AUTOMATIC CANOPY PLOT BOUNDARY DETECTION USING COMPUTER VISION

A Thesis

by

DE KWAAN WYNN

BS, Baylor University, 2017

Submitted in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in

COMPUTER SCIENCE

Texas A&M University-Corpus Christi

Corpus Christi, Texas

May 2020

May 2020

AUTOMATIC CANOPY PLOT BOUNDARY DETECTION USING COMPUTER VISION

A Thesis

by

DE KWAAN WYNN

This thesis meets the standards for scope and quality of Texas AM University-Corpus Christi and is hereby approved.

Scott King, PhD
Chair

Dr. Xavier Gonzales, PhD
Co-Chair

Dr. Longzhuang Li, PhD
Committee Member

May 2020

ABSTRACT

In Corpus Christi, Texas the United States Department of Agricultre (USDA) funded Texas A&M Agrilife research on large farmlands with hundreds of individual cotton vegetation plots. Each plot is planted uniformly in rows but not all plots grow at the same rate. Every week the plots are photographed using an Unmanned Aircraft System (UAS) flying at a height of 100 feet to record and evaluate growth for various reasons. The research scientists and farmer's current method of localizing individual plots of vegetation within an image has proven to be very time consuming and inefficient. The algorithm developed in this paper automates the localization process under sunny conditions. The algorithm uses the Hue, Saturation, and Value (HSV) color space during the preprocessing stage to provide a binary image that indicates where each green pixel is located. Various OpenCV functions are then used to automate the crop localization process. Minimum and Maximum threshold values are set for filtering by size sections of the algorithm. Then, morphological operations are employed to further refine the regions of interest. The Connected Components function is used to determine how large each remaining object is and that size is then used to determine how large each localizing polygon will be drawn. After the size of each object is found, the size of each localizing polygon to be drawn is evaluated and split into smaller polygons whenever necessary. Not only the developed algorithm able to detect and classify cotton crop locations quickly but it is able to handle various complex situations. The developed method was evaluated by its accuracy, precision, and recall, which were 92.4%, 100%, and 92.4% respectively.

ACKNOWLEDGMENTS

I would like to thank Dr. Scott King, Dr. Xavier Gonzales and Dr. Longzhuang Li for the imparting knowledge to me that helped me get to this point in my education. Without Dr. Gonzales I wouldn't have gotten the opportunity to conduct this research. I appreciate Dr. King for supporting me and guiding me during the thesis writing process. I also would like to thank Dr. Jinha Jung along with Dr. Sungchan Oh for welcoming me into Texas A&M Agrilife with open arms and providing me with this research opportunity. They pushed me into the right direction when I had questions and allowed me to encouraged me to try out new ideas whenever I had them.

I would also like to give a very special thanks to the United States Department of Agriculture for funding this research opportunity. This opportunity has provided me with the ability to travel and see things I thought I'd never see and has opened many doors for me that I didn't even know existed. The experiences and connections I created through this research will stay with me for forever.

This work is supported by Hispanic Serving Institutions Education [grant no. 2016-38422-25543/project accession no. 1009881] from the USDA National Institute of Food and Agriculture.

Any opinions, findings, conclusions, or recommendations expressed in this publication are those of the author(s) and do not necessarily reflect the view of the U.S. Department of Agriculture.

United States
Department of
Agriculture

National Institute
of Food and
Agriculture

TABLE OF CONTENTS

CONTENTS PAGE

LIST OF FIGURES

FIGURE PAGE

LIST OF TABLES

TABLE PAGE

1. Introduction

Aerial photographs are used in the agriculture industry for many reasons, including examining ground conditions for fertility rate recommendations, disease detection, assessing water management, quantifying soil compaction, etc. There are three primary sources of capturing these images:

- Satellite Imaging - Covers largest area, but limited by weather. The satellite capable of the highest resolution has a ground resolution of 1.65-meter resolution(64 inches).
- Plane Imaging - Covers more ground than Unmanned Aerial System Imaging but most expensive method. Plane imaging is capable of a resolution up to 30 centimeters per pixel.
- Unmanned Aerial System (UAS) Imaging - needs more images to take photos of full fields but it isn't effected by weather and is much cheaper than Plane imaging. A UAS equipped with a 42MP camera has a ground resolution of 0.8 cm (0.3in per pixel)

Each method has its uses, advantages, and disadvantages.

The goals of the research are:

- Develop a method that will automatically draw bounding polygons around plots of cotton vegetation quickly.
- Develop a method that will be able to work with multiple images with cotton crops of different orientations and lighting conditions, aside from any image taken at night time.
- Develop a method that will not have difficulty separating plots that grow together and/or appear as one large plot.

The images used for the purpose of this study are taken via UAS. The images captured are plots of cotton vegetation. When photographing these plots of vegetation from above there may be 100 plots within a single image. Farmers and researchers sometimes need each individual plot to be located and localized within each image. A plot is defined as a collection of vegetation so close in proximity to each other that, when viewed from above, appears as one uniform group. The current method of plot localization being used by Texas A&M AgriLife research scientists consists of using the computer mouse to draw a rectangle around the plots of vegetation in the images captured to indicate where the each plot of vegetation is located. Even though it is accurate, there can be hundreds, or thousands, of plots in one image when combined into a large dense cloud, and if the images were taken with a satellite there could be even more depending on how large the field is. Therefore, the method isn't time or energy efficient, especially when there's a large number of images. The developed method was created to automate the localization process, thus making localizing less time consuming and less labor intensive. In order for the algorithm to be effective it must be able to localize no matter the orientation of the plots of vegetation within each image in various lighting conditions.

The data used for the research is a culmination of images captured using UASs. Every week the Texas A&M University AgriLife Center go to farmlands and gather images of their crops to track their progress/yield. See Figure 1.1 for an example of the UAS images captured for the purpose of the research. The plots in one image may differ in orientation of another due to the variations in the angles the UAS captured the images in. The developed method localizes the vegetation effectively regardless of the plot orientation.

The research contribution is:

- Developed a simple and efficient algorithm to automatically localize cotton vegetation in aerial images

The next section explains the prior work found and utilized for the purpose of this re-

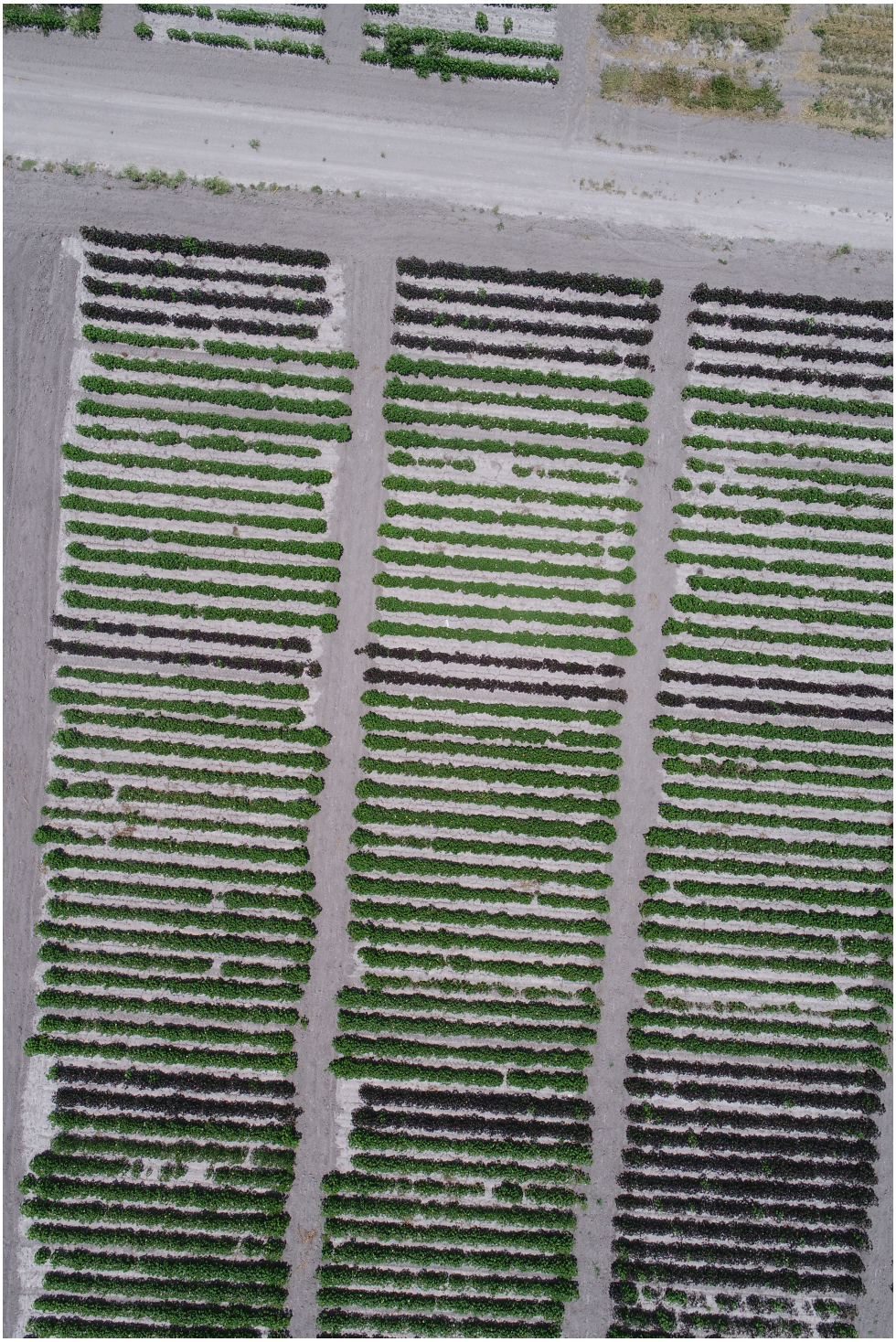

Figure 1.1: UAS captured photo of a segment of a cotton field.

search. Afterwards, the proposed methodology is explained in depth. The proposed methodology utilizes the Hue, Saturation, Value (HSV) color model to convert the images to binary images, detect the contours and apply multiple filters based on size of objects to determine which objects to localize. After the methodology is expanded upon the challenges encountered during the research and the developed solutions to those challenges are explained. Then, the final two sections consist of the results/analysis section and the summary of the research.

2. Background

For years researchers have been using computer vision to improve the farming and farm research efficiency. The researchers often develop systems to identify and localize crops and they use either machine learning or non machine learning methods. Also different color models are used during their development. This section will discuss some of the methods created by other researchers that were studied during the algorithm's development They are broken into three categories:

- Methods using HSV Color Model
- Methods using RGB Color Model
- Methods using Machine Learning

2.1 Methods using HSV Color Model

The HSV (Hue, Saturation, Value) color model is a type of color representation method that appears as a cone or cylinder, as shown in figure 2.1. The three components to the model are Hue, Saturation and Value. The first of the components, Hue, is the color portion of the model. This is a numerical value that ranges from 0 to 360 degrees. The colors Red, Yellow, Green, Cyan, Blue, and Magenta are the colors included. The colors is break the Hue down into the 60 degree increments shown below.

- Red is from 0 to 60 degrees
- Yellow is from 61 to 120 degrees
- Green is from 121 to 180 degrees
- Cyan is from 181 to 240 degrees

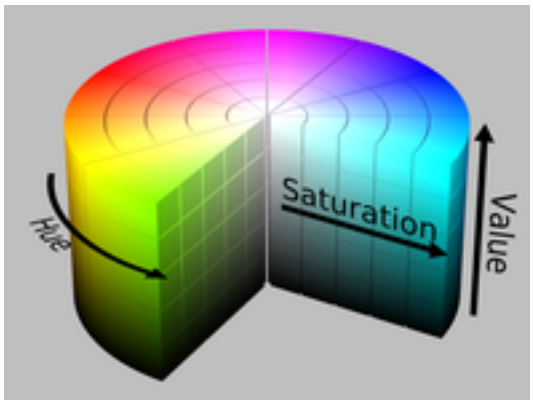

Figure 2.1: The HSV color model.

- Blue is from 241 to 300 degrees
- Magenta is from 301 to 360 degrees

The second part of the HSV model is a numerical description of the amount of gray in a certain color. The numerical range is from 0 to 100 percent. The closer the number is to zero the more faded, or less saturated, a color will appear. Another description of Saturation is the amount of dominance a particular hue has in a specific color. The final part of the HSV model is the Value portion, which works in conjunction with saturation and is also a numerical representation shown as a percentage. The range is from 0 to 100 percent and describes the brightness/intensity of the color. 0 percent is representative of black, or being devoid of brightness, and 100 percent is representative of the most color or brightness.

Crop Row Detection Procedure using Low-Cost UAV Imagery System

Hassanein et. al [1] proposed a method of detecting crop row and row orientation by using UAVs to take overhead photos of crops at different heights then applying computer vision techniques. The method proposed was evaluated based on its effectiveness in determining orientation despite height of the images and illumination of the images. The researchers took the original image and instead of converting it to a binary image, they converted the image to HSV images. Then the Hue value was extracted and used to detect the orientation

by evaluating the hue values of each row. If the hues were the same then the method made the assumption that the objects belonged to the same group of pixels. Then the Principle Component Analysis (PCA) tool was used to detect the orientation of the data collected.

Automatic crop detection under field conditions using the HSV colour space and morphological operations

Hamuda et. al. [2] developed an automatic weed detection/segmentation algorithm using computer vision techniques to find the exact location of weeds. Their method relies heavily on morphological operations and HSV color features. The proposed method gives researchers a way to distinguish between crops and weeds. The method used video data from a camera moving along rows of cauliflower seedlings.

2.2 Methods using RGB Color Model

The RGB (Red, Green, Blue) color model is a three part, additive method for representing colors. The model is split into three components: Red, Green, & Blue. The three components are added together in different proportions to produce a range of colors. The resulting color image is a composite of three gray scale images that correspond to the red, green and blue light intensities. RGB is not a true representation of how colors in real life are created but it serves as an effective way to represent them digitally and is highly effective when used in sensor and Image processing techniques. A visual representation of the RGB color space is shown in figure 2.2.

Identification and Classification of Bulk Fruits Images using Artificial Neural Networks

Dayan and Savakar [3] created an identification and classification method of bulk fruit images. There were five categories of fruit. They used a feed forward back propagation neural network. Instead of just using color to classify images, texture was extracted and combined with color to make a more accurate classification. Eighteen color features and twenty-seven texture features were extracted by their algorithm. For color hue, intensity, and saturation features were extracted from the RGB components. Mean, range, entropy, variance, and

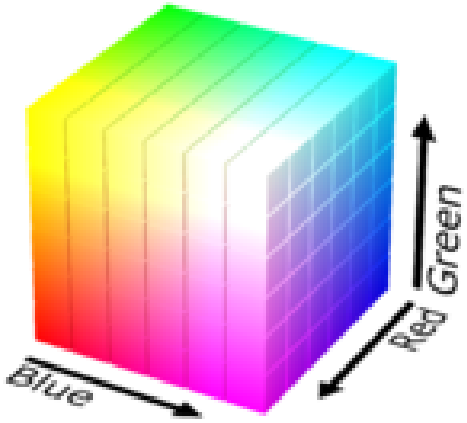

Figure 2.2: The RGB Color Model.

other features were extracted for texture using a grey level co-occurrence matrix. An ANN with one input and output layer was used for bulk fruit classification. In the article a comparison was made between the three methods (color, texture, color + texture). When they only use color to compare there was a lower accuracy than when they used texture. But both methods were not as accurate when they used both color and texture together to classify the fruit images. The method had an accuracy as high as a 94 percent success rate.

