prior to image upload, the user is asked to provide additional metadata about the subject and its environment. The only required inputs are the type of crop (e.g. tomato, pea, cucumber), location (i.e. indoors or outdoors), and lighting conditions. Optional information includes ambient air temperature estimate and approximate distance from camera to the nearest area of the plant canopy. If known, the user can also report the disease, deficiency, or other condition affecting the plant, along with an estimate of confidence (low, medium, high). For devices that contain ambient an air temperature sensor (e.g. Samsung S4), the

current ambient temperature is automatically reported. The current time is automatically reported. Geolocation data is reported if authorized by the user.

2.4 Processing

2.4.1 Lens distortion correction

Using the lens distortion information captured during one-time calibration, raw optical images are preprocessed to correct for distortion prior to performing stereo rectification. Figure 7 shows an image before and after processing by the undistortion module. Currently, the framework does not supply a module for undistortion of thermal images. The sharp edges in the checkerboard pattern are readily detectable in the visible spectrum, but this method cannot be used to correct for distortion from thermal camera lenses. Heat at object edges can be conducted by neighboring objects, making detection of edge features in thermal imagery difficult and imprecise. Yahyanejad (2011) proposes a method for correcting lens distortion in thermal imagery that could be applied to the framework in the future.

2.4.2 Visual-thermal registration

The infrared and optical imaging elements of the FLIR One are linearly separated, causing captured frames to be unaligned (Figure 8). Therefore, thermal and optical images must be registered before generating pixel feature vectors such that color and thermal

information are located at the same coordinate on the image plane. We use a combination of preprocessing, CANNY edge detection (Canny 1986), morphological transforms, masking, and direct pixel-intensity-based registration to align the two images.

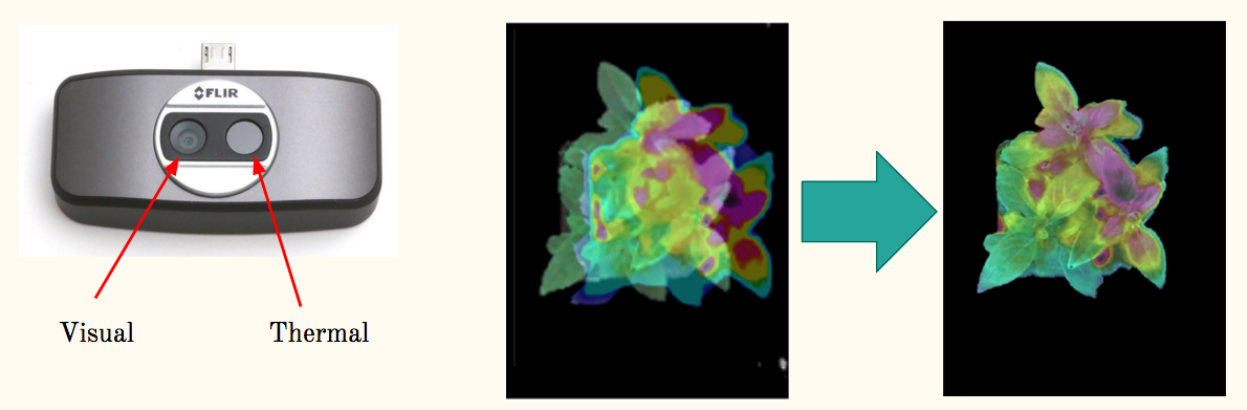

Figure 8: The offset of thermal and visual imaging elements in the FLIR One (left) produce unaligned optical and thermal images (middle) which need to be registered to align pixel features (right).

2.4.2.1 Foreground isolation

First, the optical image (Figure 9a) is grayscaled and enhanced by an increase in brightness and contrast (Figure 9b). For images where pixel intensity values in the AOI are generally darker than the background, the elevated brightness washes out the background while preserving the plant outline. We found this to reduce the influence on the CANNY detector by diffuse shadow regions around the AOI. For images where subject pixel areas are generally of higher intensity compared to the background, the grayscale image is first inverted before being processed by the edge detector. To determine whether to invert the grayscale, statistics are obtained from the AOI (a square in the center of the image) and

compared with the area outside of the AOI. The mean value of pixel intensities from inside and outside of the AOI are used to determine whether grayscale image inversion is a necessary preprocessing step. The CANNY edge detector produces a binarized 2D contour image (Figure 9c) which is then repeatedly dilated using the maximum value of a 3x3 neighborhood around each pixel. Recursive dilation helps to produce enclosed regions (Figure 9d) using the result from the initial CANNY detector. Enclosed regions are then filled using a mathematical morphological operation (Serra 1982) to produce a rough mask (Figure 9e). The mask is then repeatedly eroded (Figure 9f), and enclosed regions below a configurable size threshold are eliminated, removing artifacts that occur outside the area covered by the subject in the AOI (Figure 9g). The resulting mask is then applied to the original optical image to isolate it from the background. After this step, some amount of unmasked background area remains (Figure 9h). Thresholding is used to isolate the remaining background (Figure 9i), producing a second mask. The second mask is subtracted from the first and applied to the original optical image, resulting in an isolated foreground with black background (Figure 9j).

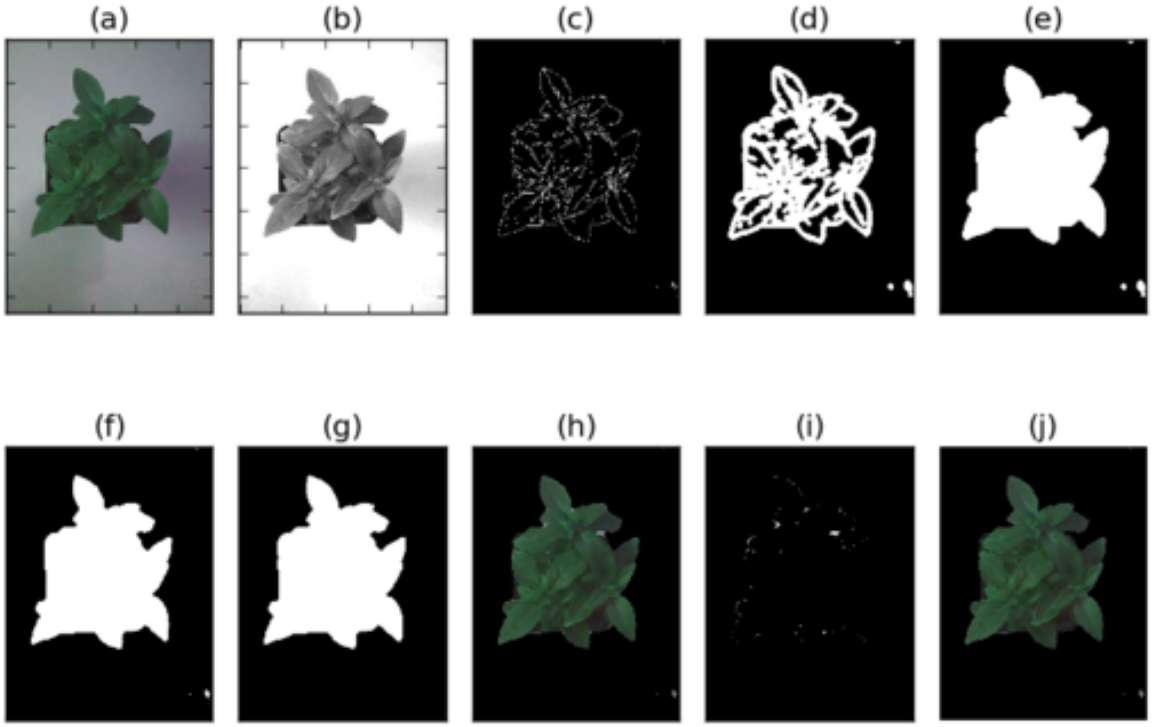

Figure 9: Optical image after each step in foreground isolation.

