Automatic Detection and Segmentation of Lentil Breeding plots from Images Captured by Multi-spectral UAV- Mounted Camera

A Thesis Submitted to the
College of Graduate and Postdoctoral Studies
in Partial Fulfillment of the Requirements
for the degree of Master of Science
in the Department of Computer Science
University of Saskatchewan
Saskatoon

By
Imran Ahmed

Permission to Use

In presenting this thesis in partial fulfilment of the requirements for a Postgraduate degree from the University of Saskatchewan, I agree that the Libraries of this University may make it freely available for inspection. I further agree that permission for copying of this thesis in any manner, in whole or in part, for scholarly purposes may be granted by the professor or professors who supervised my thesis work or, in their absence, by the Head of the Department or the Dean of the College in which my thesis work was done. It is understood that any copying or publication or use of this thesis or parts thereof for financial gain shall not be allowed without my written permission. It is also understood that due recognition shall be given to me and to the University of Saskatchewan in any scholarly use which may be made of any material in my thesis.

Requests for permission to copy or to make other use of material in this thesis in whole or part should be addressed to:

Head of the Department of Computer Science
176 Thorvaldson Building
110 Science Place
University of Saskatchewan
Saskatoon, Saskatchewan
Canada
S7N 5C9

Or

Dean
College of Graduate and Postdoctoral Studies
University of Saskatchewan
116 Thorvaldson Building, 110 Science Place
Saskatoon, Saskatchewan S7N 5C9
Canada

Abstract

Unmanned Aerial Vehicles (UAVs) can be used to extract phenotypic information of plants of individual breeding or research plots if equipped with image detection and segmentation techniques. In our research data set, which are aerial images of lentil fields, each plot contains plants of a single genetic line. Plant researchers are interested in selecting lines with preferred physical traits that increase crop yield. Automatic detection and segmentation of lentil plots will enable plant breeders to monitor and quantify plot phenotypes more frequently and with less effort.

A detection algorithm based on Laplacian of Gaussian (LoG) blob detection and a segmentation algorithm based on a combination of K-means clustering and the random walker algorithm are proposed to detect and segment lentil plots from aerial images captured by multi-spectral cameras mounted on UAVs. Our proposed method is independent of the camera used. Our detection and segmentation algorithm detects and segments lentil plots from contrast-enhanced normalized difference vegetative index (NDVI) images. The accuracy of the detection algorithm was measured using precision and recall, for the detection algorithm the average precision and recall was 96.4% and 97.2% respectively. The accuracy of segmentation algorithm was measured using Dice similarity coefficient, mean squared distance, sensitivity, and specificity. The average Dice similarity coefficient between a detected segmented plot and its ground truth was 0.90, sensitivity was 0.912 and specificity was 0.959.

Acknowledgements

I would first like to thank my thesis supervisor Dr. Mark Eramian of the department of Computer Science at the University of Saskatchewan. Whenever I was in any kind of problem or had question about thesis writing or research, Professor Eramian's door was always open. He continuously allowed this work to be my own work, and steered me in the right direction whenever I needed. In my entire Masters life, I had lots of ups and downs. I found my supervisor in every trouble beside me both as a friend and as a guardian. I will always be grateful to him and remember him in my life wherever I go.

I would also like to thank Ilya Ovsyannikov and Karsten Nielsen who helped me in making ground truths and providing necessary agricultural knowledge whenever I needed respectively. Without their passionate participation and input, my thesis wouldn't be complete.

I would also like to thank Mr. Hao Song who always helped me in any of my programming problems. Without his precious support I couldn't cope up with the latest technology and keep on implementing my thesis.

Finally, I must express my very profound gratitude to my parents for providing me with unfailing support and continuous encouragement throughout the years of my study. This accomplishment would not have been possible without them. Thank you.

CONTENTS

List of Tables

LIST OF FIGURES

1 Introduction

Since the commercialization of unmanned aerial vehicles (UAVs), their utilization in precision agriculture has become more common. UAVs combined with image acquisition, detection, and segmentation techniques can give information on physical plant properties empowering plant researchers to assess phenotypic traits. The purpose of this thesis is to develop a computer tool to automatically detect and segment lentil plots from aerial images.