Canopeo: A powerful new tool for measuring fractional green canopy cover

Fractional green canopy cover (FGCC) is used around the world to estimate canopy development, light interception and various other things. Andres et al. [4], created a quick and accurate tool to analyze FGCC from images and videos, and they called it Canopeo. The method is based on using the RGB values of the image's individual pixels. The criterion is listed below.

- Red/Green
- Blue/Green
- 2Green – Red – Blue (Excess Green Index)

The Canopeo algorithm used a binary classification to determine if each pixel was green or not green. A separate image was created to display the results.

First, the image was read and the pixels RGB values were placed into a numpy array. Then, a criterion used to determine if the pixel was green or not was applied. If the pixel satisfied all three conditions, it would be considered green, otherwise it would be non-green. The output from Canopeo was compared to SamplePoint and SigmaScan Pro, which are two other software packages used to analyze FGCC. Canopeo proved to be extremely faster than both of them and was found to be more accurate.

Illumination invariant segmentation of vegetation for time series wheat images based on decision tree model

Guo et. al. [5] proposed a segmentation model based on machine learning, decision tree and noise reduction filters using the RGB color model. It had no trouble dealing with shadowed areas and doesn't require thresholding at each image. The images used were images taking of wheat. The researchers used the CART algorithm to create the decision tree and their method was compared to many ExG methodologies for evaluation.

Mean-Shift-Based Color Segmentation of Images Containing Green Vegetation

Zheng et al. [6] developed an image segmentation method that used Back Propagation Neural network to segment the image into green pixels and non-green pixels. The method used the Red, Green, Blue values in the RGB color model for evaluations, mean shift segmentation. The method consists of two stages, feature extraction and segmentation. The researchers used 100 images with different plant types, illuminations and soil types. Median mis-segmentation was 4.2%.

Color indices for weed identification under various soil, residue, and lighting conditions

Woebbecke et. al. [7] used the RGB color model to analyze images of vegetation. Because they understood the color values of greenery and the soil had to be different, they developed a methodology to distinguish between the plant and non-plant background. The original

purpose for this was to identify weeds amongst the soil, but the indices can be used for more than weed identification. A modified hue was also used for differentiation, but it was more expensive computationally.

Crop Growth Estimation System Using Machine Vision

Kataoka et. al [8] used RGB values to segment images of crops as a preprocessing method for growth estimation. The principal component analysis was used to separate the green pixes from the background (soil) pixels. A certain equation was applied to the R,G,B values to get what they called the Color Index of Vegetation (CIVE). This CIVE value was used to obtain the resulting binary image. This information was used to determine the vegetation cover area of each plant.

An automated, high-throughput plant phenotyping system using machine learning based plant segmentation and image analysis

Lee et al. [9] conducted research for the purpose of acquiring plant images rapidly with little difficulty. An automatic plant phenotype system was created using simple hardware and a machine learning based segmentation method. The method used for segmentation was required to be able to differentiate between plant and background, considering the plant and background may be the same color. Also, it had to take into consideration that the light conditions may not be the same. The machine learning method places each pixel into groups and the color features are extracted. These features are placed into a trained color classifier and the result is a binary image indicating where the images plant and background pixels exist.

Soybean Canopy Cover Measured with Canopeo Compared with Light Interception

Considering Canopeo had been compared to, and was found to be a faster, more accurate, way of quantifying FGCC than other software used around the world Shepherd et al. [10] conducted research to compare the Canopeo method to line quantum sensor for light interception measurements. Both methods proved to have their advantages and dis-

advantages. Canopeo was faster at calculating a canopy cover percentage and proved that it could be applied easily in the field. Canopeo had a problem detecting very dark plants even though you could adjust the equation to find different colors. With the line quantum sensor, data collection time per plot was variable becasue of cloud cover. This meant it was important to take the photos in the full sun to avoid fluctuation of sunlight levels. When light interception was used it found all colors of plants in all conditions. To conclude, the authors found there was a linear relationship between Canopeo and line quantum sensor. The relationship showed, because of the speed, using Canopeo was a good alternative to using light interception methods for canopy cover measurement.

Using Digital Image Analysis to Describe Canopies of Winter Oilseed Rape (Brassica napus L.) during Vegetative Developmental Stages

Behrens et al. [11] used digital image processing to predict characteristics in a winter oilseed rape canopy. They used a large number of images that were taken from 2002 to 2003 measuring an area of 1 square meter. The images were evaluated by a program that analyzed the Red-Green-Blue color channels of the pixels and determined the number of green pixels compared to the total amount of pixels. This information was used to determine canopy structure that could be used to determine soil coverage, leaf is index (LAI), and dynamics of plant number during developmental stages. They also determined the number of plants per square meter. They found that during the developmental stages emergence lasted thirty days which resulted in large differences in growth/development of individual plants. The photos were taken at a 2 meter height and centered above each plot.

When measuring the dynamics of soil coverage they found that the rate of coverage changed depending on the daily average temperature. Because it was wintertime they found that the leaves ends rolled up because of frost and some small plants were completely lost. The seed emergence wasn't uniform so that contributed to the differences within the canopy. The delayed emergence caused smaller plants to die in the winter.

Quantitative Analysis of Cotton Canopy Size in Field Conditions Using a Consumer-Grade RGB-D Camera

Manual assessment of canopy size is laborious and imprecise, and cannot measure multi-dimensional traits such as projected leaf area and canopy volume. Jiang et al. [12] developed a method to expediate the process. The goal was to create a 3D imaging approach to analyze quantitatively, cotton canopy development in field conditions. The field worked on for the research consisted of 128 plots with four genotypes that had thirty two plots for each one. The researchers scanned the field with a program called GPhenoVision. GPhenoVision is a customized field based high throughput phenotyping system). The program acquires color and depth images using GPS information. The field was scanned during two phases

- Canopy Development
- Flowering/boll development

The data processing pipeline developed for the method of extraction consisted of three steps

1. Point cloud reconstruction - This step was done using color and depth images with GPS information.
2. Plant canopy segmentation – This was done using an excess-green (ExG) color filter and the weeds were further separated from the canopies based on height, size and position.
3. Trait extraction – morphological traits were extracted daily (maximum and mean canopy height and width, projected canopy area, and concave and convex volumes)

Growth rate was also calculated for the purpose of quantifying growth and development of canopy.

Crop Detection by Machine Vision for Weed Management

Ashitosh et. al. [13] developed a way to detect weeds in onion, corn and sugar cane crops. The method developed used images collected in JPEG format from different angels and found their excessive green values of each pixel using an equation they developed themselves. The result of the exessive green equation was a masked image that turned everything that wasn't a green pixel to a black one. Next, they enhanced the image, used filtering to remove noise, labeled each group of pixels, then used size based feature extraction to distinguish between weed pixels and non weed pixels.

2.3 Methods using Machine Learning

Machine learning is a branch of AI (Artificial Intelligence) that's based on machines identifying patters and making decisions with little to no human interaction. There are different machine learning methods:

- Supervised learning- Algorithms are trained using labeled data/examples. The desired output is known before hand when using the labeled examples. The algorithm is also given the correct outputs with corresponding sets of inputs and it learns by comparing its actual output with the expected.

- Unsupervised learning - The algorithm isn't trained with labeled data/examples. This leaves the algorithm to figure out, on its own, what it is being shown.

- Semi-supervised learning - Uses both labeled and unlabeled data for training. Normally the amount of unlabeled data is larger than the labeled. It is typically used for classification, regression and prediction.

- Reinforcement learning - This type of learning is often used for robotics, navigation, and gaming. The algorithm figures out, through trial and error, which actions are the right ones, or which ones give the highest rewards over a given amount of time. Reinforcement learning has 3 components:

- Agent - Learner/Decision maker.
- Environment - What the agent has to interact with.
- Actions - What the agent does.

With the implementation of machine learning into the agriculture realm many advancements have been made. Using different, high-precision algorithms, has been key to increasing the quantity and quality of agricultural products worldwide in differing situations.

2017 Third International Conference on Sensing, Signal Processing and Security (ICSSS) Shreya Lal and the other authors [14] saw that the selection of apples based on ripeness was very time consuming and caused eye fatigue, so they proposed a new approach to solve the problem. They created a method, using machine learning and image processing, to detect the apples and classify them based on ripeness by using their RGB values. Neural Networks emulate the human brain and the RGB color model emulates the human eye. Color images have a range of intensity from 0 to 255 for each color component, which are combined to obtain a certain hue. They began with a simple color recognition algorithm that was first proposed using RGB values of bananas. The algorithm had three steps:

- Preprocessing
- Feature Extraction, using histograms
- Ripeness Classification

With the bananas, the algorithm proved to be 96 percent accurate. When preparing the dataset, the fruit images were downloaded from google. Segmentation was applied to images and the data from MATLAB was saved and organized in an excel sheet. There were two categories in the excel sheet: Mature and Immature. The training images were resized, the features were extracted, and placed in a training data database; the same is done to the testing images. Then the Artificial Neural Network is trained and tested. Machine learning

was used for recognition and counting purposes. The method used Back-propagation and tan sigmoid for classification. The method was 98.1 percent accurate.

An Agricultural Mobile Robot with Vision Based Perception for Mechanical Weed Control

Asgrand and Baerveldt [15] proposed an automated weed control robot that employs vision techniques. The robot is able to recognize crop rows and their structure. The rows determine which direction the robots move. The robots also can differentiate between crops and weed plants. The vision techniques help the robot determine where to remove weed vegetation contained within the crops.

Crop segmentation from images by morphology modeling in the CIE L a b color model

A new morphology modeling method was employed using supervised learning for crop image segmentation as a result of the research conducted by Bai et al.[16]. The purpose of the morphology is to handle the color of the crops and establish crop color models. The method used 56 test images and the performance was compared to other famous approaches.

Crop Disease Leaf Image Segmentation based on Genetic Algorithm and Maximum Entropy

Du and Zhang [17] used a segmentation method to identify crop diseases in leaf images. The method employed genetic algorithm and maximum entropy and thresholding. The results showed that the proposed method can select the threshold automatically in order to select the diseased portion of the crops. The three algorithms compared to the proposed method showed that they were inferior to the method proposed.

Environmentally adaptive segmentation algorithm for outdoor image segmentation

Tian and Slaughter [18] developed a supervised learning segmentation algorithm. They called it the environmentally adaptive segmentation algorithm (EASA). The purpose was for outdoor field plant detection. The algorithm could learn from different conditions and create a look-up table on the fly. It was able to adapt to most conditions that occur dur-

ing the daytime and soil types. EASA improved segmentation, when compared to static segmentation methods, by up to 54.3%.

Segmentation of green vegetation of crop canopy images based on mean shift and Fisher linear discriminant

Zheng et al. [19] proposed a mean shift and Fisher linear discriminant hybrid method that aimed to improve the performance of segmentation. It uses point-line-distance based strategies to weight the training data. The results showed that the distance of class means were enlarged by the point-line-distance strategy which increased the between-class scatter and decreased the within-class scatter.

A comparison of methods for estimating fractional green vegetation cover within a desert-to-upland transition zone in central New Mexico, USA

Xiao et al. [20] compared a set of methods for estimating the fracitonal green vegetation cover, also known as fractional green canopy cover (FGCC). The study was done over a 4000 square kilometer region of central New Mexico in the United States. The methods were tested using, what many calls, "true color" orthoimages. Orthoimagery is basically a remote sensed image that is corrected geometrically for lens distortion, tilt and topographical relief. DNVI based methods performed well for estimating the vegetation cover but had a lot of overestimation in thinly dispersed vegetated areas with bright soil. 3,4, and 5 end member spectral mixture (SMA) methods were tested also. The models all performed generally the same except for the SMA5 best captured the FGCC estimation for the rarer landscapes. Automatic Extration of Plots from Geo-Registered UAS Imagery of Crop Fields with Complex Planting Schemes Anthony Hearst [21] conducted research on how to extract plots from imagery based on map coordinates. A UAS was used to gather images of soybean fields that had 6 square meters with a complex planting scheme. His goal was to create an algorithm that can extract any collection of plots. At first the algorithm he wanted had to work as long as there was a uniform spatial grid of rows and ranges could be overlaid on the

image. The problem was that not all the plots were planted in a perfectly uniform spatial row. This error was common even though they were planted using a GPS guided planter. Because of this, an approach had to be developed that didn't depend on the complexity of the planting scheme. The extraction of plots based on map coordinates makes it possible for repeatedly and automatically extracting plots from imagery of a crop field during different times of the growing season. The method assumes the map coordinates of the corners of the plot have been measured, and the image coordinate have been assigned to map coordinates. These coordinates are used to define the polygon that will be used to extract the plot. Sixteen artificial targets were targets were setup before flights. Up to six of the targets were used as Ground Control Points (GCP) for geo-registration which, as a result, provided 175 images with a broad range of accuracies. The plot extraction accuracy was based on the percentage of plot area extracted. The plot extraction accuracy was at least 70% with a mean plot extraction accuracy of 92% when 4 GCPs were used.

Using Pattern Recognition to Automatically Crop Framed Art

Indianapolis Museum of Art (IMA) owned thousands of high resolution images of art and about 5000 of them were uncropped. This means that the images had frames, swatches and many other diverse backgrounds. Because of this the images couldn't be shown to users so an approach to crop these images automatically was developed. Kavlerov's main goal was to find the rectangle that best fit the artwork in the photos[22]. The rectangle discovered had to be small enough to only contain the art, none of the background or frames. Kavalerov didn't want to overcrop the images because sometimes the edges are important to keep. Commands in Python was used using OpenCV. Gaussian Blur was used, which is a technique that removes noise by averaging pixels and the ones surrounding it. Other methods used for extracting the paintings from their frames and backgrounds consisted of Sobel Filter, Median Blur Filter, Canny Edge Detector, Hough Transform, Contour Search. The method used had a 85% success rating.

Implementing Rectangle Detection using Windowed Hough Transform Akhil Singh

Singh [23] used Hough Transform to recognize rectangles in images using preprocessing methods and performing widowed applications to increase the speed of the algorithm. Hough transform is a technique used for feature extraction and image processing. It finds geometrical shapes in images using voting. It assumes that any line on an X-Y plan can be described using the below equation:

$$p = x\cos\theta + y\sin\theta \tag{2.1}$$

p is representative of the normal distance. θ is representative of the normal angle of a straight line.