For thermal image foreground isolation, a high-contrast false color rendition of the thermal image (Figure 10a) is grayscaled (Figure 10b), processed by the CANNY edge detector (Figure 10c), and processed through similar dilation (Figure 10d), filling (Figure 10e), erosion (Figure 10f), and artifact removal (Figure 10g) steps to produce a mask which is then applied to the original thermal image for foreground isolation (Figure 10h). The processed optical and thermal images and their masks are stored and available for further processing, for example, by the registration module.

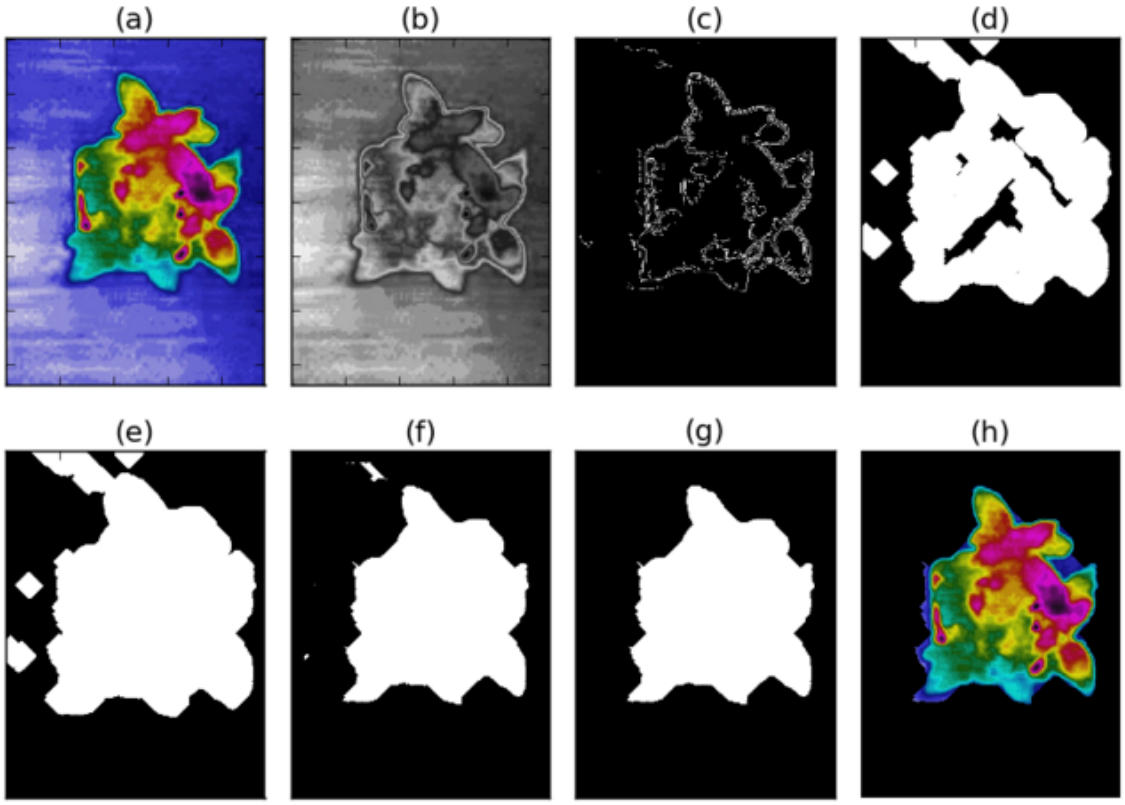

Figure 10: False-color thermal image after each step in the foreground isolation process.

While this process of foreground isolation works well in the presence of a relatively uniform background, and when there is a sufficient difference in mean pixel intensities between the AOI and the background, it does not perform well in the case of noisy or textured backgrounds, or when there is little difference in intensity between foreground and background. More robust methods of edge detection for foreground isolation should therefore be explored and incorporated into the framework. Applications of Stationary Wavelet Transform, for example, have been shown to match or outperform CANNY edge detection in noisy environments (El Menzeni et al 2014) and such approaches are targeted for incorporation into the framework in the future.

2.4.2.2 Registration

As the authors of Raza et al. (2015) point out, performing optical-thermal image registration is difficult using conventional methods due to the divergent representation of plant textures in optical and thermal spectra. The lack of surface feature fidelity in the thermal image precludes feature-based registration using algorithms like SIFT, SURF or ORB, so we instead use a direct pixel-based registration method using the optical and thermal masks derived during foreground isolation. Ideal displacement is found by iteratively shifting the thermal mask to minimize the sum of squared differences (SSD) between it and the optical mask. The displacement found through SSD minimization is then applied to the thermal image to achieve a best-fit alignment with the optical image. Figure 11 shows a combined thermal and optical image after registration and masking. The registered optical and thermal pixel vectors are stored for creating pixel-based features used in model training and testing.

Figure 11: Combined visual and false-color thermal image after masking and registration.

2.4.3 Stereo rectification

Rectification is the process of projecting two images taken from different perspectives onto a common plane. In our case, images are captured by hand and are thus assumed to be non-coplanar. Efficiently generating good disparity maps, and in turn extracting depth information, generally requires stereo pairs to be well-rectified. To achieve rectification, two images taken from unknown points in space are transformed, or warped, such that pairs of conjugate epipolar lines are made collinear and parallel to the x-axis. In this projection, a pixel feature in the left image corresponds to a pixel feature in the right image with the same y coordinate, but shifted along the x-axis. The magnitude of this shift gives a disparity measure which is inversely proportional to depth. The relative depth can

be calculated for each pixel via triangulation using the disparity between corresponding points. Including depth information in remote disease detection models has been shown to improve accuracy when compared with models that do not include depth information (Raza et al. 2015). Light, humidity, and thermal gradients occur throughout the layers of the plant canopy due to occlusion, air flow mechanics, and transpiration variation in leaves at different levels. Additionally, symptoms can manifest differently at various levels of the canopy. For example, calcium is generally not transportable via phloem in plant tissues and thus not available for redistribution, causing calcium deficiencies to become apparent at upper levels of the plant canopy where new growth develops. Conversely, nitrogen is more easily transported, and deficiencies may first become apparent at lower levels of the canopy as new growth borrows available nitrogen from older leaves. Natural senescence in older leaves at lower levels of the canopy can cause plants to exhibit morphological features similar to those induced by stress factors. Thus, depth information may help models distinguish features occurring at different levels of the canopy.

The stereo image rectification module in the framework implements the following procedures:

1. Identify ORB features in left and right optical images
2. Find pairs of corresponding ORB features
3. Estimate the fundamental matrix from feature pairs using the 8-point algorithm
4. Transform images onto a common plane to achieve epipolar alignment

Each step is further described below.