1.1 Motivation

Canada led the production of lentils in 2017 [1] with 65% of the world's total and 95% of these are produced in the prairie province of Saskatchewan. Because the economy of Saskatchewan is linked with agriculture, this production of lentils plays a major role in Saskatchewan's economy. Having an extreme continental climate which contributes severe winters throughout the province from early November to mid-April, the breeders of Saskatchewan must plant lentils in early May and then harvest in mid-August [35]. As they get little time to grow lentils, continuous monitoring of the lentil fields is required for healthy gross production in shorter period of time. Ordinarily, there are more than 950 plots containing 324 different varieties in a typical Saskatchewan breeding trial and to check each physically is prohibitively time-consuming. These trials are used in early generations of breeding when less seed is available and the number of varieties is large. The goal of breeding is to select varieties with beneficial phenotypic traits, thus, a high-throughput method for image analysis and phenotyping is extremely beneficial at this stage. Automating the process of observing lentil plots can help the agriculturists or plant researchers to screen lentils significantly faster without visiting every one of the plots physically. Figure 1.1 shows an RGB image of a breeding trial captured in 2016.

1.2 Problem Description

Phenotypic analysis of plants from images by plant breeders has received increasing attention in the last few years. As production of lentils play a major role in Saskatchewan economy, University of Saskatchewan has hosted a major lentil breeding program where the goal is to select varieties of lentils with beneficial phenotypic traits. That is why monitoring and phenotyping each individual lentil plot physically is an important but time-consuming task performed by plant breeders. An automatic high-throughput method for lentil plot monitoring would realize significant time savings. Automating this process is challenging due to high diversity in appearance of different lentil plots at growth stages.

Modern UAVs are equipped with high resolution multi-spectral cameras to capture images of any region of interest for further analysis. Capturing multi-spectral plant images using UAVs and using them for agricultural analysis is one of the popular techniques for plant phenotyping using imaging. Our problem is to accurately detect and segment lentil plots which are diverse in size and shape because of different genotypes, semi-regular spacing of plots, and variable in appearance at different stages of growth.

1.3 Objective

The objective of our research is to automate the detection and segmentation of lentil breeding plots, such as those shown in figures 1.1 and 1.2, from aerial images for the purposes of extracting phenotypic information. These aerial images are orthomosaics stitched from many individual images. The plots are quite small, semi-regularly spaced, and contain varying amounts of space between them. Segmentation of individual plots is necessarily the first step in any high-throughput pipeline for phenotype analysis from aerial imagery.

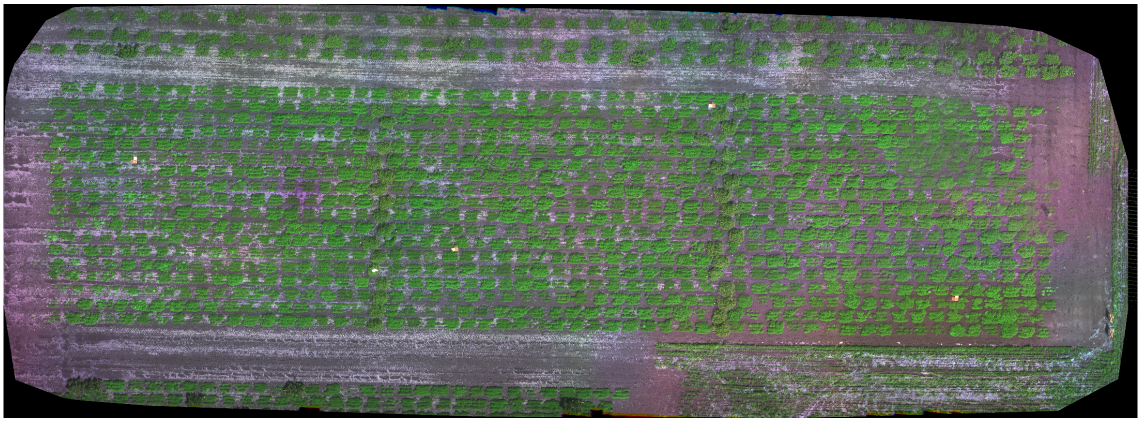


Figure 1.1: Orthomosaic image (RGB) of lentil plots captured on July 13, 2016.

1.4 Proposed Solution

Our algorithm is generalized to work on the images captured by two different cameras. It can detect and segment any lentil plot at different growth stages. The detection algorithm is based on blob detection. The segmentation algorithm is based on a combination of unsupervised clustering and the random walker algorithm for image segmentation. First, we create a normalized difference vegetation index (NDVI) or green NDVI (gNDVI) image depending on whether the input image is from the Micasense camera or Sony α5100, respectively. NDVI evaluates vegetation by estimating the contrast between near infrared and red light. gNDVI quantifies vegetation by measuring difference between near infrared and green light.