A sliding Hough Transform window detects rectangles using the peaks of the transforms extracted and other predefined geometric conditions. If the rectangle in the image is larger than the sliding window the window will be unsuccessful at detecting the rectangle. Therefore, for the method to work the window has to be large enough to detect all rectangles. It is important to make sure the window isn't too large so that it doesn't contain parts of other structures. The algorithm was effective when it came to identifying rectangles in images. When the images are overlapped the method doesn't work so they had to use an alternative method

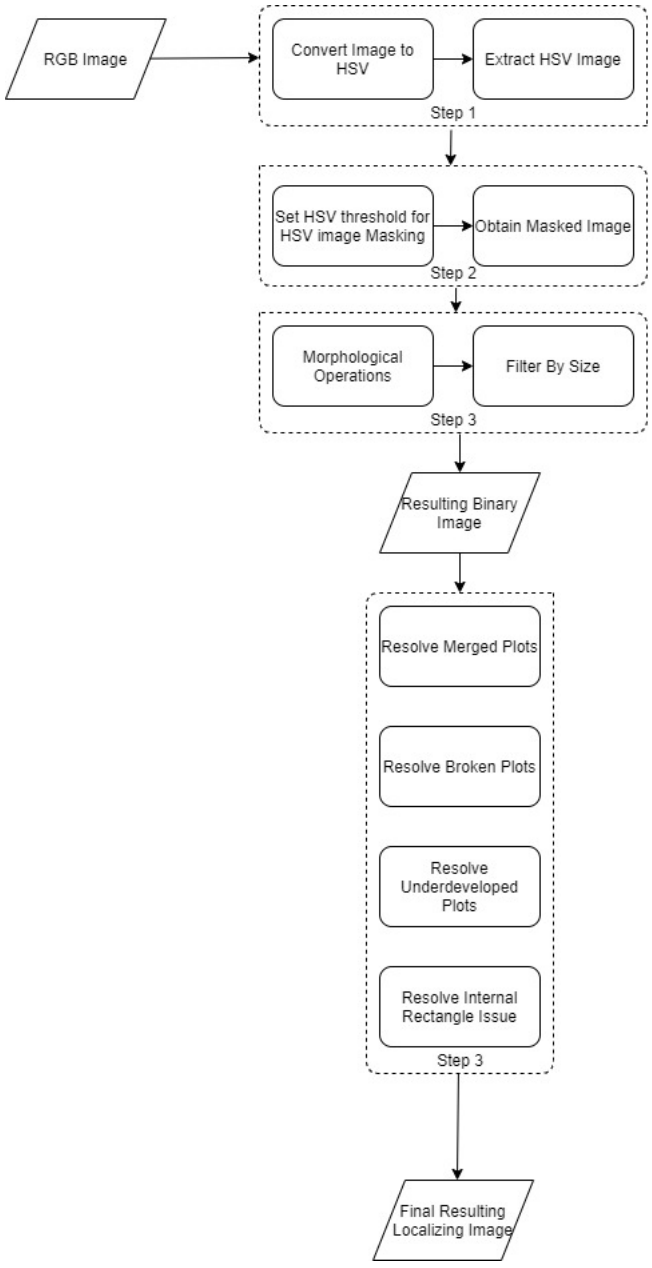

Figure 3.1: Illustration of our algorithm.

Our system steps are shown using figure 3.1. The images used have varying resolutions. The largest have resolutions of 4000 x 3000 pixels in a Joint Photographic Experts Group (JPEG) format. They were collected in Beeville, Texas weekly via Unmanned Aerial Systems (UAS) by a team of Texas A&M University Agrilife researchers. Figure 3.2 is an example of the images being used for the purpose of the research.

Figure 3.2: An example of one of the sample images used during the development of our algorithm.

3.1 Convert to Binary using HSV Color Space, and Preprocess Images using Morphology

Not all of the plots of vegetation in the images being evaluated are green; sometimes there are plots of red vegetation. These red plots are to be used as boundary lines which means they're not being evaluated for any particular studies. With that being said, during the construction of the our algorithm it was important to ensure the method used to extract

individual plot features must distinguish between the green and non green pixels in an image. An example of the different colored plots are shown in Figure 3.3.

Figure 3.3: Visual representation of the different colored vegetation.

3.1.1 Masking

The first step in the proposed method is to mask the image using OpenCV. In order for that to happen the image must be converted to HSV color space from RGB color space. HSV stands for Hue, Saturation, Value and RGB stands for Red, Green, Blue. The HSV channels are described below:

- Hue - Description of pure color
- Saturation - How much pure color is diluted with white light
- Value - The brightness of the color

Instead of making calculations for evaluating the proportion of Red, Green, and Blue a pixel contains it is much faster to just evaluate the Hue value of each pixel in the original image when making decisions on which pixels in the masked image should be green and which should be black; this makes the HSV threshold method more attractive. The color hues range from 0 to 360. These values represent the place where each color lies on the hue spectrum. In this image we focus on finding the green pixels so a max and minimum value for the hue was set accordingly. See Figure 3.4 for the resulting HSV Image.

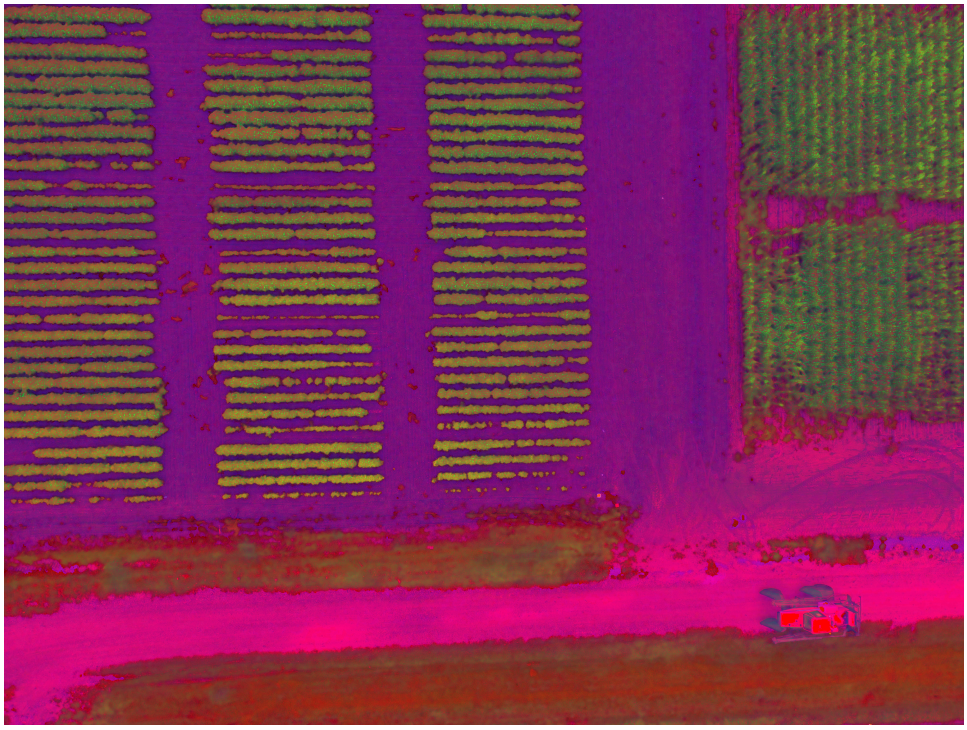

Figure 3.4: Original Image Converted To HSV Color Space.

Any pixel hue that isn't within the range set previously is considered to be pixels not belonging to any plots of green vegetation. These non-green vegetation pixels were turned to a black pixel for the masking and the others remained green. Figure 3.5 shows the result of the masking method used.

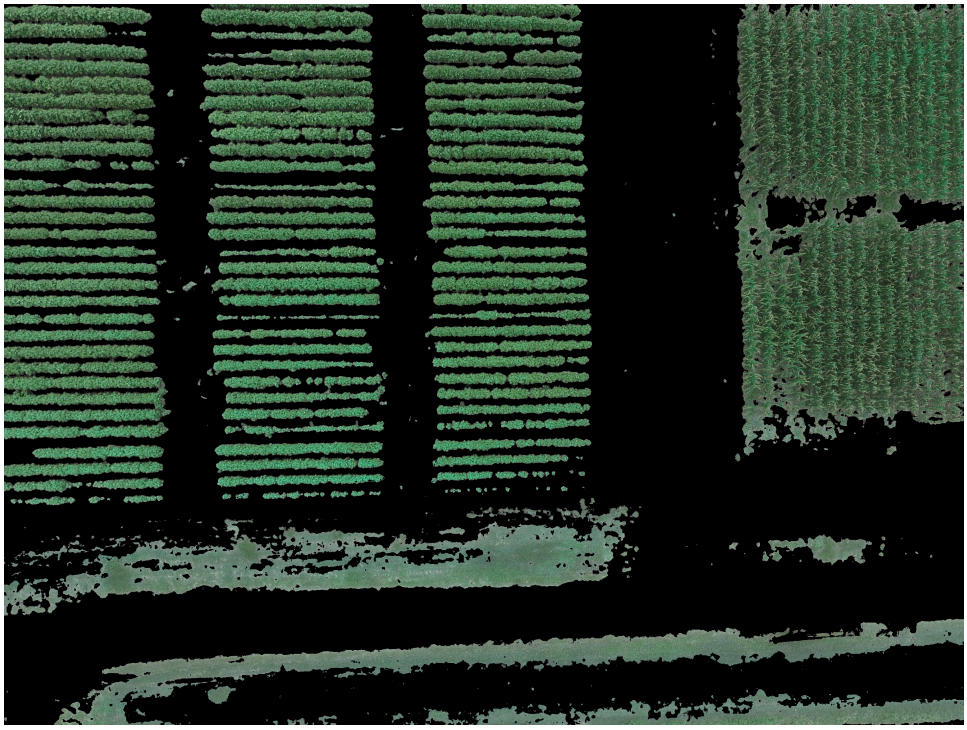

Figure 3.5: Image Masked using the HSV color space.

3.1.2 Conversion

The resulting masked image created is then converted to a binary image using OpenCV's conversion algorithm. Binary images are more computationally inexpensive than working with 8 or 24 bit images. The white pixels in figure 3.4 correspond to the green pixels in the masked image.

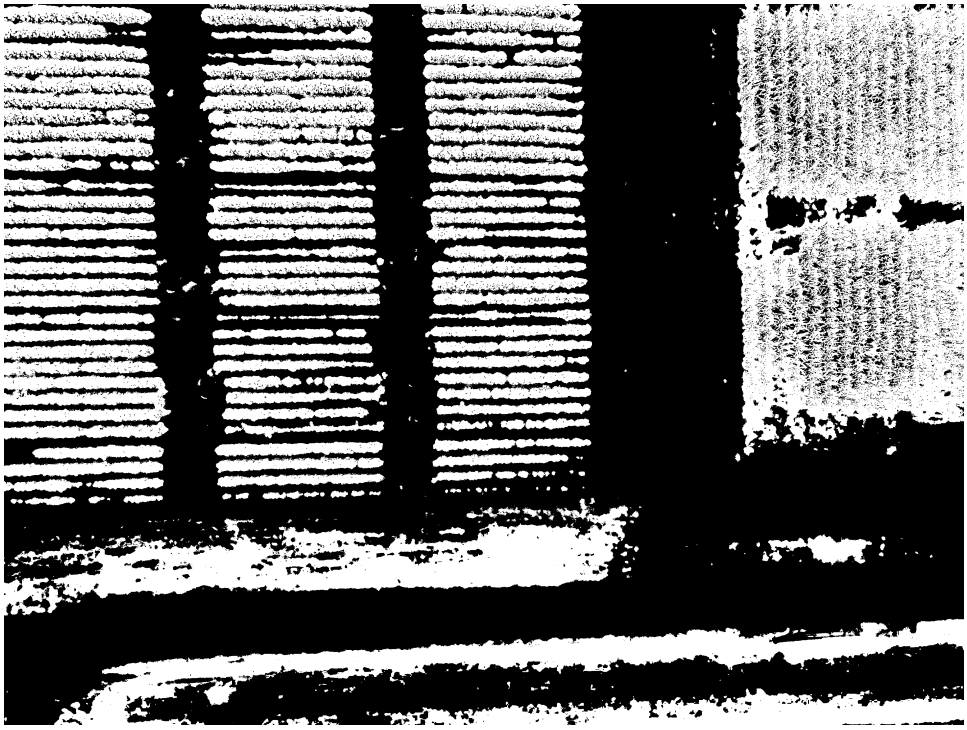

Figure 3.6: Binary image after being converted from masked image.

3.1.3 Morphology and Filtering using ConnectedComponents

The binary image that results from the previous step shows groups of white pixels representing the green vegetation and a black background. This binary image will be the main image worked with when detecting the plots in the image. The groups of white pixels will need to be evaluated. Any groups that are too big or too small will be turned into black pixels. This will be done by setting a minimum and maximum group size. These maximum and minimum sizes will be determined by figuring out the size of all the groups. If there are any outlying group sizes they are taken into account when deciding which numbers are chosen as the maximum and minimum group sizes. The results of the operations along with the filtering using ConnectedComponents are shown in figure 3.7.

Figure 3.7: Result from filtering by size.

Because not every group of pixels will be perfectly grouped certain morphological operations must be applied. What's meant by perfectly grouped is not every group will be one color all the way through. The groups of white pixels may have a few black pixels present throughout them. Also, the groups may be linked together because of irregular growth of the crops or neighboring vegetation. These two things give reason to use morphological transformations. The morphological transformations are preprocessing operations used to alter objects in binary images as a means of cleaning the data for evaluation. The transformations usually use two inputs, the image and a kernel. The kernel decides the nature of the operation. There are four main kinds of morphological transformations.

- Erosion
- Dilation

- Opening
- Closing

Erosion is an operation that erodes away boundaries of objects. The kernel is set and it slides through the image pixels similar to the kernel in a 2D convolution. The kernel will look at each pixel in the original image, and will determine the pixel to be a 1 if all the other pixels under the kernel are 1. Otherwise it will be considered a 0. Erosion is effective at disconnecting objects one may not want to be connected. It can also completely erase an object if applied in multiple iterations. An illustration of the effects of erosion are shown in figure 3.8.

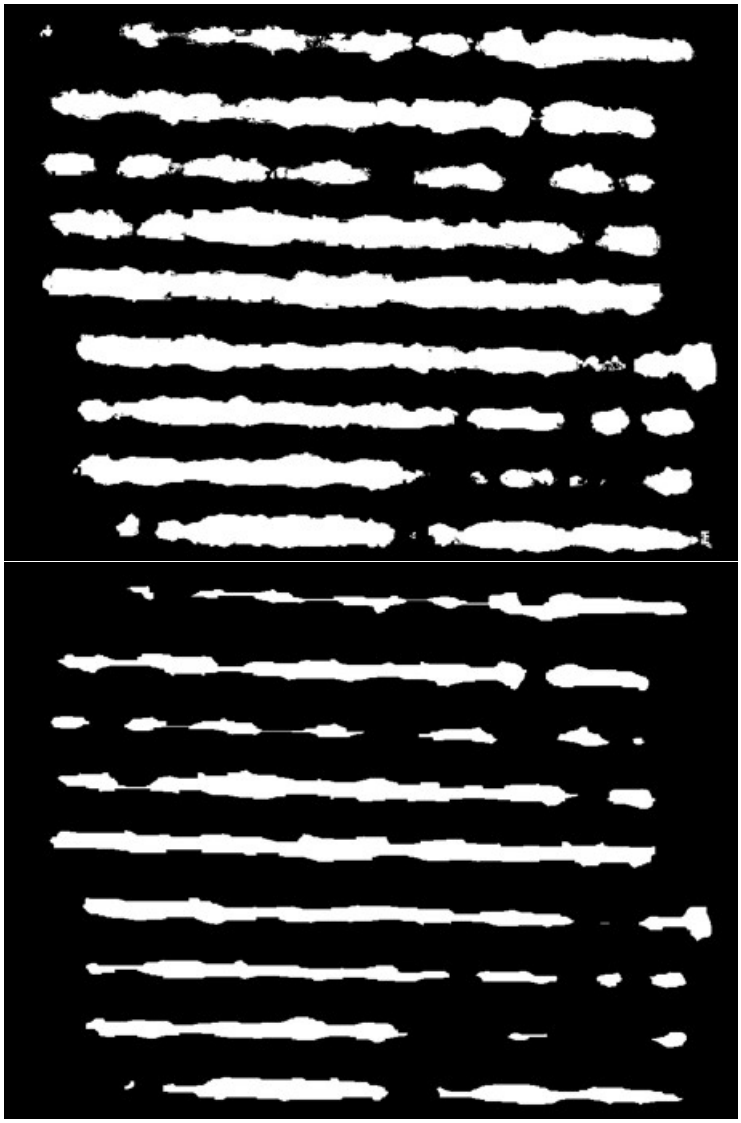

Figure 3.8: The before and after images of erosion being applied to vegetation plots.