2.4.3.1 ORB feature detection

In the absence of a known geometry describing the coordinates and pose of the camera in each image of the stereo pair, features common to both images can be used to identify the relative spatial attributes of the camera at the time each of the left and right images were obtained. To find these points, local features are identified and described in each image using the Oriented FAST and Rotated BRIEF (ORB) algorithm (Rublee et al. in 2011). The choice of ORB over Scale-Invariant Feature Transform (SIFT) and Speeded-Up Robust Features (SURF) was made primarily to avoid patent-based restrictions on the use of those algorithms by applications which might be built on top of our open-source framework.

2.4.3.2 Feature correspondences

Corresponding features are identified by finding ORB feature pairs from left and right images with the least Hamming distance among nearest neighbors, and outliers are filtered by the RANSAC algorithm (Fischler and Bolles 1981). Figure 12 shows the top 15 feature pairs extracted from an example stereo set by the rectification module.

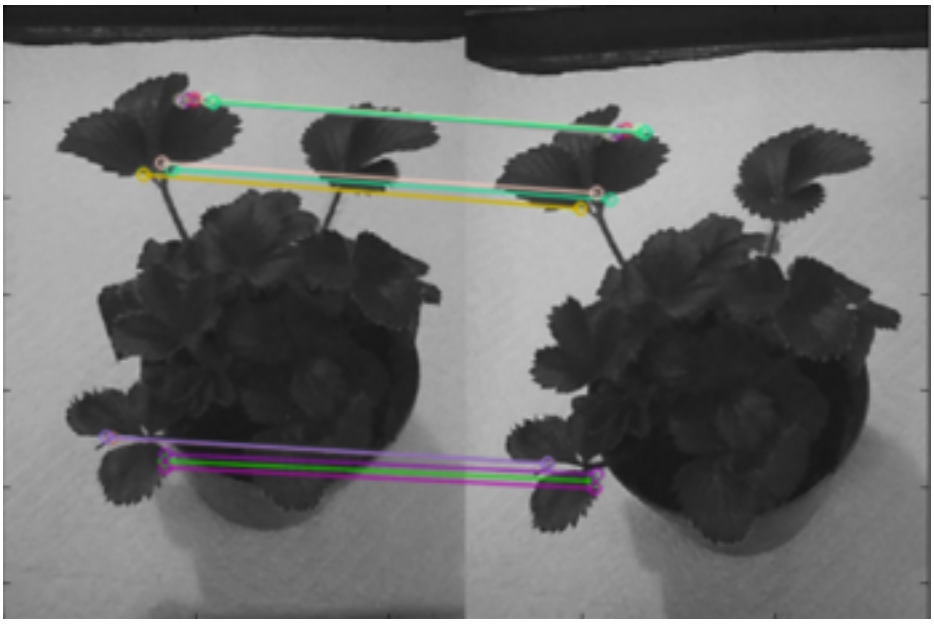

Figure 12: Top 15 correspondence pairs from ORB.

2.4.3.3 Rectification

After extracting ORB features and matching correspondence pairs from the optical stereo imagery, rectification is performed using the normalized 8-point algorithm (Hartley 1997). Left and right images are related by finding the fundamental matrix $\boldsymbol{F}$ which encapsulates intrinsic and extrinsic parameters of the camera. For a point $\boldsymbol{x}$ in the left image, $\boldsymbol{F}$ determines an epipolar line $\boldsymbol{Fx}$ on which the corresponding point $\boldsymbol{x'}$ lies in the right image. At least eight corresponding points are required to estimate the fundamental matrix, though generally more than 8 pairs are used to generate an over-determined result. In our framework, at least 10 pairs are used.

The fundamental matrix ***F*** is used in a projective transformation of the left and right images onto a common plane. The result of this projection is a new representation satisfying the constraints required for extracting depth information from the stereo pair: that all epipolar lines are parallel to the x axis, and that corresponding points have the same y-coordinate in each image. Figure 13 shows the parallel-eyed pair of stereo images before and after processing by the rectification module.

Figure 13: Stereo pair before (top) and after (bottom) rectification.

2.4.4 Disparity mapping

Given a pair of rectified stereo images satisfying the epipolar constraint, the distance between corresponding pixels (disparity) along horizontal lines can be calculated. Our framework implements a standard Block Matching (BM) algorithm with New Three Step Search (NTSS) (Li 1994) in which a configurable NxN pixel search window is used to scan across horizontal regions in left and right rectified images to find pixel correspondence areas. Matching areas are found by minimizing the sum of absolute differences (SAD) in RGB values between blocks. Block matching is computationally efficient and straightforward to implement, but superior methods exist and are being pursued for future additions to the framework. Specifically, the Guided Dynamic Programming from Block Matching Stereo (BM+DP) algorithm described by Nguyen (2014), which uses a combination of BM and Symmetric Dynamic Programming Stereo (SDPS) (Gimel'farb 2002), has been shown by the algorithm's authors to produce disparity maps with 28% lower error compared to SDPS alone, and is a more robust alternative to our initial implementation of BM.

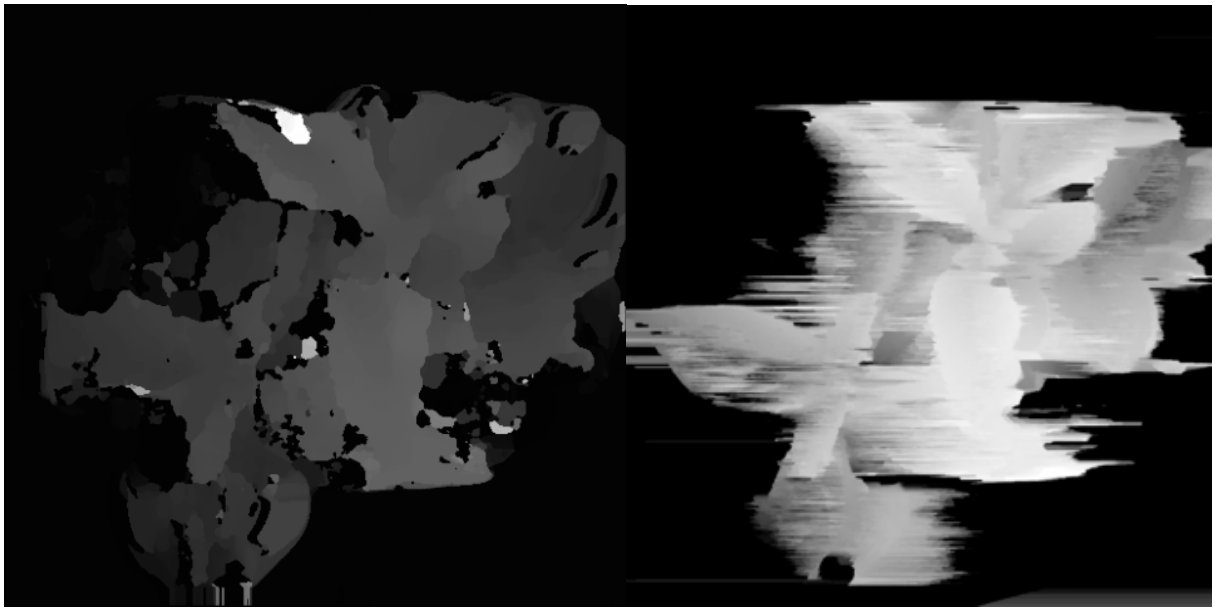

Figure 14: Grayscale depth images from disparity maps created using Block Matching (left) with the framework's Disparity Mapping module and BM+DP (right) produced with the Online Computational Stereo Vision system by Minh Nguyen at the University of Auckland.