Dilation is the opposite of erosion. It expands the object in the image. A pixel will be determined a 1 if at least one pixel under the kernel is 1. Dilation is effective in connecting objects whenever they're needed. It can also increase the size of small objects for that need to be made more visible. See figure 3.9 for an illustration of how dilation effects plots of vegetation.

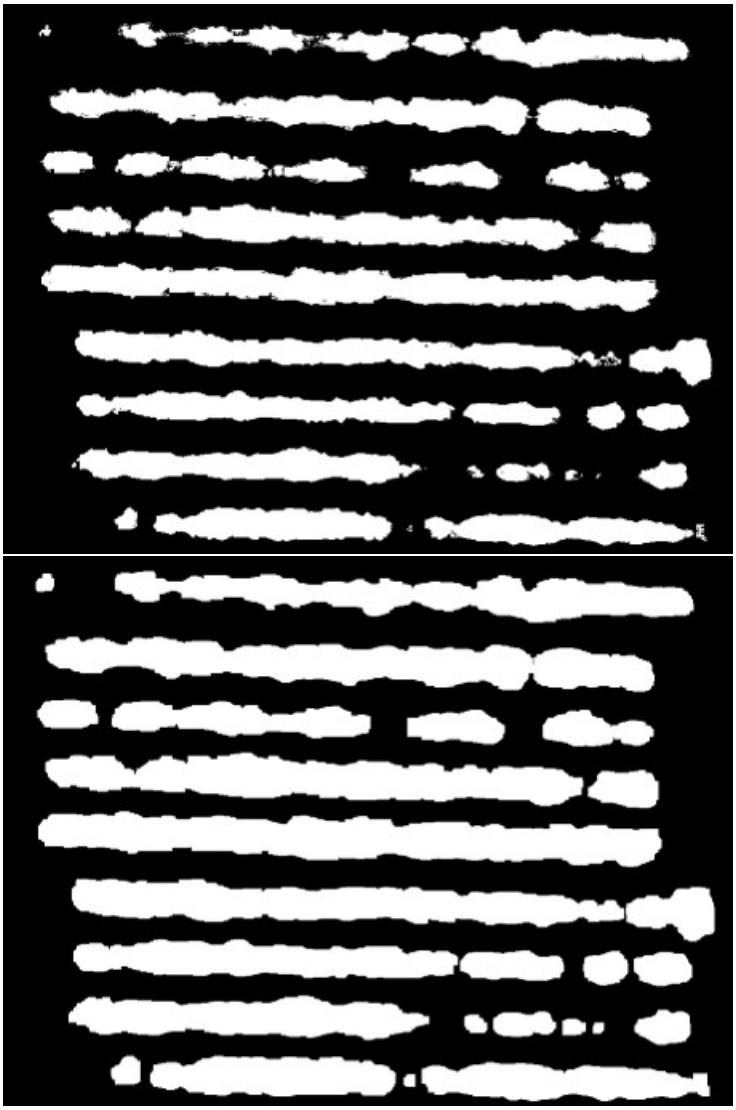

Figure 3.9: The before and after images of dilation being applied to vegetation plots.

Opening and closing are combinations of erosion and dilation. Opening is erosion followed by dilation. This process is effective in removing noise. It can be good at disconnecting objects, like erosion, but allowing the size of the objects to remain the same. See figure 3.10 for an illustration of opening's effect on vegetation the plots.

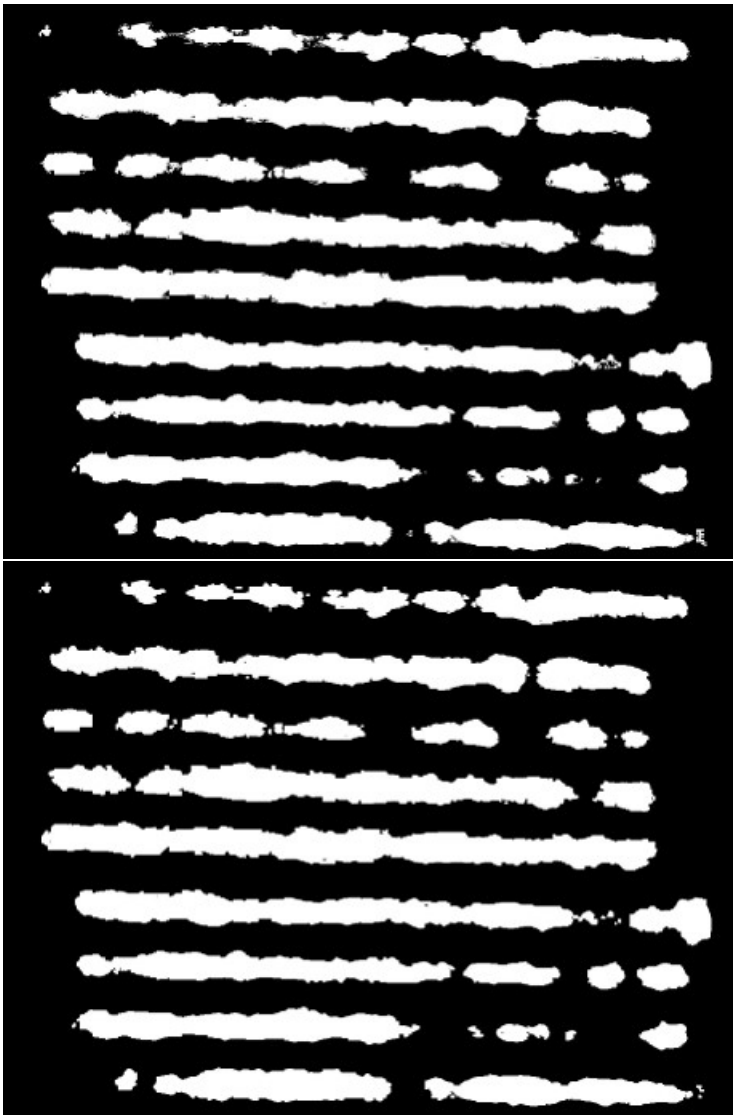

Figure 3.10: The before and after images of opening being applied to vegetation plots.

Closing is dilation followed by erosion. It is effective in removing black spots inside of white regions/objects. Also, it can be effective in removing black spots inside of white regions. See figure 3.11 for an illustration of closing's effects.

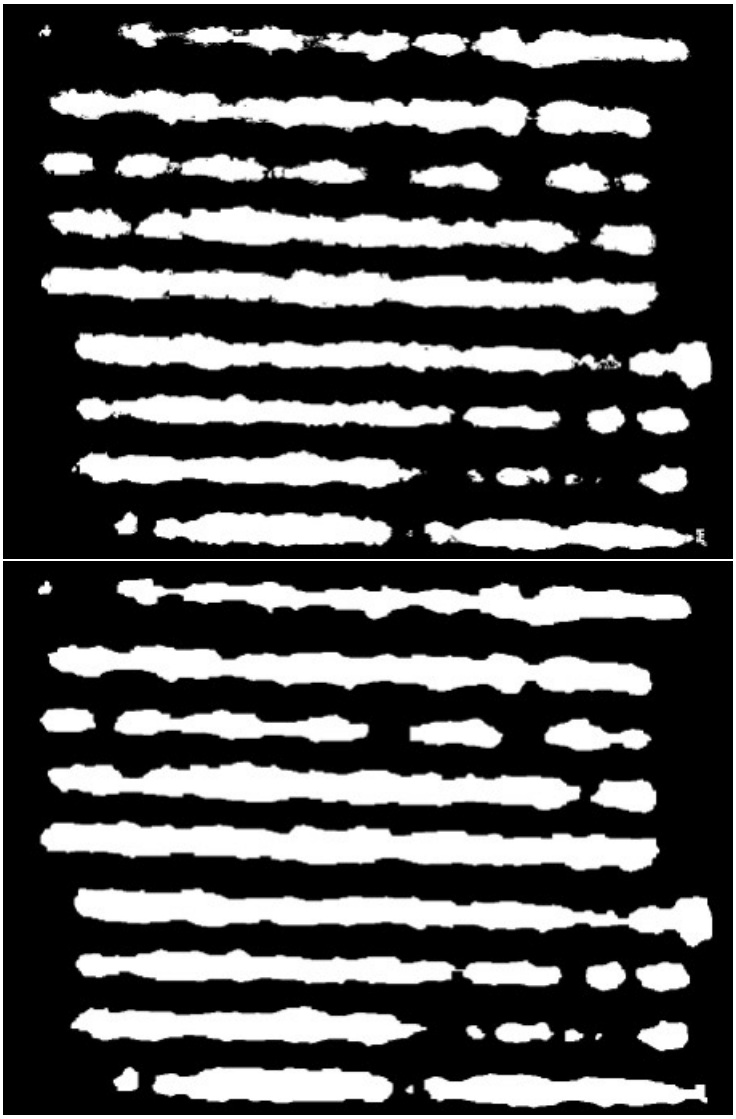

Figure 3.11: The before and after images of closing being applied to vegetation plots.

For the images used for the purpose of the research closing was applied in an attempt to remove holes (black pixels within a group of white pixels) in the white pixels. Next Erosion was applied because the next step was to use dilation to expand the objects. If the objects were expanded before applying erosion then the dilation may cause overlap between objects.

3.2 Contour Detection and Connected Component Pixel Counting

The contours of each object in the image were found. A contour is a curve joining all the continuous points along the boundary of an object that have the same color. This is most effective when used with binary images. The result from contour detection can be used for shape analysis and object detection. For the purpose of this research these contours will be used with a method called connected components. connected components is a function that counts how many white pixels, which represent objects, in each contour are connected. Connected Components has two ways of searching for connected neighboring pixels:

- 8 neighbor
- 4 neighbor

With both 4 and 8 connectivity the center pixel is being evaluated. The difference lies in the direction that the function searches for connected pixels. With 4 neighbor, the function looks to one pixel below and above and also one pixel to the right and left of the center pixel. If either of those pixels are white it is considered a part of that object. With 8 neighbor connectivity the function looks in every direction the 4 neighbor looks but also looks at the pixels in all four directions diagonally to the center pixel.

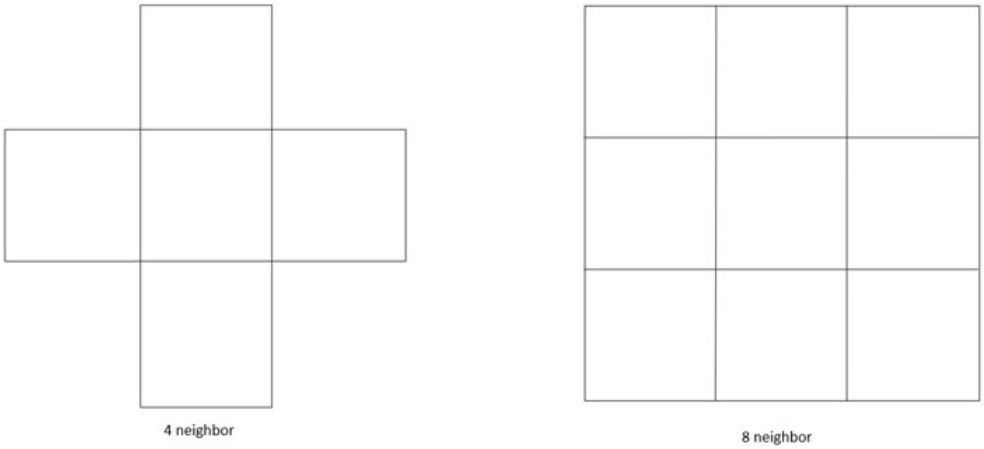

Figure 3.12: The difference between 4 and 8 neighbor connectivity evaluation.

These contours will then have a polygon drawn around them to localize each plot of cotton vegetation is located. The final step would be to localize each plot, but before that can happen a few issues that arise from abnormal growth patterns must be resolved.

3.3 Specific Challenges Caused by Abnormal Growth Patterns

Because of the irregular growth patterns of some plots a few complex situations arise when evaluating each individual plot of vegetation. Every plot is planted with the same amount of seeds. Therefore if every seed blooms every canopy will generally look the same assuming they all grow at the same rate. The problem is that not every seed will bloom or sometimes the plots grow in unexpected patterns, so a large number of vegetation plots will look different from the others. This is due to different treatment, access to water, insects/animals, pesticides, etc. Figure 3.13 illustrates a couple of the complex situations from irregular growth patterns. The challenging issues to be resolved include:

- Broken plots
- Connected plots
- Rogue objects
- Underdeveloped plots
- Contours created from connected greenery
- Smaller polygon drawn within larger localizing polygon

3.3.1 Broken Plots

Because of the fact that not all seeds grow just because they are planted, the crops may not grow all the way across, and some sections of the crop may be more grown than others. During localization there could be multiple polygons drawn to indicate the location of a single plot of vegetation. This is a problem because even though the polygons are drawn around the correct plots, there needs to be only one polygon drawn around individual plots at once. A method should be implemented to deal with this complex situation. Figure 3.14

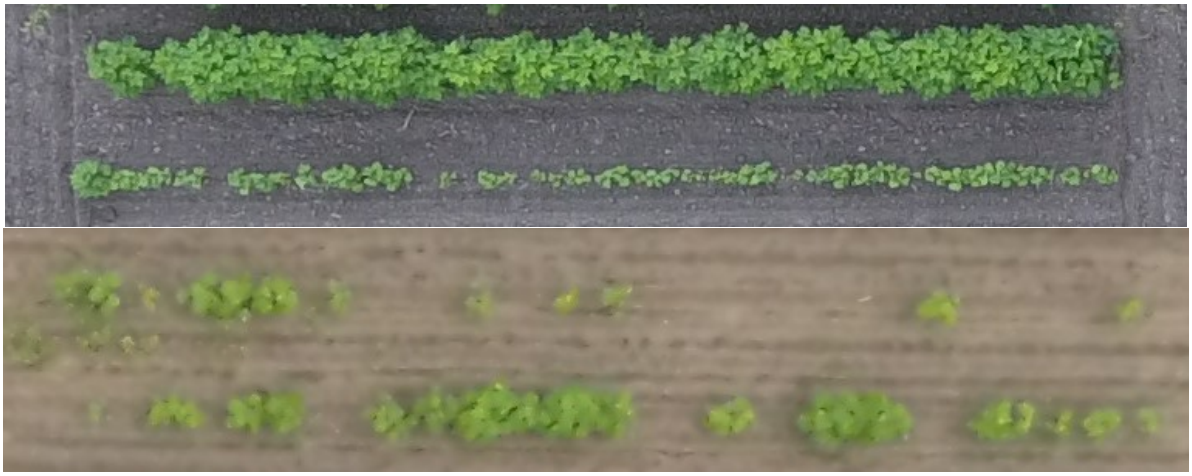

Figure 3.13: Examples of Challenging situations. The top image is a image of a plot of vegetation with a normal growth pattern next to a underdeveloped plot. The bottom image is an example of two severely underdeveloped plots of vegetation.

Figure 3.14: An example of a broken plot (Bottom) of vegetation next to a plot with a normal growth pattern (Top).

is an example of a broken plot next to a plot of vegetation with a normal/expected growth pattern.

3.3.2 Connected Plots

Some plots grow in different directions than originally intended. Because of this, the plots may appear to be connected. This is difficult to deal with because the erosion and dilation may not get rid of the connections. The issue may cause the polygons to be drawn too large. Instead of the polygons being placed around just one of the plots, it will place them around all the connected plots and may even create a misaligned localization polygon. This is shown in figure 3.15. A method to divide the connected plots must be developed to

resolve this issue.