2.4.5 Color space transform

Each optical image can be passed through transform modules to extract RGB, CMYK, HSI, and Lab color space attributes at each pixel. The transformed color channel values are normalized from 0.0-1.0 and stored as 1D arrays for later use as pixel-based features for model training or in further processing by the Global Feature Extraction module, for instance, to extract statistics from select color channels. The color transforms are straightforward arithmetic conversions, with Lab being the only color space with a configurable variable to set the white point. Our framework uses the International

Commission on Illumination (CIE) standard illuminant D50 to set the white point for Lab conversion.

2.4.6 Feature Extraction

For each pixel of input data supplied to the framework, 67 characteristics are extracted from data processing modules (Table 2) and are available for generating pixel-based or global feature vectors for training and classification. Each feature is stored in the Elasticsearch database as a JSON representation of a scalar or vector of floating point values normalized to a range of 0.0-1.0. The framework allows the researcher to define the subset of characteristics used to construct the feature vectors according to a user-defined configuration. A graphical interface in the web application generates a YAML file for setting the feature vector's characteristics, the processing pipeline steps and their parameters, and the machine learning algorithms and their parameters. For example, using the SVM module, the user may define the kernel, kernel parameters, and soft-margin parameter.

Characteristic	Key	Module
RGB: red channel (R)	rgb_r	RGBExtractionService
RGB: green channel (G)	rgb_g	RGBExtractionService
RGB: blue channel (B)	rgb_b	RGBExtractionService
CMYK: cyan channel (C)	cmyk_c	RGB2CMYKService
CMYK: magenta channel (M)	cmyk_m	RGB2CMYKService
CMYK: yellow channel (Y)	cmyk_y	RGB2CMYKService
CMYK: black channel (K)	cmyk_k	RGB2CMYKService
Lab: lightness (L)	lab_l	RGB2LabService
Lab: red/green (a)	lab_a	RGB2LabService
Lab: yellow/blue (b)	lab_b	RGB2LabService
HSI: hue (H)	hsi_h	RGB2HSIService
HSI: saturation (S)	hsi_s	RGB2HSIService
HSI: intensity (I)	hsi_i	RGB2HSIService
Temperature	thermal	ThermalExtractionService
Depth	disparity	DisparityMappingService
RGB: red mean	rgb_r_mean	GlobalFeatureExtraction
RGB: red mean (masked)	rgb_r_masked_mean	GlobalFeatureExtraction
RGB: red variance	rgb_r_var	GlobalFeatureExtraction
RGB: red variance (masked	rgb_r_masked_var	GlobalFeatureExtraction
RGB: green mean	rgb_g_mean	GlobalFeatureExtraction
RGB: green mean (masked)	rgb_g_masked_mean	GlobalFeatureExtraction
RGB: green variance	rgb_g_var	GlobalFeatureExtraction
RGB: green variance (masked	rgb_g_masked_var	GlobalFeatureExtraction
RGB: blue mean	rgb_b_mean	GlobalFeatureExtraction
RGB: blue mean (masked)	rgb_b_masked_mean	GlobalFeatureExtraction
RGB: blue variance	rgb_b_var	GlobalFeatureExtraction
RGB: blue variance (masked)	rgb_b_masked_var	GlobalFeatureExtraction
CMYK: cyan mean	cmyk_c_mean	GlobalFeatureExtraction

Characteristic	Key	Module
CMYK: cyan mean (masked)	cmyk_c_masked_mean	GlobalFeatureExtraction
CMYK: cyan variance	cmyk_c_var	GlobalFeatureExtraction
CMYK: cyan variance (masked)	cmyk_c_masked_var	GlobalFeatureExtraction
CMYK: magenta mean	cmyk_m_mean	GlobalFeatureExtraction
CMYK: magenta mean (masked)	cmyk_m_masked_mean	GlobalFeatureExtraction
CMYK: magenta varianc	cmyk_m_var	GlobalFeatureExtraction
CMYK: magenta variance (masked)	cmyk_m_masked_var	GlobalFeatureExtraction
CMYK: yellow mean	cmyk_y_mean	GlobalFeatureExtraction
CMYK: yellow mean (masked)	cmyk_y_masked_mean	GlobalFeatureExtraction
CMYK: yellow variance	cmyk_y_var	GlobalFeatureExtraction
CMYK: yellow variance (masked)	cmyk_y_masked_var	GlobalFeatureExtraction
CMYK: black mean	cmyk_k_mean	GlobalFeatureExtraction
CMYK: black mean (masked)	cmyk_k_masked_mean	GlobalFeatureExtraction
CMYK: black variance	cmyk_k_var	GlobalFeatureExtraction
CMYK: black variance (masked)	cmyk_k_masked_var	GlobalFeatureExtraction
Lab: lightness mean	lab_l_mean	GlobalFeatureExtraction
Lab: lightness mean (masked)	lab_l_masked_mean	GlobalFeatureExtraction
Lab: lightness variance	lab_l_var	GlobalFeatureExtraction
Lab: lightness variance (masked)	lab_l_masked_var	GlobalFeatureExtraction
Lab: red/green mean	lab_a_mean	GlobalFeatureExtraction
Lab: red/green mean (masked)	lab_a_masked_mean	GlobalFeatureExtraction
Lab: red/green variance	lab_a_var	GlobalFeatureExtraction
Lab: red/green variance (masked)	lab_a_masked_var	GlobalFeatureExtraction
Lab: yellow/blue mean	lab_b_mean	GlobalFeatureExtraction
Lab: yellow/blue mean (masked	lab_b_masked_mean	GlobalFeatureExtraction

Characteristic	Key	Module
Lab: yellow/blue variance	lab_b_var	GlobalFeatureExtraction
Lab: yellow/blue variance (masked)	lab_b_masked_var	GlobalFeatureExtraction
HSI: hue mean	hsi_h_mean	GlobalFeatureExtraction
HSI: hue mean (masked)	hsi_h_masked_mean	GlobalFeatureExtraction
HSI: hue variance	hsi_h_var	GlobalFeatureExtraction
HSI: hue variance (masked)	hsi_h_masked_var	GlobalFeatureExtraction
HSI: saturation mean	hsi_s_mean	GlobalFeatureExtraction
HSI: saturation mean (masked)	hsi_s_masked_mean	GlobalFeatureExtraction
HSI: saturation variance	hsi_s_var	GlobalFeatureExtraction
HSI: saturation variance (masked)	hsi_s_masked_var	GlobalFeatureExtraction
HSI: intensity mean	hsi_i_mean	GlobalFeatureExtraction
HSI: intensity mean (masked)	hsi_i_masked_mean	GlobalFeatureExtraction
HSI: intensity variance	hsi_i_var	GlobalFeatureExtraction
HSI: intensity variance (masked)	hsi_i_masked_var	GlobalFeatureExtraction

Table 2: List of 67 features extracted by framework modules and available for creating feature vectors.