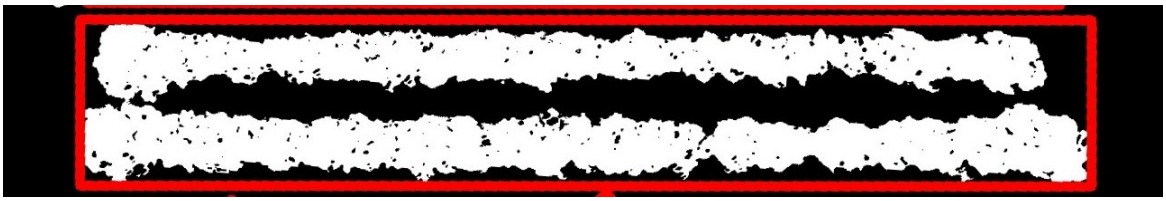

Figure 3.15: Binary image of polygon drawn for connected plots.

3.3.3 Rogue Objects/Vegetation

Because a filtering method based on the connected components is used to get rid of objects that are too large or small, it may not get rid of every object that isn't necessary. The connected components in these objects may fall into the range indicated during the filtering process. A large amount of the rogue objects are caused by the small amount of green pieces of vegetation growing in the red plots of vegetation, as shown in figure 3.16 This can cause the localizing polygons to be drawn around these objects. This is one of the more difficult things to deal with depending on the size of the objects. A method was developed to decide whether or not to include these rogue objects in the localization process or not.

3.3.4 Contours Created from Connected Greenery

Normally, the contours detected are connected white pixels but sometimes they are a collection of black pixels that appear to be an object created by the connected white pixels;

Figure 3.16: Rogue objects created by green vegetation within red plots of vegetation.

see figure 3.17. When this happens, the polygon drawn may be inside another polygon. A method was developed to remove polygons from within others. This issue solved some of the rogue objects issues along with the issue being discussed in this section.

Figure 3.17: Binary image of black internal contours found resulting from objects growing together.

3.3.5 Smaller Polygon Drawn within Larger Localizing Polygon

Sometimes the contours detected appear inside the polygon of another contour. This is usually the case for the smaller/disconnected contours being found in the same general area as a few connected object, as shown in figure 3.18. This issue is somewhat of a hybrid of the previous two issues. A solution was developed to resolve the internal

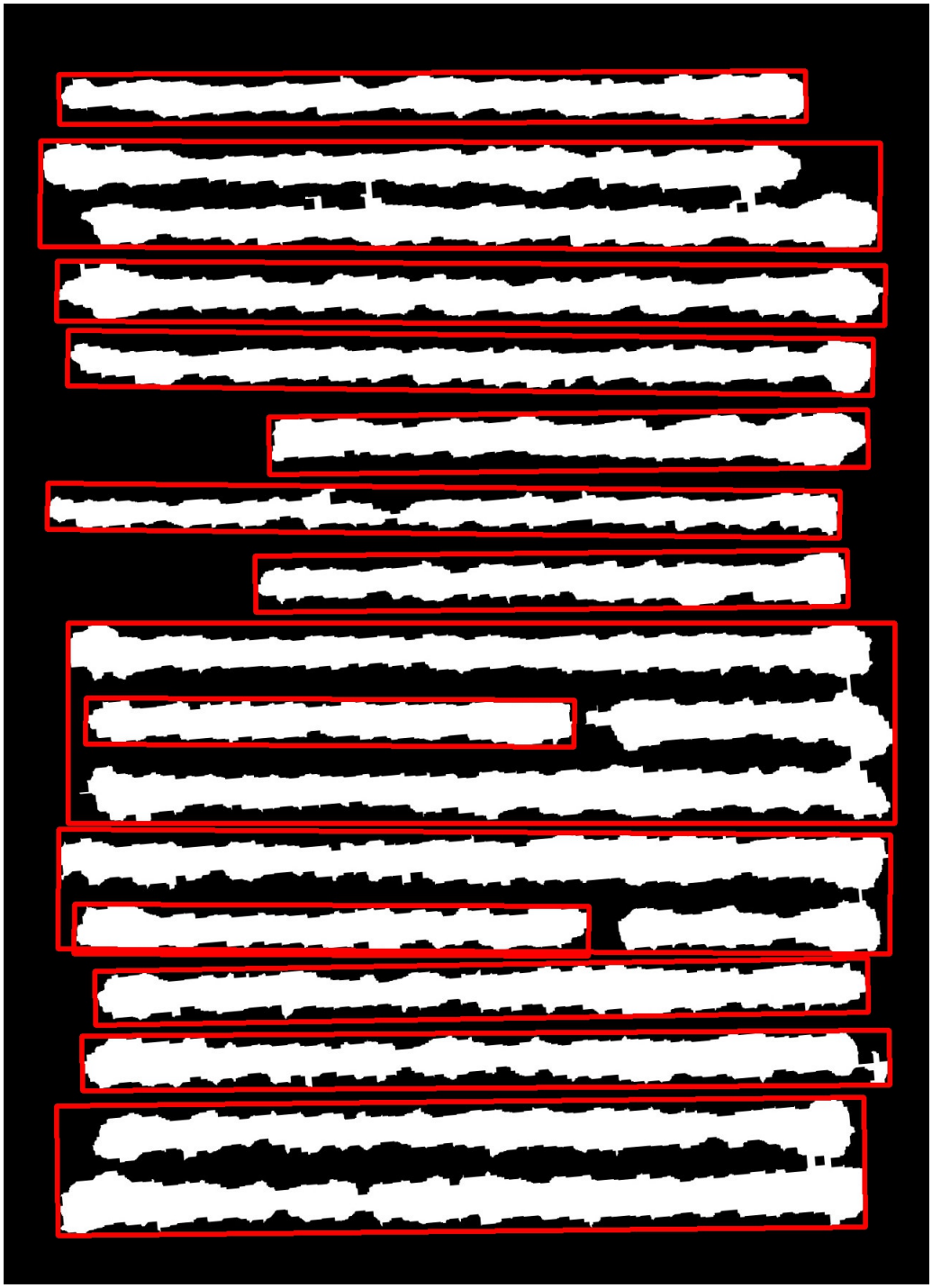

Figure 3.18: Binary image of polygons drawn for broken plots inside larger polygons.

3.4 Developed Solutions to Specific Challenges

The previously mentioned challenging issues to be resolved required subsystems to be developed for each one. This section explains the solutions in detail.

3.4.1 Solution for Connected Plots: Divide Plots

The plots sometimes appear to be connected. The connected plots can cause a problem for the algorithm because if the plots connected are large, and weren't filtered out during the filtering by size process. The algorithm will draw a polygon around the contours and the polygon will be one polygon surrounding multiple plots. To resolve this the issue these polygons will need to be split up to make the large polygon into the necessary amount of polygons needed for the number of plots of vegetation it contains.

$N \leftarrow 0$
$Total \leftarrow 0$
for C in Contour do
    $Total \leftarrow$Total+C.width
    N $\leftarrow N + 1$
$StandardWidth \leftarrow Total/N$

Figure 3.19: Algorithm to find out what the standard width of a polygon should be.

The equation in figure 3.19 was employed to find out how many plots were found inside of the box to be drawn. There was a standard width of polygons used for evaluation, which was found using through averaging the widths of the polygons drawn. The width of the box was divided by the standard width established and the integer result was the amount of polygons the large polygon would be divided into. The division of the large polygon was carried out by displayed in the algorithm in Figure 3.20. The Algorithm is regulated by an integer 'i' in a loop.

Result: Splitting the localizing polygons.
$X \leftarrow StandardWidth$
for C in Contour do
  $Y \leftarrow C.width$
  $i \leftarrow 1$
  $Times \leftarrow Y/X$ ▷ Integer representing number of times larger than the Standard
  if Times is greater than 1 then
    while i is less than Times do
      $G \leftarrow (X_1 + ((i/Times)$ x $(X_2$ - $X_1)))$ ▷ Finding the X coordinate of one end of the dividing line
      $Z \leftarrow (Y_1 + ((i/Times)$ x $(Y_2$ - $Y_1)))$ ▷ Finding the Y coordinate of one end of the dividing line
      $G_1 \leftarrow (X_4 + ((i/Times)$ x $(X_3$ - $X_4)))$▷ Finding the X coordinate of the other end of the dividing line
      $Z_1 \leftarrow (Y_4 + ((i/Times)$ x $(Y_3$ - $Y_4)))$ ▷ Finding the Y coordinate of the other end of the dividing line
      $i \leftarrow i + 1$
      Draw line from coordinate (G, Z) to coordinate (G_1, Z_1)

Figure 3.20: Developed algorithm for splitting connected plots.

Let x_1,y_1,x_2,y_2,x_3,y_3, x_4, y_4 be equal to the first, second, third and fourth polygon x and y corner coordinates. The lines drawn will split the polygon into the proper amount of polygons and give each cotton plot its own rectangular polygon. The coordinate for one end of the line will be at (G, Z) and the splitting line will be drawn to (G_1, Z_1). The results are shown in figure 3.21.

Figure 3.21: Image of large localizing polygon being divided into the proper number of rectangles.

3.4.2 Solution for Contours created from Connected Greenery

If there happens to be a contour within a contour, there can be an assumption made that it is mainly composed of black connected components. This assumption can be used as a means to decide whether or not to draw a rectangle. To make the decision a averaging method can be used. When drawing polygons around contours there must be points found that will be the corners for the polygons. In this case the polygons are rectangles. The four points of the rectangles can be used as reference points to scan the pixel values inside the rectangles and find the percentage of black pixels inside the rectangles. If the rectangles percentage is greater than the threshold value set, then the rectangle will not be drawn.

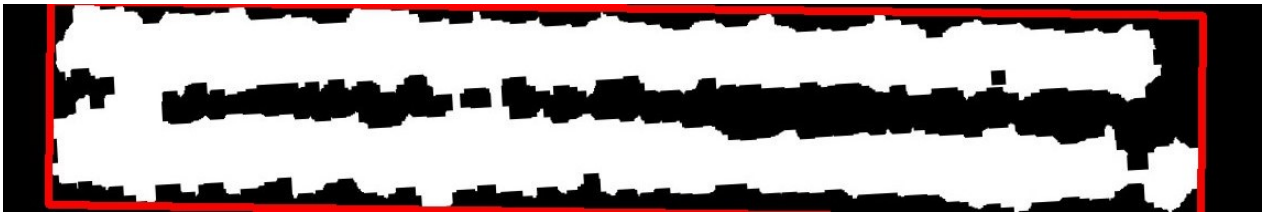

Figure 3.22: Connected Greenery Solved.

Another solution is to figure out if any of the points of any polygon lies within the points of another polygon. This is done by comparing the x and y coordinates of one polygon with the x and y coordinates of each other polygon in the image. If this proves to be the case, the polygon found to have coordinates within any other polygon will be deleted before the contours are drawn. First a list, called *Removed*, is created to hold all the polygons detected to have points inside of other polygons. This assumes a separate list of polygons to be drawn, called *Polygons*, has already been created. Once the polygon point has been detected to have points inside of another localizing polygon then that polygon is inserted into the *Removed* list. Once all of the polygons that have points inside others have been

identified a conditional statement is created to removed the polygons inside of *Removed* from the *Polygons* list. The algorithm developed is displayed in figure 3.23.

```
G ← False
Removed is created
for P in Polygons do
    for R in range(4) do
                                                    ▷ Cycle through 4 points

        point ← Point(P[R])   ▷ The Point keyword converts the coordinate to the proper
        format.

        for X in Polygons do
                                          ▷ Cycle through the list of polygons.
            polygon ←Polygons[X]
            if polygon.contains(point) then
                Removed.append(P) G ← Truebreak                ▷ Stop third loop.
        if G == True then
            G ← Falsebreak                                     ▷ Break second loop.
updatedPolygons ←Polygons.copy()        ▷ Copy Polygons into updatedPolygons list
for i in range(len(Removed)) do
    updatedPolygons ← updatedPolygon element not present in Removed
    _
```

Figure 3.23: Algorithm created to remove polygons who have points that have been detected to be inside others.

3.4.3 Solution to Polygon Drawn within Larger Localizing Polygon

To solve the polygon inside larger polygon issue, two loops were created. The first loop cycled through a list of polygons to be drawn and the other cycled though an array of polygon coordinates that correspond to the polygons being cycled through in the first loop. The coordinates were evaluated using Python's Point and Polygon elements imported from the shapely.geometry library. These elements help by detecting if a specific Point is contained within a Polygon. If the Point is found to reside inside of the boudaries of another

polygon, the polygon that corresponds to that point will be removed from the list of polygons to be drawn. When the initial loop is finished cycling through the list of polygons to be drawn, the list is then cycled through a final time to draw the remaining coordinates polygon coordinates. These remaining coordinates are the polygons that weren't found to have points within other polygons.

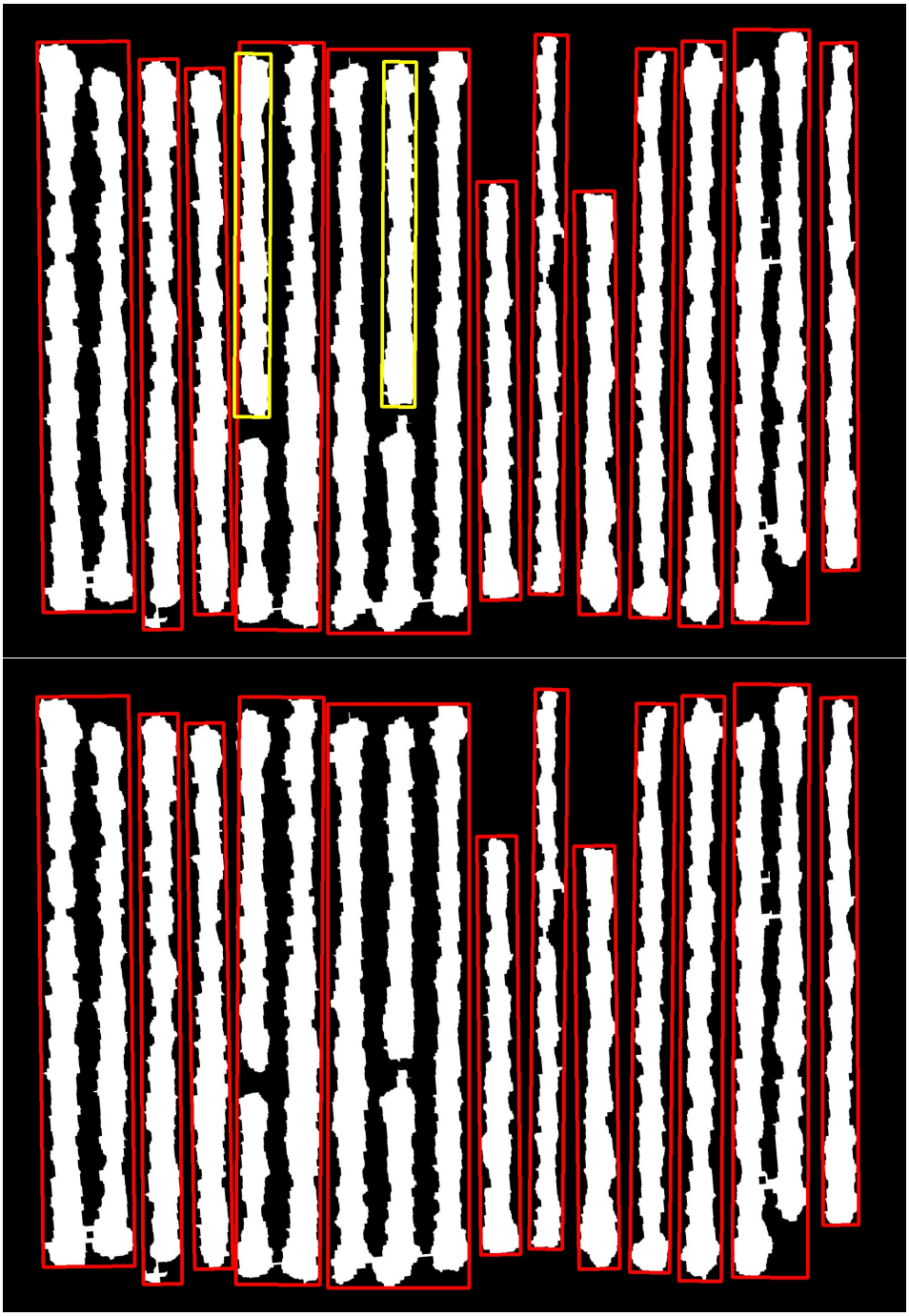

Figure 3.24: Binary image of result when removing incorrectly drawn localizing polygon within a larger one. The top image shows the polygons to be removed from the image, they are colored yellow. The bottom image shows where the polygons have been removed.

3.5 Localization

The final step of the algorithm is to localize the objects detected by drawing a rectangle around the corresponding pixels in the original image. Because of imperfections from the previous steps, there was a need for another filtering process to be applied in order to determine which objects to draw rectangles around. The preprocessing methods were effective in removing most noise but in order to accurately indicate location of the correct plots there had to be a threshold set to determine which polygons were drawn. The filter to be used must use the area/size of the contours detected to determine which to localize. Using OpenCV's contours the area of the polygons to be drawn is found and if the area is within the threshold set, then the polygon will be drawn; otherwise the algorithm continues without drawing.

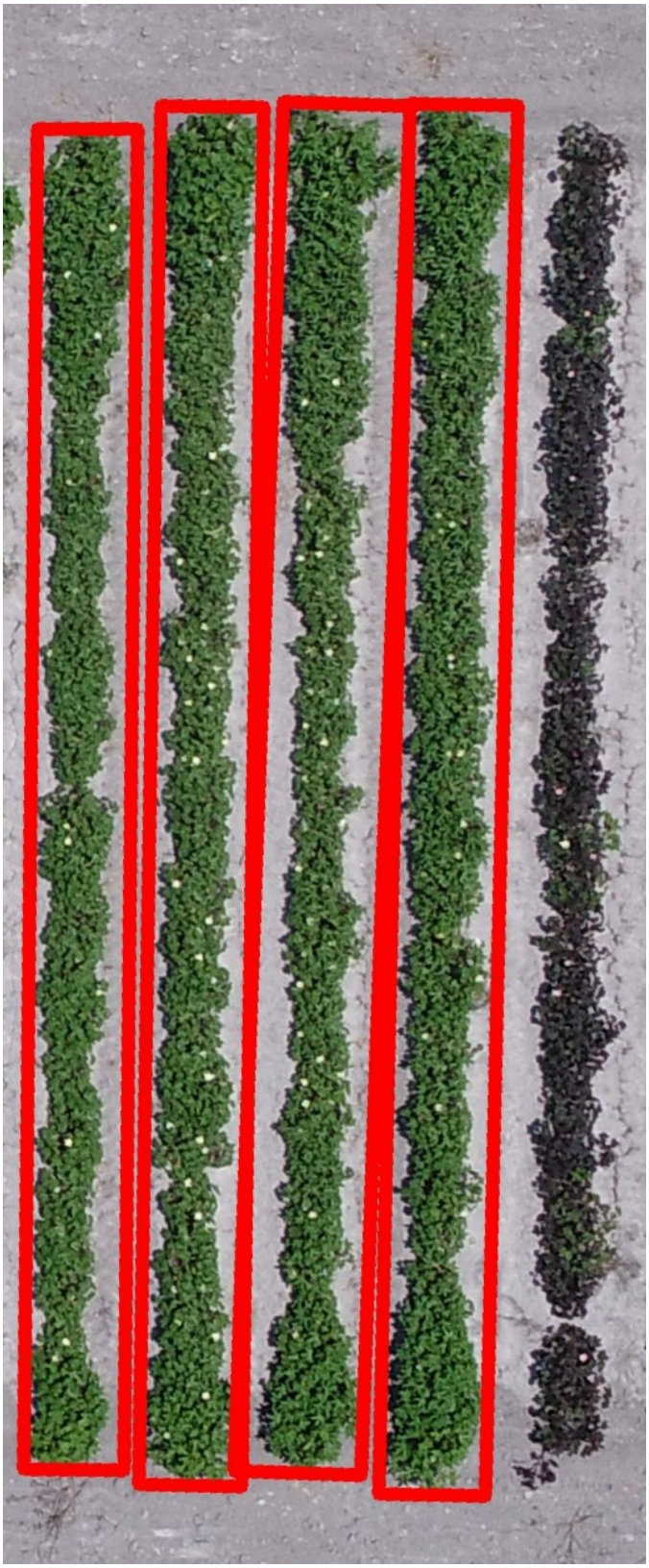

Figure 3.25: Image of four individual plots of vegetation being localized.

4. Analysis

After much development and testing the finalized algorithm was then applied to the 12 sample images. Using the algorithm's output it then had to be evaluated based on a set of criteria:

- Precision
- Recall
- Accuracy

This section is where those metrics are calculated for our algorithm and explained in detail. Also, a discussion regarding why the HSV color model was chosen over the RGB model.

4.1 Evaluation Criteria

In order to determine the overall performance of our algorithm an appropriate performance metric must be employed. In the paper by Hamuda [2] a ratio between area of detected bounding box and area of annotated bounding box was used to determine accuracy. Also, a measurement of precision and recall was used to evaluate the classification performance of their method. precision and recall formulas are listed below:

$$Precision = \frac{T_p}{T_p + F_p}$$
$$Recall = \frac{T_p}{T_p + F_n}$$

T_p refers to the true positive. True positive is equal to the number of correctly detected objects. F_p refers to false positives, which is the number of incorrect detections. F_n refers to

false negatives. A false negative is a failed detection of an object. The precision and recall equations employed by Hamuda will be used to evaluate the classification performance. The equation for accuracy is listed below:

$$Accuracy = \frac{T_p}{T_p + F_p + F_n}$$

Our method method was evaluated based on how accurate it is at localization of both simple and complex plots of vegetation and if the research goals were met. The measurement of accuracy is calculated as a ratio of plots correctly detected and total number of plots. For a plot of vegetation to be considered correctly detected the algorithm must draw a polygon around the whole plot or at least half of the plot. If the polygon was drawn around less than half of the target plot of vegetation, then it was counted as incorrectly detected.

4.2 Evaluation

There were 12 sample images given for the purpose of the research, each had a differing amount of plots. In total the amount of plots in the images came out to 435 plots. Our system determined the the plots at an accuracy of 94.2%, a precision of 100% and a recall of 94.2%. The fact that there were no incorrectly detected plots (False Positives) was the reason why the precision came to 100%. The results are displayed in table 4.1.

Image	1	2	3	4	5	6	7	8	9	10	11	12	Accuracy	Precision	Recall
System	54	19	22	26	22	19	9	26	140	22	20	31	94.2%	100%	94.2%
Actual Amount	57	19	23	27	30	19	9	26	141	26	26	32	100%	100%	100%

Table 4.1: Table displaying the results of our algorithm by image.

4.3 Canopeo RGB thresholding vs. OpenCV HSV thresholding Comparison

The Canopeo algorithm was the original binary creation method being used for the purpose of the research. The problem was the fact that it took too long for each image to be converted if the image was large, so OpenCV's HSV thresholding method was used instead. Canopeo and OpenCV HSV thresholding were both highly accurate in creating a binary image from the green and non-green pixels. The main advantage the OpenCV method had over Canopeo was the time it takes to complete. The Canopeo method takes a much longer time to complete because of the equations needed to be applied to each individual pixel evaluation of its Red, Green, and Blue values. The computation time depends on how large the image is. The larger the image, the longer the Canopeo method takes to complete. For one 3000x4000 pixel image the Canopeo algorithm took 120.86 seconds to complete, see table 4.2. The speed of the OpenCV HSV thresholding was significantly faster which proved that it would have the advantage over Canopeo if it was to be used on a large number of images at once no matter the size. The time it took the OpenCV HSV thesholding method was .63 seconds. See figure 4.2 for the results of both the HSV binary image and the Canopeo binary image.

	Canopeo Algorithm	HSV Binarization Algorithm
Time(sec)	120.824	.636

Table 4.2: Table displaying Canopeo vs. OpenCV HSV Binarization.

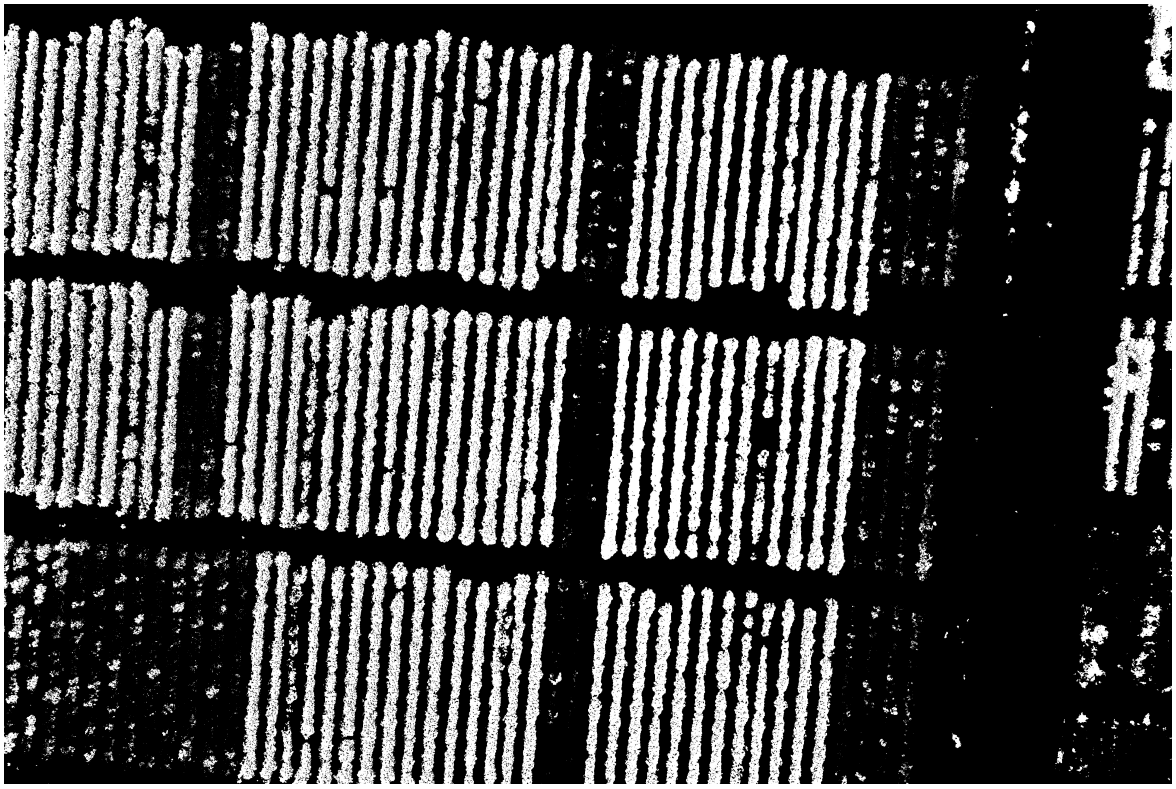

Figure 4.1: HSV thresholding binary output

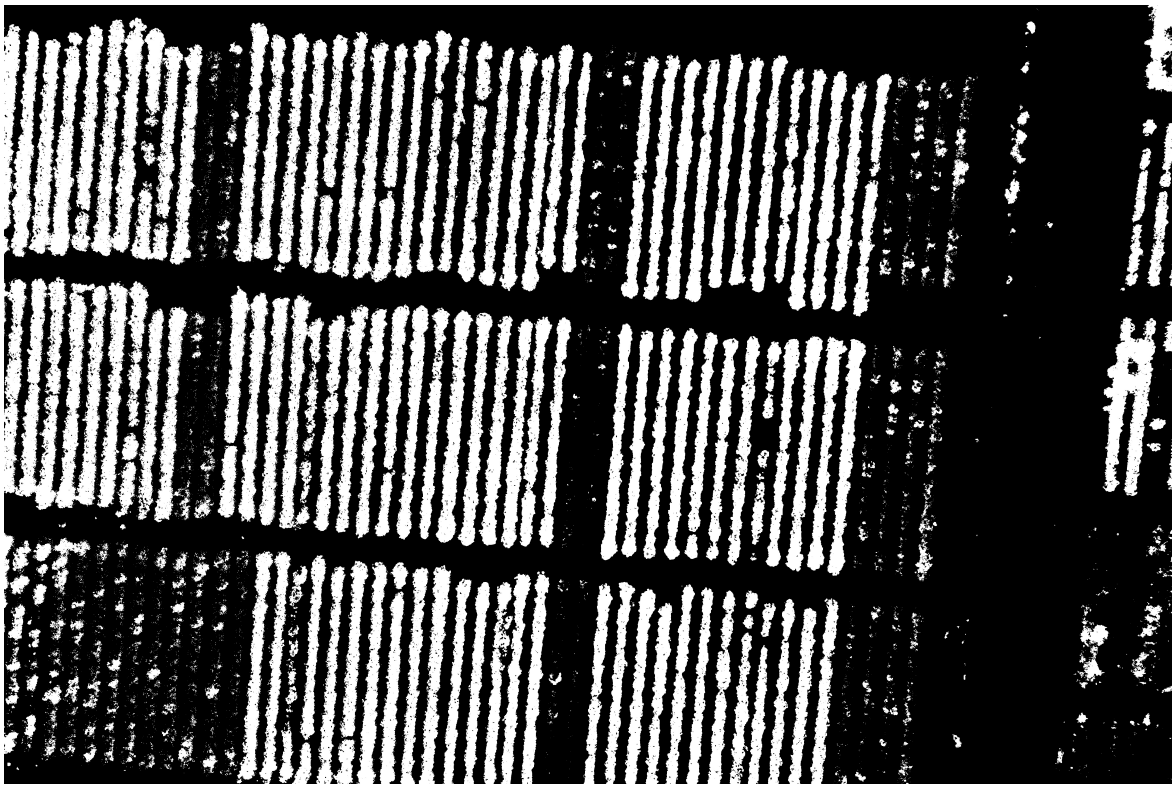

Figure 4.2: Canopeo algorithm binary output.

Figure 4.1 and 4.2 shows the results of the two binarization methods compared. With the Canopeo algorithm some of the groups appeared slightly larger and more solid than in the HSV result. This can be a problem because it can result in the need for more morphology to be employed in order to differentiate between the plots.

5. Results

This section explains the steps for experimentation and the results of our algorithm is explained in detail. Then, a comparison of our algorithm to the algorithm developed by Hamuda is given and the limitations of our algorithm are listed.

5.1 Experimentation

The process of experimentation started with counting the number of plots in each image, all 435 plots were recorded. Then, our algorithm was applied to all 12 of the sample images given. The precision, recall and accuracy of our algorithm was evaluated based on the results outputted after being applied to the images. The next final step was to apply the Hamuda algorithm [2] to the images and calculate the precision, recall, and accuracy of both system's results to evaluate how the they compare to each other in effectiveness.