2.5 Classification

Five machine learning modules are initially provided by the framework for performing binary classification tasks (e.g. healthy vs stressed): Support Vector Machine (SVM), Linear Regression, Logistic Regression, Random Forest, and Neural Network. Each learning module can take an input vector of arbitrary length by configuring the module to select any combination of the 67 extracted features. The neural network training module

provides a configurable N-layer artificial neural network with stochastic gradient descent (SGD) optimized backpropagation, and supports dropout. The implementation was verified using the MNIST dataset for handwritten digit recognition, but has yet to be tested on plant classification tasks. As more data is contributed to the system, the neural network module will become more practical as these types of networks, when used on pixel matrices, generally require large amounts of training data to achieve usable generalization error rates.

2.5.1 Binary vs. multiclass classification

Discriminative binary classification with Support Vector Machines, for example, works by finding an optimal hyperplane and associated margin which maximally separates multidimensional feature vectors onto either side of a classification boundary. If data are not linearly separable, the user can configure the framework's SVM module with radial basis function (RBF) and polynomial kernels to map input vectors into a higher dimensional space, enabling nonlinear decision boundaries in normally linear classifiers such as SVM. Basic implementations of SVM serve to classify, but do not by default provide probability estimates or multiclass classification. Extensions to SVM add the ability to assign an example to one of more than two classes by using a one-against-one approach in which the input vector is tested against all combinations of two classes in the set of classes. Using this approach, a binary classification algorithm can be applied to a multiclass classification problem by testing examples against K (K-1) / 2 individual binary classifiers in a system of K classes. Our initial implementations of the machine learning modules, however, provides only binary classification (healthy vs. stressed) but could be extended to

perform multiclass classification (e.g. low stress, medium stress, high stress) or multiclass identification (e.g. plant is infected with pythium, powdery mildew, mosaic virus, etc.) using these techniques. An example of modeling with various machine learning modules using global features extracted by the framework is explained in Section 4.1 which describes a water stress detection experiment performed on basil plants.

CHAPTER 3

FRAMEWORK ARCHITECTURE

3.1 Android application

A mobile application was developed for data acquisition and labeling. Currently, the app is only produced for mobile devices running the Android operating system. The Android app performs collection and labeling of data which are uploaded to the cloud for processing and classification by the microservices pipeline. It also provides functions for collecting lens calibration data, collecting and labeling optical imagery using the built-in camera, and collecting thermal and optical infrared imagery using the required FLIR One attachment. Future revisions of the application may be designed to support other thermal infrared or hyperspectral imaging devices as they come to market. The application was built using Android Studio with the FLIR One SDK for Android and, at the time of this writing, is available as an APK from the author (see Conclusion for contact information). The app is targeted for release in the Google Play app store in the summer of 2016.

3.2 Cloud infrastructure

The framework is designed as a microservices architecture. That is, each service is developed, maintained, and deployed individually in order to provide a high level of flexibility regarding choice of programming languages and approaches to module development. Each step in the data processing pipeline is packaged and deployed as a

service inside of a Docker container. A RabbitMQ message broker provides the cluster of containers with a common message bus for handling the flow of data through the processing pipeline. Each microservice subscribes to a processing queue where it listens for messages representing workloads to be executed. After each processing step, the next service in the pipeline is instructed to carry on the next processing step.

Deploying pipeline elements as individually containerized services supports portability and scalability, and perhaps most importantly, allows individual developers to use their preferred programming languages and code libraries when developing service modules. Containerized applications are more portable across host operating systems and are designed to overcome challenges of host machine configuration and dependency management. New containerized services can be quickly deployed without significant prior provisioning of the host system. Wrapping each microservice into its own Docker container allows each service in the pipeline to be scaled independently and load balanced as bottlenecks in processing are identified. For instance, if the ORB detector in the RegistrationService module is found to be a bottleneck in the pipeline, additional RegistrationService containers can be deployed and subscribed to a shared task queue, dividing jobs among a greater number of service workers, thus distributing the load of that specific task.

CHAPTER 4

FRAMEWORK TESTING & DISCUSSION

4.1 Basil water stress experiment

An experiment was performed to demonstrate the functionality of the pipeline when applied towards detection of water stress in basil plants. Before continuing, this section must be prefaced by stating that the results from the stress detection are not statistically significant and that the experiment is not to be considered scientific in its execution. That is, it was not conducted in a controlled environment, nor were the quantities of healthy and water-stressed plants sufficient to claim statistically significant results. The purpose of the experiment was to evaluate whether the data processing pipeline is functional, and to demonstrate how the framework could be used for a controlled experiment given proper resources and protocols in line with horticultural experiments of this type. Nonetheless, various models were trained and tested using data acquired and processed by the framework and the initial results are promising, validating the functionality of the framework and indicating that further experimentation should be performed to formally evaluate the framework's plant stress detection abilities. Furthermore, it should be noted that if the initially implemented modules we have provided do not achieve adequate performance for specific tasks, the modular and open-source nature of the framework provide a foundation for continuous improvement by project participants.

4.1.1 Methods

A group of 24 basil plants (*Ocimum basilicum*) in 2.5-inch pots were purchased from a local nursery. The plants were divided into training and test sets consisting of 16 and 8 plants, respectively, and were grown in an indoor environment under artificial lighting supplied by an AcroOptics CRAVE 24 Reef Flats LED light for 14 hours per day. All plants were watered with 100mL of water on Day 1. After Day 1, half of the plants in each set received between 50-100 mL of water, and the other half were deprived of water to induce stress. Stereo visual and thermal imagery of each plant were collected once per day using the FLIR One with our Android application on a Galaxy Note 5 and uploaded to the cloud for storage and pipeline processing. Data were collected for a period of 5 days.

The experiment produced 140 stereo optical pairs and 140 thermal images which were processed by the framework through a pipeline as configured in Fig X. Note that the disparity mapping module was not used in this experiment. The Block Matching disparity maps produced from the basil imagery acquired with the FLIR One optical camera were quite noisy and will need to be further tested and improved before being used with optical data from the FLIR One. On the other hand, disparity maps generated from imagery taken using the built-in smartphone camera of the Note 5 were of higher quality, perhaps due to the higher resolution and better picture quality of the smartphone camera (see Figure 14 for an example of a disparity map generated from stereo smartphone camera imagery). For this reason, only the left optical and thermal images were used in the experiment.

For each plant, a vector of 28 global features was produced using the Global Feature Extraction module. The features included the global mean and variance from the masked thermal data and each channel of the masked RGB, HSI, Lab, and CMYK color spaces. Models were trained from these feature vectors using the SVM, Linear Regression, Logistic

Regression, and Random Forest classifier modules. The SVM model was trained with an RBF kernel. The Random Forest Classifier was trained using 100 decision tree estimators, and test accuracy was calculated by taking the average test accuracy of 10 random forest models. For each training method, the full feature vector of 28 features was used. A second test used a subset of five features selected by identifying the top five most significant features used by the RF classifier during the first test. Models were trained using all data from Days 1-5, and model accuracy was evaluated against data from each day's test set separately, as well as combined data sets from Days 1-5, 2-5, and 3-5. By testing data from each day individually, we attempt to evaluate the early detection abilities of the model. In other words, we ask the question, "how soon after water deprivation, and how accurately, can the model classify plants as stressed?" Results are discussed below.