5.2 Results

The images were captured in a cotton field in Beeville, Texas under sunny conditions. The images of the crops are captured weekly and our algorithm for detection of the crops is most effective when the images to be evaluated are of the cotton crops in the middle stages of the growth cycle. To determine which contours to localize, a filter by contour area was implemented. This was most effective when the minimum contour area was 1000 and the maximum was 10000. The minimum and maximum contour area can be altered depending on the size of the crops being evaluated and the height of the UAS when capturing images, but those were the numbers chosen for the purpose of this study. The crop detection algorithm was evaluated using mostly images taken on sunny days. It is able to distinguish between crops, weeds, and soil to correctly identify cotton crops with an average precision, recall and accuracy of 94.2%, 100% and 94.2% respectively.

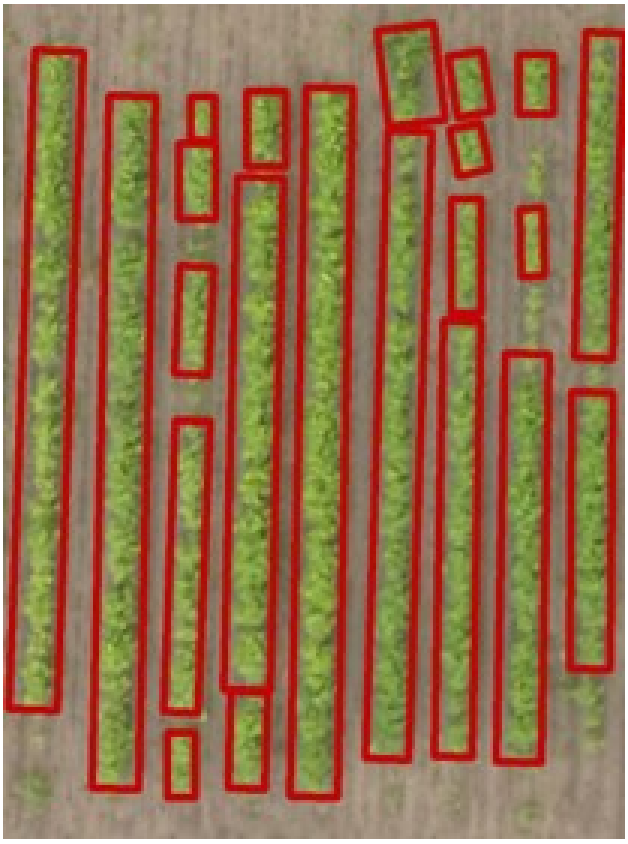

Figure 5.1: Our system's results on one of the sample images.

In figure 5.1 every plot is considered to be correctly detect. There's multiple split plots and each one of the fragments are detected individually but in each plot more than half of them are detected. So, in this image our algorithm had a recall, accuracy and precision of 100%, 100% and 100% respectively.

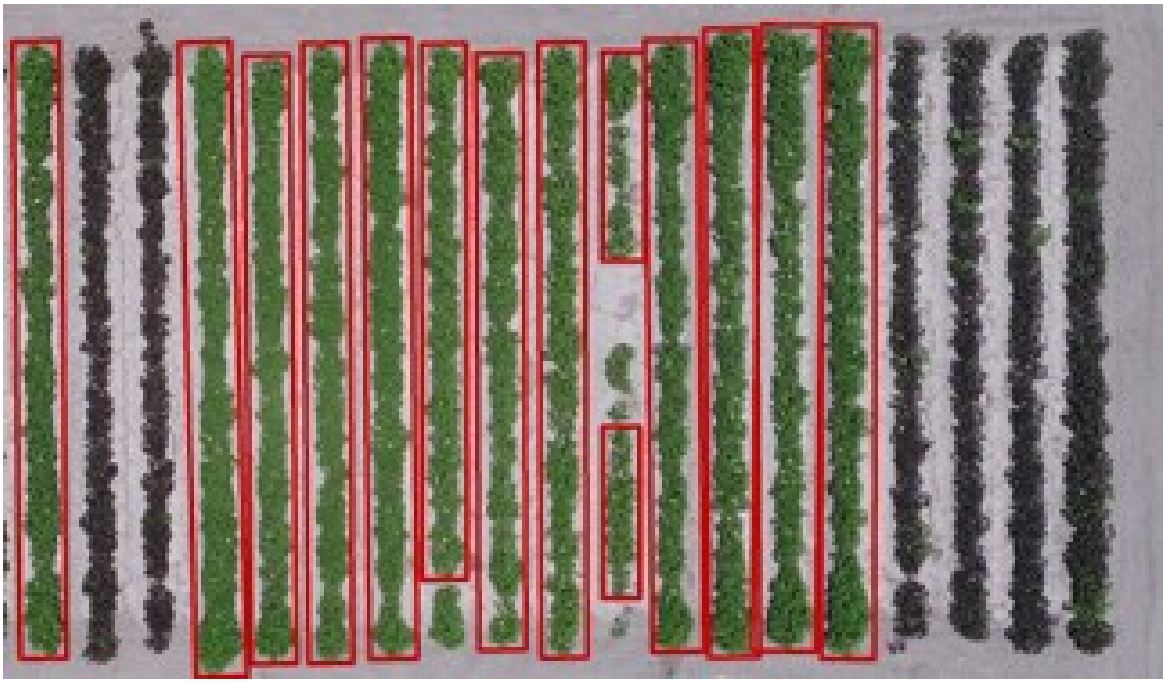

Figure 5.2: Another output of our algorithm.

In figure 5.2 there are both green plots and non green plots present in the image. The non-green plots were ignored, as they were supposed to be, and the green plots were effectively detected. There is a broken plot in the middle of the image. The fragments were detected individually, as expected.

Figure 5.3: Image of green plots correctly identified and non-green plots effectively ignored. The plots are oriented vertically.

In figure 5.3 there are green and non green plots. The plots are also at a different angle than the plots in figure 5.2. The results in this image prove that the orientation of the plots in the image have no effect on the results of the algorithm. To develop a system to achieve these results was one of the goals of the research.

5.3 Comparison to Hamuda's crop detection algorithm

Hamuda et. al. developed an algorithm that is able to detect cauliflower with precision and recall as high as 99.04% and 98.91% respectively. The problem with that algorithm was that it was used on cauliflower, which appear to be easily distinguishable because of the distance apart each crop is planted from each other. When the crops are planted far apart there is no need for an algorithm that knows how to determine if there are multiple of the same crop in the area. When employing Hamuda's algorithm using images of cotton vegetation this fact was easier to see.

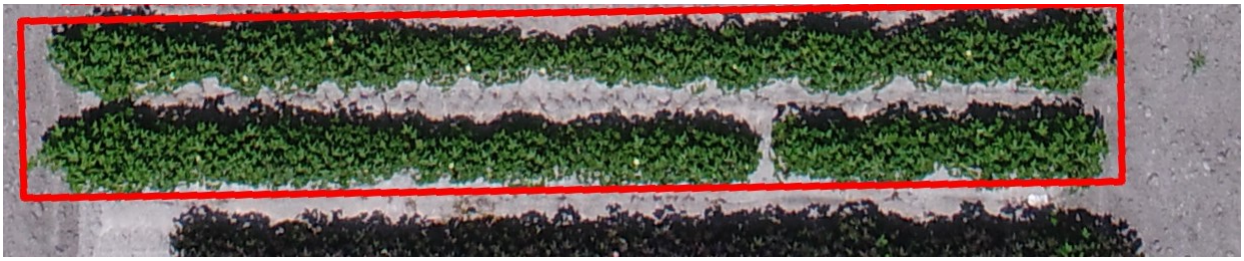

Figure 5.4: Cotton vegetation results when Hamuda's algorithm was applied

Figure 5.5: Developed method results when applied to cotton vegetation.

The results of Hamuda's algorithm is displayed in table 5.1. The results were evaluated

based on how many plots were correctly detected and localized. If it failed to localize a plot by itself by localizing the plots as a group the plots inside of the localizing polygon were counted as being incorrectly identified. Based on the numbers its clear that the method has trouble distinguishing between connected plots and a single plot.

Image	1	2	3	4	5	6	7	8	9	10	11	12	Accuracy	Precision	Recall
Hamuda	22	19	21	24	15	13	9	24	128	13	15	31	77.9%	100%	77.9%
Actual Amount	57	19	23	27	30	19	9	26	141	26	26	32	100%	100%	100%

Table 5.1: Table displaying results of Hamuda's algorithm.

Because there was no way to tell how many plots of vegetation exist in an area when using Hamuda's algorithm, our algorithm in this paper takes a slightly different approach to detecting and classifying the crop pixels. It first determines how large the width of each polygon it will draw around the crops will be and divides it depending on how much larger it is than the width of a polygon should be when drawing around just a single plot of vegetation. Our algorithm was able to determine how many lines of crops were in one group of plots that appear grouped together. The results of our algorithm and the Hamuda algorithm were placed into a table in figure 5.2 so the results can be compared side by side. Based on the results it is apparent, more often than not, that our algorithm correctly identifies the plots at a with greater accuracy than Hamuda's algorithm. The figures 5.6, 5.7, 5.8 shows the recall, accuracy and the precision of the algorithms respectively.

Image	1	2	3	4	5	6	7	8	9	10	11	12	Accuracy	Precision	Recall
Hamuda	22	19	21	24	15	13	9	24	128	13	15	31	77.9%	100%	77.9%
System	54	19	22	26	22	19	9	26	140	22	20	31	94.2%	100%	94.2%
Actual Amount	57	19	23	27	30	19	9	26	141	26	26	32	100%	100%	100%

Table 5.2: Table displaying results of our algorithm and Hamuda's algorithm.

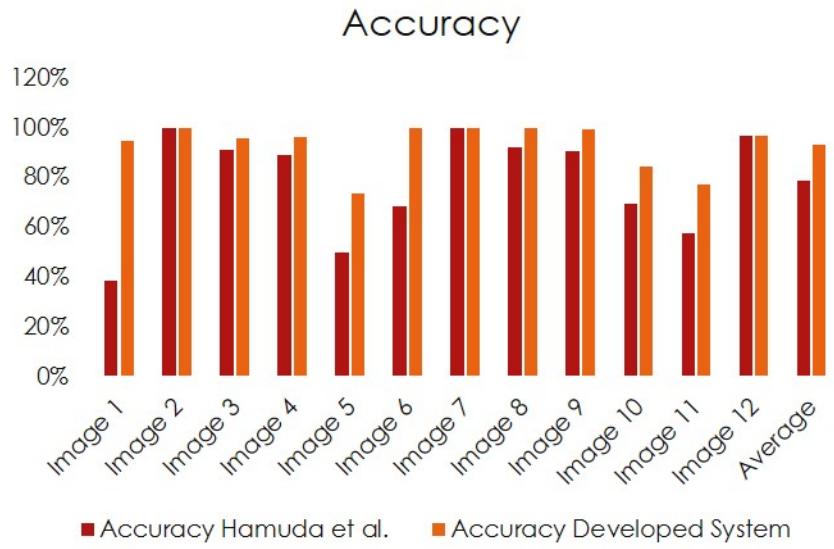

Figure 5.6: The accuracy of Hamuda's algorithm and our algorithm by image along with the average of both systems.

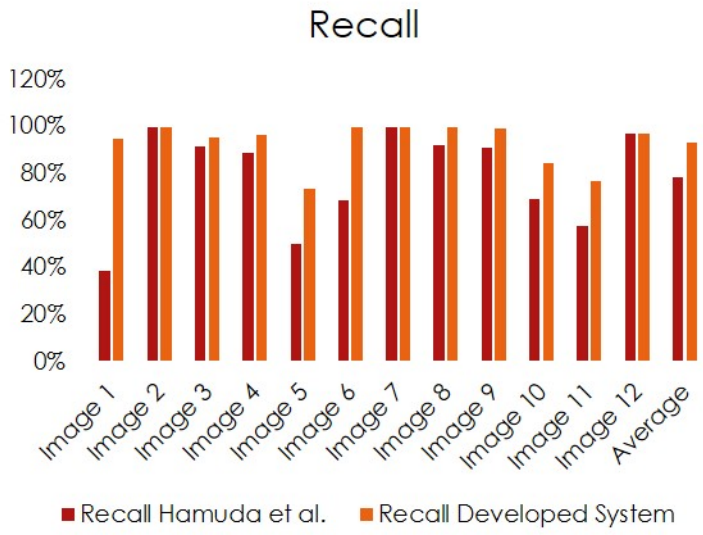

Figure 5.7: The recall of Hamuda's algorithm and our algorithm by image along with the average of both systems.

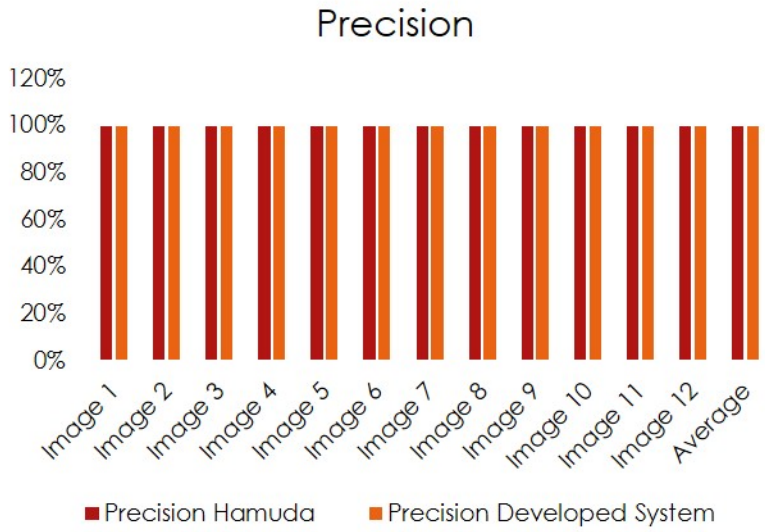

Figure 5.8: The precision of Hamuda's algorithm and our algorithm by image along with the average of both.

5.4 Limitations

Even though some plots can appear small because of them being underdeveloped, they are important to the research. Because of this, when developing the proposed method it was understood that severely undeveloped plots will be lost due to connected component size filtering and morphological operations. Because of this, researchers will lose the ability to collect valuable information from these plots. Also, different times of the year bring different heights and thickness of the plots. Inclement weather is capable of bringing less than preferable growth patterns when dealing with crops. These are reasons why it is important to pay close attention to the preprocessing methods being used in the process.

Another limitation is how inaccurate the method gets when it comes to plots farther away from the center of the image. The plots closer to the center of the image have more clear boundaries than the ones farther away. The ones farther from the center may have clear boundaries when viewed from above but when viewed from the angle the photo was taken from, depending on the plot heights, the crops may appear to be connected. Also,

some of the crops may have expected growth patterns which may make them grow towards each other. In both of these situations the crops may appear to be connected which causes our algorithm to have trouble disconnecting plots during morphological operations.

Plots that are smaller may be missed due to the morphological transformations applied early in the process. Because of this, polygons won't be drawn around them to indicate they have been detected. This means that the development of the plots may cause another noticeable limitation.

The proposed method is not effective in the beginning or toward the end of the crop growth stages. The images used are images captured of cotton crops in the middle stages of their growth. This is the most effective time to use the proposed method because of the fact that this is the time where the individual crops have grown enough to appear as one crop. At the final stages it is harder to use the method because the crops normally appear to be one large mass of vegetation. This makes it extremely difficult for the proposed method to differentiate between each individual plot.