4.1.2 Results

Table 3 shows the performance of each learning method when tested with single-day data sets. Among the four machine learning methods used, the Random Forest classifier identified 100% of healthy and stressed plants accurately using data from Day 5. It also outperformed the other models with single-day data from Days 2-4, achieving average accuracies of 87.5%, 87.5%, and 97.5%, respectively. The Linear Regression classifier correctly identified the labeling of 75% of the plants for Day 1. The SVM model performed worst, achieving its highest accuracy of 62.5% on Day 5. In general, model accuracy was highest for the last days of the test, the time at which the plants were most severely affected by water stress. SVM and LogReg classifiers exhibited worse-than-random results for data sets from Days 1 and 2. This is likely due to a combination of the plants exhibiting

little stress in early days due to soil buffering capacity (making differentiation from healthy plants difficult) and the small size of the test set (8 plants per day), which, as mentioned earlier, is too small to make these results conclusive. Further experimentation using larger quantities of test data should be performed.

Day	SVM	LinReg	LogReg	Random Forest
1	0.125	0.75	0.375	0.5
2	0.5	0.625	0.25	0.875
3	0.5	0.625	0.625	0.875
4	0.5	0.625	0.875	0.975
5	0.625	0.75	0.5	1.0

Table 3: Accuracy of models tested using data (28 features) from each day individually.

After training the Random Forest Classifier with all 28 features, variable importances were calculated using the technique described by (Reiman 2001) in his original paper on Random Forests. The top five most important variables are listed in Table 4. Interestingly, the global thermal mean and variance were not very significant influences on the Random Forest model, ranking 12th and 14th, respectively. The most important feature was found to be the mean value of pixels in the green channel from the RGB color space in masked optical images. As the basil plants grew increasingly water-stressed, the leaves curled, creating shadowed areas within leaves that would lead to a darker shade of green being more prevalent in leaf imagery. Additionally, as leaves curled, the area covered by the canopy decreased, exposing more of the underlying soil. The exposed soil manifests as proportionally larger dark areas in optical images. We speculate that factors such as these

may have led to the color channel statistics having greater significance than thermal statistical features. An experimental setup that occluded the soil area with a material matching the dominant background color could help to reduce the influence of exposed soil regions on classification. Or, a better foreground isolation algorithm that can isolate the plant from both the background and the underlying soil could be developed to focus training on features derived only from pixels containing plant tissue.

Feature	Importance
rgb_g_masked_mean	1
cmyk_m_masked_var	2
lab_l_masked_var	3
cmyk_m_masked_mean	4
rgb_b_masked_var	5

Table 4: Top five variable importances from the Random Forest classifier.

A second training and testing phase was performed using these top 5 features only, and results are reported in Table 5. Limiting the dimensionality of these small data sets showed improvements in classification in some cases.

Day	SVM	LinReg	LogReg	Random Forest
1	0.5	0.625	0.5	0.4
2	0.625	0.75	0.625	0.575
3	0.625	0.75	0.625	0.988
4	0.5	0.875	0.5	0.975
5	0.75	0.875	0.75	1.0

Table 5: Accuracy of models tested using data (top five RF features) from each day individually.

Another test was performed using grouped data from Days 1-5, Days 2-5, and Days 3-5. Results are reported in Table 6 and charted in Figure 15. As expected, model accuracy was best when tested on data from later days when water-stressed plants exhibited greater changes in morphology compared to earlier days.

Days	SVM	LinReg	LogReg	Random Forest
1-5	0.6	0.775	0.6	0.775
2-5	0.688	0.813	0.594	0.895
3-5	0.67	0.917	0.583	0.958

Table 6: Accuracy of models tested using data from multiple days.

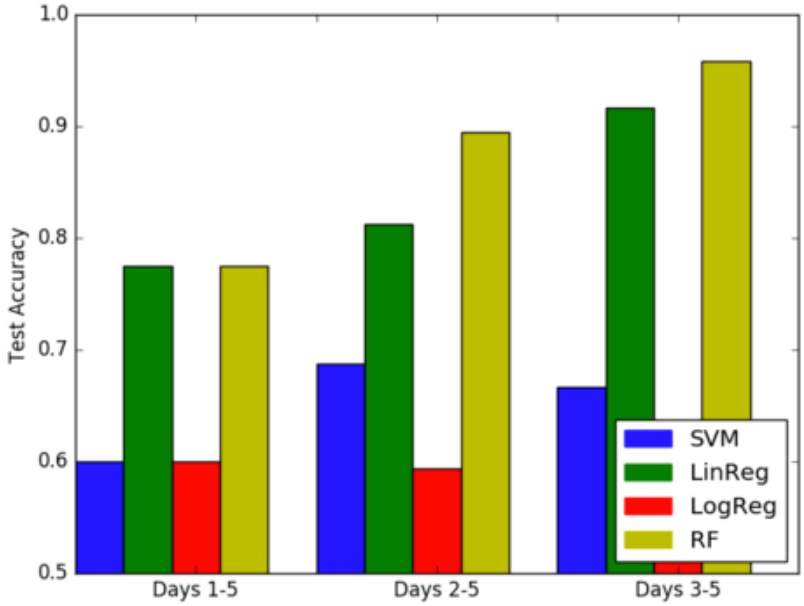

Figure 15: Comparison of model accuracies using data from multiple days.

4.2 Risk and reward: challenges of a public project

Community involvement and continuous improvement are both essential ingredients for success in citizen science and open source projects. Two-sided network effects and economies of scale are some of the challenges faced by social networking startups and citizen science projects, and are present in this project as well. Before the technology is usable by the mainstream, a steady R&D effort is required, which in turn depends on contributions from the mainstream. The long-term viability of the project will be realized at scale, but only if the two-sided network thrives and all stakeholders find value in its use. The future of the framework will depend on contributions from both sides: the mainstream plant-growing community submitting labeled imagery, and the researcher/developer community improving framework modules and producing usable models and applications

that return value to plant producers. This leads to a situation of both high risk and high reward. Here we explore a few topics for consideration.

4.2.1 Expert participation

Before the system can offer value to producers in the form of automated tools for stress detection, at least a few models will need to be developed and proven in field conditions. The data for these early improvements will likely come entirely from the expert communities already involved in this type of research. Initial outreach will be made to researchers, encouraging those with existing data sets to upload their data to the cloud and to integrate their development efforts into framework modules, establishing a repository of pipelines and models that have already been validated in experimental settings. This is the first hurdle—persuading researchers to release their data sets and to allow potentially proprietary algorithms to be integrated into the framework and made available to the public. Proper attribution and perhaps licensing arrangements to protect intellectual property will be important areas of consideration for the research community. Source code encryption for proprietary algorithms may also be an option in certain cases if contributors are otherwise not comfortable integrating their work. If even a small user base of committed early adopters can be cultivated, the rewards will come in the form of a growing repository of optical and thermal imagery, case studies, expert feedback, and traction to encourage new users to participate.

4.2.2 General participation

The second hurdle involves the other side of the network; the community at large will need to be exposed to the technology, understand how to use it effectively for data collection and analysis, and derive value from using it in the field. One way to reach the public may be through strategic partnerships with organizations that already interface with the public. For example, Extension Service professionals and Master Gardener program participants have responsibilities imposed by their occupation or certification requirements to assist growers with diagnosing and mitigating plant stress in crops. Exposing these organizations to our project could lead to a pathway for integration into field work by leveraging existing relationships those organizations have with the larger plant-growing community.

4.2.3 Quality control

Data submitted by users is at risk of mislabeling and other quality-control issues. To mitigate these risks, general use tutorials and training materials should be developed. In the case of mislabeled or unlabeled data, experts should be incentivized to assist with correcting errors in the public data set. For general data sanitation tasks that require little to no expertise, services such as Amazon's Mechanical Turk could be utilized. Participants could also assist with metadata tagging of user-submitted imagery similar to the way citizen science project Galaxy Zoo teaches users with little to no astronomy knowledge to classify galaxies according to their shape.On the R&D side, some standard measure of performance will be needed to compare modules and rank their effectiveness in various

applications. In the computer vision subfield of disparity mapping, for example, researchers test algorithms against stereo images from the Middlebury data set. In object recognition research, ImageNet is a typical standard data set. Curating data sets like these, but specific to plant stress detection research, may be useful in developing a standardized measure to compare and evaluate module and pipeline performances.

4.2.4 Overhead costs

Operating the system in a publicly accessible production environment on the internet comes with hosting, bandwidth, and storage costs. These costs are currently being covered by the author, but as the system scales, overhead will increase, and additional funding will be needed to support the project. This may come in the form of small donations by users, public and private grants, and strategic partnerships with research institutions and industry players.

4.3 Future work

Immediate improvements are targeted at the registration and disparity mapping modules. Plant foreground isolation could be improved by implementing the Stationary Wavelet Transform approach used by Raza et al. (2015) to isolate tomato plant tissue from background pixels. Disparity mapping is a well-studied and active area of computer vision research, and newer, more robust approaches such as BM+DP introduced by Nguyen (2014)

should be implemented to improve the framework's extraction of 3D information from stereo imagery.

The framework is currently only accessible by submitting requests to the HTTPS API, making it accessible to users with programming experience. However, a graphical web interface is being developed at the time of this writing and should continue to be developed to enable non-programmers to interact with and utilize the framework. When the web interface is functionality, we can begin outreach efforts to engage with individuals and organizations in order to cultivate an active community around the project.

4.4 Conclusion

We have presented a novel Android and cloud-based framework designed to support data collection, research, and application development in the field of remote plant stress detection. An initial experiment using the framework shows that the framework and its pipelines function as intended but will require more data, experimentation, and continued improvement by contributors to realize its full potential. It is our hope that by providing this resource, we can begin to cultivate a common repository of stereo optical and thermal imagery of plants in various settings and conditions around the world. These data sets can be used by researchers to create robust models that are accessible by plant producers in the mainstream community through a variety of user-friendly mobile and web-based applications.

Source code for the Android application and processing modules are available on the author's Bitbucket page at https://bitbucket.org/dzukowski-masters-thesis/. Interested

contributors are welcome to contact the author at daniel.zukowski@gmail.com to get involved in the project.

REFERENCES

1. ALink Computer Solutions Inc. 2014. A&L Plant Disease Diagnosis (Version 1.30) [Mobile application software]. Retrieved from https://play.google.com/store/apps/.

2. Allègre M, Daire X, Héloir MC, Trouvelot S, Mercier L, Adrian M, Pugin A. 2007. Stomatal deregulation in Plasmopara viticola-infected grapevine leaves. New Phytologist. 173: 832–840.

3. Amodei, Dario, et al. 2015. "Deep Speech 2: End-to-End Speech Recognition in English and Mandarin." arXiv preprint arXiv:1512.02595.

4. Belin E, Rousseau D, Boureau T, Caffier V. 2013. Thermography versus chlorophyll fluorescence imaging for detection and quantification of apple scab. Computers and Electronics in Agriculture. 90: 159–163.

5. Biskup B, Scharr H, Schurr U, Rascher U. 2007. A stereo imaging system for measuring structural parameters of plant canopies. Plant, Cell & Environment. 30: 1299–1308.

6. Blum A, Mayer J, Gozlan G. 1982. Infrared thermal sensing of plant canopies as a screening technique for dehydration avoidance in wheat. Field Crop Research. 5: 137–146.

7. Bock CH, Parker PE, Cook AZ, Gottwald TR. 2008. Visual rating and the use of image analysis for assessing different symptoms of citrus canker on grapefruit leaves. Plant Disease. 92: 530-541.

8. Bock CH, Poole GH, Parker PE, Gottwald TR. 2010. Plant disease severity estimated visually, by digital photography and image analysis, and by hyperspectral imaging. Critical Reviews in Plant Science. 29: 59-107.

9. Bradski, G. 2000. The opencv library. Doctor Dobbs Journal. 25 (11): 120-126.

10. Breiman, L. 2001. Random Forests. Machine Learning 45 (1): 5–32.

11. BugGuide. Retrieved March 15, 2016 from http://bugguide.net.

12. Camargo A, Smith JS. 2009. Image pattern classification for the identification of disease causing agents in plants. Computers and Electronics in Agriculture. 66: 121-125.

13. Canny J. A Computational Approach To Edge Detection. 1986. IEEE Transactions on Pattern Analysis and Machine Intelligence. 8 (6): 679–698.

14. Chan Y, Gimel'farb G, Delmas P, Valkenburg R. 2009. Accurate 3D modelling by fusion of potentially reliable active range and passive stereo data. Proceedings of the Computer Analysis of Images and Patterns. Munster, Germany. 848–855.

15. Conrad CC, Hilchey KG. 2011. A review of citizen science and community-based environmental monitoring: issues and opportunities. Environmental Monitoring and Assessment. 176: 273–291

16. Docker, Inc. 2016. Docker (Version 1.10.3) [software] Retrieved from https:// www.docker.com.

17. Elasticsearch. 2016. Elasticsearch (Version 2.3.1) [software]. Retrieved from https:// www.elastic.co/products/elasticsearch/.

18. FAO's Director-General on How to Feed the World in 2050. 2009. Population and Development Review. 35 (4): 837-39.

19. Fischler M, Bolles R. 1981. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography. Communications of the ACM. 24 (6): 381–395.

20. FLIR One Software Development Kit. Retrieved Jan 10, 2015 from http:// developer.flir.com/flir-one-software-development-kit/.

21. FLIR Systems, Inc. Retrieved March 15, 2015, from http://www.flir.com.

22. FLIR Systems, Inc. 2015. FLIR One Software Development Kit for Android (Version 1.0.3) [Mobile application development software] Retrieved from http:// developer.flir.com/flir-one-software-development-kit/.

23. Flynn P. 2003. Biotic vs. Abiotic – Distinguishing Diseases Problems from Environmental Stresses. Horticulture and Home Pest News. IC-489 (22).

24. Fusiello A, Irsara L. 2008. Quasi-Euclidean uncalibrated epipolar rectification. 19th International Conference on Pattern Recognition. Tampa, FL. 1-4.

25. GalaxyZoo. Retrieved March 15, 2016 from https://www.galaxyzoo.org.

26. Gimel'farb GR. 2002. Probabilistic regularisation and symmetry in binocular dynamic programming stereo. Pattern Recognition Letters. 23 (4): 431-442.

27. Goeau H, Joly A, Selmi S, Bonnet P, Mouysset E, Joyeux L, Molino JF, Birnbaum P, Bathelemy D, Boujemaa N. 2011. Visual-based plant species identification from crowdsourced data. 19th ACM international conference on Multimedia: 813–814.