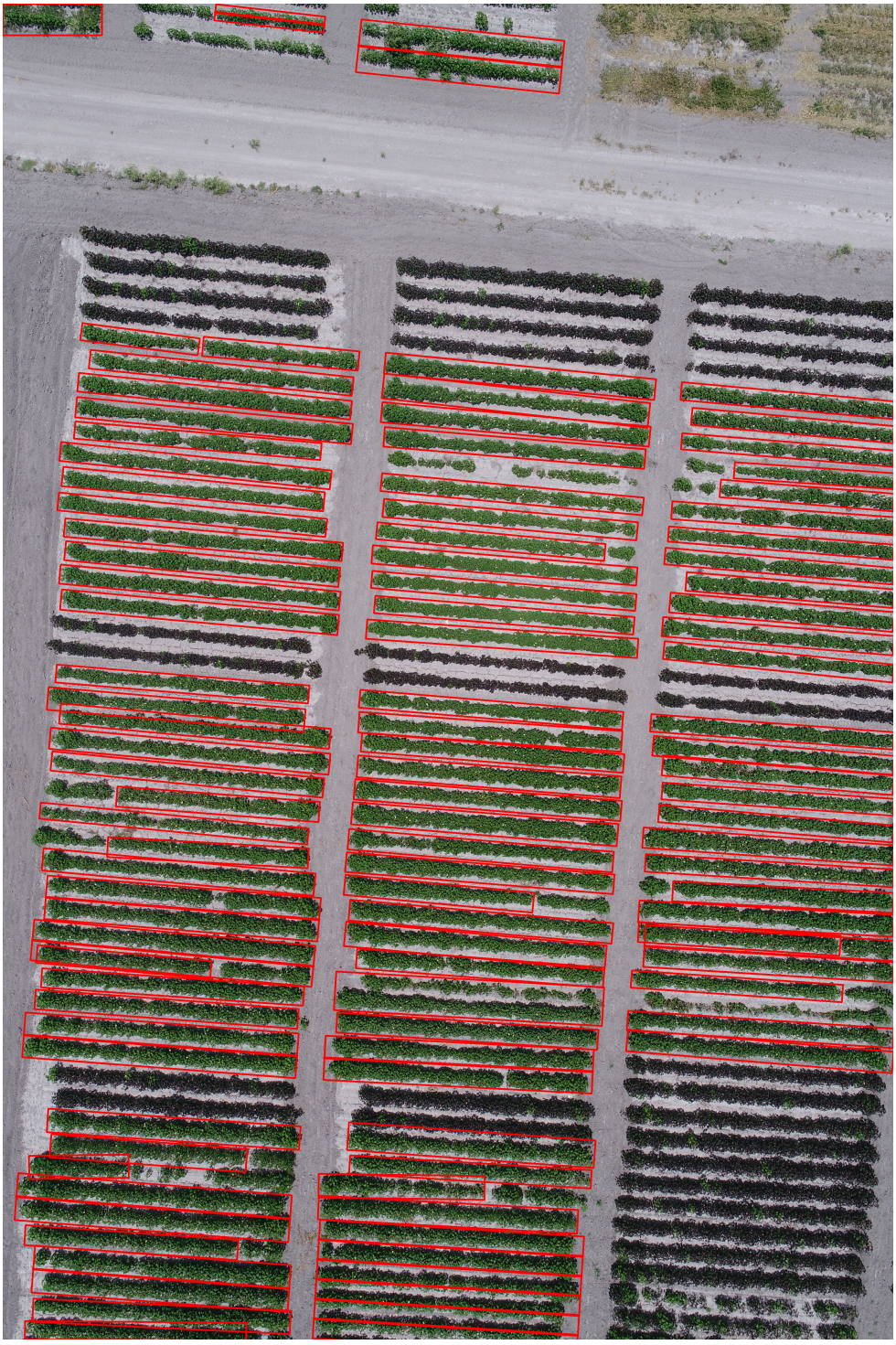

Figure 5.9: Final result of our crop detection algorithm.

6. Conclusion

An automatic crop detection algorithm was developed to be able to detect and localize plots of cotton vegetation. The images used were taken in the middle of the growth cycle, using Unmanned Aerial Systems (UAS), because that's when the algorithm is most effective. If images from early parts of the growth cycle are used the algorithm will not be able to detect the crop locations because the algorithm filters by size of objects. This means that since the objects in the images are too small then they are ignored which results in a decrease in effectiveness. If the images used are taken closer to the end of the growth cycle the crops in the image will appear to be one large collection of greenery because of the spread of fully matured cotton vegetation. There were three main goals of the research.

- Develop a method that will automatically draw bounding polygons around plots of cotton vegetation quickly
- Develop a method that will be able to work with multiple images with cotton crops of different orientations and lighting conditions, aside from any image taken at night time.
- Develop a method that will not have difficulty separating plots that grow together and/or appear as one large plot

These goals were set so that a system could be developed to improve the working conditions of agricultural researchers and farmers. Our algorithm takes less than a second per image to detect canopy plot boundaries and localize plots of vegetation. This is exponentially faster than the manual method used by the Agrilife research scientists. The plots in the images had differing orientations and as long as the images were well lit, our algorithm had no difficulty detecting the plots and drawing the polygons, regardless of the orientation. Our algorithm effectively deals with complex situations where the plots are harder to differentiate because

they grew together. It takes the large polygon drawn around the plots and divides it into the correct amount of rectangular polygons.

Our algorithm was built using the Python programming language. It uses OpenCV to create a binary image (mask) by applying filters after the RGB to HSV conversion has taken place. The filter applied singles out any green pixel, which lies in the hue value range of 40-70, and turns the green pixels to white while everything else is turned to black. The resulting mask was then put through a combination of morphological operations, such as erosion and dilation, and filtering using ConnectedComponents. ConnectedComponents is a method often used as a means of noise reduction. It counts the white pixels that are grouped together, by means of 4 neighborhood or 8 neighborhood, and the amounts are then used to set limits and filter out groups that are too large or too small. If the groups do not fall into the range then the pixels are turned to black in the mask, otherwise they are left white. Those of the plots that are easily distinguishable are easily localized correctly after the morphology and filters but there are situations in which the morphological operations are needed to connect or disconnect plots. Afterwards, the contours are detected. A contour is a curve or line segment joining all continuous points along the outside/boundary of an object.

Sometimes the plots don't connect or disconnect the necessary plots or fragments of plots, so there are a few challenging issues that arise and must be resolved. The issues are:

- Broken plots
- Connected plots
- Underdeveloped Plots
- Contours created by connected greenery
- Polygon drawn inside larger polygon

For the broken plot issue a situation could not be developed to resolve it. So the best solution was to localize the fragments of the plots, assuming they were a certain size, individually. For the connected plots issue a mathematical method was developed that found a standard width the polygons are supposed to be drawn with and divided the polygons into the proper number of sections to localize the plots individually. The underdeveloped plots tend to get lost during the preprocessing phase in system. With the morphological operations, such as erosion, the smaller objects sometimes get eroded away completely or made into objects too small to be contained in the condition set during the ConnectedComponents filtering step. Effective methods were developed for the Contours created by connected greenery and also the polygons drawn inside larger polygons.

Once all the issues have been resolved then the algorithm localizes each plot. The algorithm was evaluated in terms of accuracy and the classification effectiveness was measured in terms of precision and recall. In the 12 sample images used for the research there were 435 total plots. The algorithm was applied to all images and the accuracy, precision and recall of the results were then calculated. Then Hamuda's algorithm was applied to the 12 images and its accuracy, precision and recall were calculated. The results of both systems were compared to each other. The accuracy was found to be 94.2%, precision was 100%, and recall was 94.2% for our algorithm. Hamuda had an accuracy of 77.9%, precision of 100%, and recall of 77.9%. Compared to the algorithm developed by Hamuda et. al [2], our algorithm takes the localization of vegetation a step further. Hamuda's used localization to determine where instances of cauliflower are located, but only focused on the indication of a single plant. There was no cases where multiple cauliflower plants may be in close proximity to each other which may cause them to appear merged. Our algorithm gives researchers a way to not only indicate where the plots are located, but also indicate how many plots are located in the specified area if they appear merged and draw the correct number of localizing polygons to show the number of plots in the area.

The decision to use the HSV color space was made mainly because of how much faster the HSV method proved to be than the RGB color space. Speed was a deciding factor because there can be hundreds of images that need to be analyzed. If there is a large amount of images then the algorithm used must be able to deal with every image in a timely matter.

The accuracy, precision and recall of the method developed by Hamuda was similar but their method wasn't able to determine how many plots were within the localizing polygons, which is the largest contribution of this research is. In the future, the algorithm can be improved by developing a way for the plots that are not well developed to be localized and also there should to be a way to connect the parts of individual crops and localize them as one plot of vegetation if they are broken/disconnected. Also, if a situation were to occur where a disease may have turned the cotton crops into a color other than a shade of green the algorithm will lose its effectiveness as it may not be able to find the discolored crops and identify them. So, in the future a method should be developed to handle both healthy and unhealthy plots of vegetation. Also, an improved system should be include an improved polygon splitting algorithm along with a way to draw the polygons at an angle that ensures the polygons don't intersect.

7. Future Work

The algorithm still has its issues. Because of this a more improved system should have a few improvements. The improved system should:

- Correctly localize plots without the plots intersecting

  Because of the orientation of the plots in the images the polygons are sometimes drawn at an angle that makes them intersect with other polygons. This is an issue because when the algorithm from section 3.4.3 is applied, even though the polygons are sometimes drawn correctly the intersection makes our algorithm delete the polygons from the list of polygons to be drawn. This decreases the accuracy and recall of our algorithm. An attempt to make all the polygons drawn at the same angle was attempted but it wasn't effective. The first method tried and applied to the images averaged the angles and straightened all the angles, but that didn't prove to be effective. The second took the mode of the angles and applied the polygons according to the calculated mode but it proved to be just as ineffective. The problem was that the angle chosen often didn't fit every plot correctly which resulted in the pixels containing sections of the plots to be located outside the polygons drawn to localize them. A more effective method could not be developed for our algorithm.

- Correctly localize under developed plots

  Our algorithm has trouble detecting where an undeveloped plot is. During the preprocessing section of our algorithm, the objects in the images have morphological operations applied to them. These operations are explored in section 3.1.3. Sometimes the operations decrease the area of the objects. This is not good because when the plots are underdeveloped the corresponding white objects in the mask are already small. Sometimes decreasing the area of these objects can completely erase the objects from

the image. Also when the objects are evaluated based on size using ConnectedComponents, this is also explained in section 3.1.3, if the size of the objects in the mask that correspond to the underdeveloped plots are small they are often erased because the area of the objects are below the lower threshold set by our algorithm.

- Contain an improved polygon splitting algorithm

  In our algorithm if there is a polygon that has been drawn too large, instead of removing the polygon and drawing the correct number of localizing polygons the algorithm, draws lines to split the large polygon. When drawing the polygons the pixels in the image act as a coordinates. In order to draw the correct amount of localizing polygons the coordinates of the image where the four corners must be drawn have to be determined. During the localizing process the polygon's corners are usually determined by means of finding the angles, length, width and centroid. A system could not be developed to draw multiple polygons without first having characteristics.

- Detect broken plots and connect the corresponding localizing polygons

  The algorithm places the broken portions of the plots inside of their own polygon instead of placing them inside one large polygon. A solution could not be developed to resolve this issue. There must be a way to determine when the polygons should be merged. When the objects are in close proximity sometimes they can be merged using methods from section 3.1.3, such as closing or dilation, and then localized together. But if they aren't close enough to each other to be brought together using any of the methods in section this will not be the case.

REFERENCES

[1] M. Hassanein, M. Khedr, and N. El-Sheimy, "Crop row detection procedure using low-cost uav imagery system," International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences, 2019.

[2] E. Hamuda, B. Mc Ginley, M. Glavin, and E. Jones, "Automatic crop detection under field conditions using the hsv colour space and morphological operations," Computers and electronics in agriculture, vol. 133, pp. 97–107, 2017.

[3] S. Dayan, "Identification and classification of bulk fruits images using artificial neural networks," International Journal of Engineering and Innovative Technology (IJEIT), vol. 1, no. 2, 2012.

[4] A. Patrignani and T. E. Ochsner, "Canopeo: A powerful new tool for measuring fractional green canopy cover," Agronomy Journal, vol. 107, no. 6, pp. 2312–2320, 2015.

[5] S. N. Wei Guo, Uday K. Rage, "Illumination invariant segmentation of vegetation for time series wheat images based on decision tree model," Computers and Electronics in Agriculture, vol. 96, pp. 55–66, 2012.

[6] Q. W. Liying Zheng, Jingtao Zhang, "Mean-shift-based color segmentation of images containing green vegetation," Computers and Electronics in Agriculture, vol. 65, pp. 93–98, 2009.

[7] D. M. Woebbecke, G. E. Meyer, K. Von Bargen, and D. Mortensen, "Color indices for weed identification under various soil, residue, and lighting conditions," Transactions of the ASAE, vol. 38, no. 1, pp. 259–269, 1995.

[8] T. Kataoka, T. Kaneko, H. Okamoto, and S. Hata, "Crop growth estimation system using machine vision," vol. 2, pp. b1079–b1083, 2003.

[9] S. C. Unseok Lee, "An automated, high-throughput plant phenotyping system using machine learning based plant segmentation and image analysis," PloS ONE, vol. 13, 2017.

[10] M. J. Shepherd, L. E. Lindsey, and A. J. Lindsey, "Soybean canopy cover measured with canopeo compared with light interception," Agricultural & Environmental Letters, vol. 3, no. 1, pp. 1–3, 2018.

[11] T. Behrens and W. Diepenbrock, "Using digital image analysis to describe canopies of winter oilseed rape (brassica napus l.) during vegetative developmental stages," Journal of agronomy and crop science, vol. 192, no. 4, pp. 295–302, 2006.

[12] Y. Jiang, C. Li, A. H. Paterson, S. Sun, R. Xu, and J. Robertson, "Quantitative analysis of cotton canopy size in field conditions using a consumer-grade rgb-d camera," Frontiers in plant science, vol. 8, p. 2233, 2018.

[13] A. K. Shinde and M. Y. Shukla, "Crop detection by machine vision for weed management," International Journal of Advances in Engineering & Technology, vol. 7, no. 3, p. 818, 2014.

[14] S. Lal, S. K. Behera, P. K. Sethy, and A. K. Rath, "2017 third international conference on sensing, signal processing and security (icsss)," pp. 361–368, May 2017.

[15] A.-J. B. Bjorn Astrand, "An agricultural mobile robot with vision based perception for mechanical weed control," Dordrecht, vol. 13, pp. 21–35, 2002.

[16] X. Bai, Z.-G. Cao, Y. Wang, Z. Yu, X. Zhang, and C. Li, "Crop segmentation from images by morphology modeling in the cie l*a*b* color space," Computers and Electronics in Agriculture, vol. 99, p. 21–34, 11 2013.

[17] S. Z. Minggang Du, "Crop disease leaf image segmentation based on genetic algorithm and maximum entropy," Applied Mechanics and Materials, vol. 713, pp. 1670 – 1674, 2014.

[18] L. F. Tian and D. C. Slaughter, "Environmentally adaptive segmentation algorithm for outdoor image segmentation," Computers and electronics in agriculture, vol. 21, no. 3, pp. 153–168, 1998.

[19] L. Zheng, D. Shi, and J. Zhang, "Segmentation of green vegetation of crop canopy images based on mean shift and fisher linear discriminant," Pattern Recognition Letters, vol. 31, no. 9, pp. 920–925, 2010.

[20] A. M. Jingfeng Xiao, "A comparison of methods for estimating fractional green vegetation cover within a desert-to-upland transition zone in central new mexico, usa," Remote Sensing of Environment, vol. 98, 2015.

[21] A. Hearst, "Automatic extration of plots from geo-registered uas imagery of crop fields with complex planting schemes," Open Access Theses, p. 332.

[22] I. Kavalerov, "Using pattern recognition to automatically crop framed art," 2014.

[23] A. Singh, "Implementing rectangle detection using windowed hough transform," 2015.