28. Gomez S. 2014. Infection and spread of Peronospora sparsa on Rosa sp. (Berk.) - a microscopic and a thermographic approach. Dissertation, University of Bonn, Germany.

29. Hajiboland R. 2012. Effect of micronutrient deficiencies on plants stress responses. In: Ahmad P, Prasad MNV (eds) Abiotic stress responses in plants. Springer, New York. 283–329.

30. Hampton RO, Ball EM, De Boer SH. 1990. Serological Tests for Detection of Viral and Bacterial Pathogens. St Paul. MN: APS Press.

31. Hansson DH. 2015. Ruby on Rails (Version 4.2.6) [software]. Retrieved from http:// rubyonrails.org.

32. iSpot. Retrieved March 15, 2016 from http://www.ispotnature.org.

33. Jones HG, Serraj R, Loveys BR, Xiong L, Wheaton A, Price AH. 2009. Thermal infrared imaging of crop canopies for the remote diagnosis and quantification of plant responses to water stress in the field. Functional Plant Biology. 36: 978-989.

34. Kaiming H et al. 2015. Deep Residual Learning for Image Recognition. arXiv preprint arXiv:1512.03385.

35. Knipling EB. 1970. Physical and physiological basis for the reflectance of visible and near-infrared radiation from vegetation. Remote Sensing of Environment. 1: 155–159.

36. Kumar S, Micheloni C, Piciarelli C, and Foresti GL. 2010. Stereo rectification of uncalibrated and heterogeneous images. Pattern Recognition Letters. August 2010.

37. Lamari L. 2008. Assess 2.0: Image Analysis Software for Plant Disease Quantification (Version 2.0). [Desktop application software]. Retrieved from http://www.apsnet.org/ apsstore/shopapspress/Pages/43696m5.aspx.

38. Li R, Zheng B, Liou ML. 1994. A New Three-Step Search Algorithm for Block Motion Estimation. IEEE Transactions on Circuits And Systems For Video Technology. 4 (4): 438–442.

39. Lindenthal M, Steiner U, Dehne HW, Oerke EC. 2005. Effect of Downy Mildew Development on Transpiration of Cucumber Leaves Visualized by Digital Infrared Thermography. Phytopathology. 95: 233-240.

40. Longuet-Higgins HC. 1981. A computer algorithm for reconstructing a scene from two projections. Nature. 293 (5828): 133–135.

41. Mahlein, A-K. 2016. Plant Disease Detection by Imaging Sensors - Parallels and Specific Demands for Precision Agriculture and Plant Phenotyping. Plant Disease. 100 (2): 241-251.

42. MathWorks, Inc., The. 2016. MATLAB and Statistics Toolbox Release 2016a (Version 2016a) [Desktop application software]. Retrieved from http://www.mathworks.com/ products/matlab/.

43. Neumann M, Hallau L, Klatt B, Kersting K, Bauckhage C. 2014. Erosion band features for cell phone image based plant disease classification. Proceeding of the 22nd International Conference on Pattern Recognition (ICPR), Stockholm, Sweden, 24-28 August 2014. 3315-3320.

44. Ng A. Keynote Presentation: Deep Learning. GPU Technology Conference. San Jose, CA. 19 Mar. 2015. Lecture.

45. Numpy developers. 2016. NumPy (Version 1.11.0) [software]. Retrieved from http://www.numpy.org/.

46. Oerke E-C, Lindenthal M, Fröhling P, Steiner U. 2005. Digital Infrared Thermography for the Assessment of Leaf Pathogens. The 5th European Conference on Precision Agriculture. Uppsala. 9-12 June 2005. 91-98.

47. Oerke E-C, Steiner U, Dehne H-W, Lindenthal M. 2006. Thermal Imaging of Cucumber Leaves Affected by Downy Mildew and Environmental Conditions. Journal of Experimental Botany. 57: 2121-2132.

48. Papadimitriou D, Dennis T. 1996. Epipolar line estimation and rectification for stereo image pairs,. IEEE Transactions on Image Processing. 5 (4): 672-676.

49. Pedregosa et al. 2011. Scikit-learn: Machine Learning in Python. JMLR. 12: 2825-2830.

50. Photosynq. Retrieved March 15, 2016 from http://www.photosynq.org.

51. Pivotal Software, Inc. 2016. RabbitMQ (Version 3.6.1) [software]. Retrieved from https://www.rabbitmq.com/.

52. PlantNet-Project.org. 2015. PlantNet Plant Identification (Version 1.5.0) [Mobile application software]. Retrieved from https://play.google.com/store/apps/.

53. PostgreSQL Global Development Group. 2016. PostgreSQL (Version 9.5.2) [software]. Retrieved from http://www.postgresql.org/.

54. Prithiviraj B, Vikram A, Kushalappa AC, Yaylayam V. 2004. Volatile metabolite profiling for the discrimination of onion bulbs infected by Erwinia carotovora ssp. carotovora, Fusarium oxysporum and Botrytis allii. European Journal of Plant Physiology. 110: 371–377.

55. Project BudBurst. Retrieved March 15, 2016 from http://budburst.org.

56. RabbitMQ. Retrieved March 15, 2016 from https://www.rabbitmq.com.

57. Roberts MJ, Schimmelpfennig D, Ashley E, Livingston M, Ash M, Vasavada U. 2006. The value of plant disease early-warning systems. Economic Research Service. 18.

58. Rublee E, Rabaud V, Konolige K, Bradski G. 2011. ORB: an efficient alternative to SIFT or SURF. Computer Vision (ICCV), 2011 IEEE International Conference on.

59. Saettler AW, Schaad NW, Roth DA. 1989. Detection of Bacteria in Seed and Other Planting Material. St Paul, MN: APS Press.

60. Safecast. Retrieved March 15, 2016 from http://blog.safecast.org.

61. Schaad NW, Frederick RD. 2002. Real-time PCR and its application for rapid plant disease diagnostics. Canadian Journal of Plant Pathology. 24 (3): 250–258.

62. Schaad NW, Jones JB, Chun W. 2001. Laboratory Guide for Identification of Plant Pathogenic Bacteria. St Paul, MN: APS.

63. Scikit-learn. Available from: http://scikit-learn.org/stable/.

64. Schmidhuber J. 2015. Deep Learning in Neural Networks: An Overview. Neural Networks. 61: 85-117.

65. Serra J. 1982. Image Analysis and Mathematical Morphology. Academic Press.

66. Szegedy C et al. 2015. Going deeper with convolutions. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

67. TeamSOA, Inc. 2016. Garden Answers Plant Identification (Version 1.3.9) [Mobile application software]. Retrieved from https://itunes.apple.com.

68. United Nations Department of Economic and Social Affairs Population Division [(accessed on 15 January 2016)]. Available online: http://www.unpopulation.org.

69. Wijekoon CP, Goodwin PH, Hsiang T. 2008. Quantifying fungal infection of plant leaves by digital image analysis using Scion Image software. Journal of Microbiological Methods. 74: 94-101.

70. Yahyanejad S, Misiorny J, Rinner B. 2011. Lens distortion correction for thermal cameras to improve aerial imaging with small-scale UAVs. IEEE International Symposium on Robotic and Sensors Environments (ROSE).

71. Zhang Z. 2000. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence. 22 (11): 1330-1334.