The detection algorithm is run on a specified region of interest of the orthomosaic consisting only of the lentil trial so that plots from neighboring trials around the periphery of the orthomosaic are not detected. Once the plots are detected we crop the plot patches based on the scale of the blobs. Finally, we run our segmentation algorithm on each of the cropped patches. The workflow of our entire algorithm can be divided into two phases:

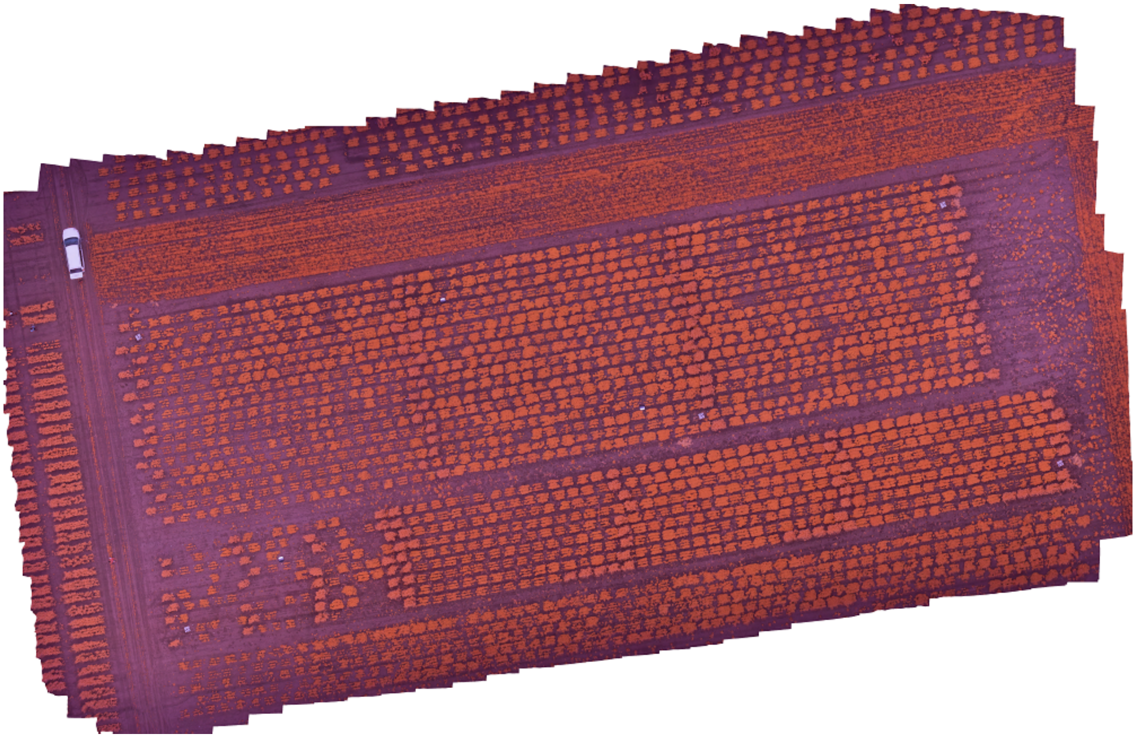


Figure 1.2: Orthomosaic image (NIR-G-B) of lentil plots captured by Sony α5100 camera on July 11, 2017

1. Use of LoG blob detection to determine the location of the plots and crop plot patches using the scale of the blobs.

2. Segmentation of plots from cropped patches using K-means clustering and random walker algorithm.

These phases are discussed in detail in Chapter 4.

1.5 Research Challenges

In the development of the method of detection and segmentation of lentil plots from aerial images, we faced a number of research challenges.

- The first obstacle we faced was when the lentils grow, the plants can grow into adjacent plots which makes the detection of individual lentil plots harder.

- Another challenge we faced in our research is that discriminating between shadows within plots and shadows outside plots. As all the shadows both inside and outside plants had same intensity value, it was very difficult to differentiate between them as we only wanted to keep the shadows within the plot region and discard the shadows outside the plots. The shadows within the plot region are very small in appearance compared to the vegetation within the plot, and hard to distinguish, that is why we planned to keep the shadows within the plots.

- The aerial images we worked with were low contrast, the color of soil and lentil is almost same within individual channel images, which made the segmentation problem harder. This is overcome by using the NDVI metric.
- Another challenge we faced was that the plots were small, semi-regularly spaced, and irregular in size and shape. If the shapes of the plots were all the same it would have been much easier to detect and segment a plot.
- Another difficulty in our research was to generate ground truths for hundreds of plots to validate both the segmentation and detection accuracy of our proposed algorithm.

From the above-mentioned challenges, it is apparent that detecting lentil plots from aerial images which appears in different shapes and segmenting them in different growth stages is fairly a difficult problem to solve and required a dynamic solution.

1.6 Thesis Outline

The rest of the thesis is arranged as follows:

- Chapter 2 presents background material related to this research including image processing.
- Chapter 3 reviews papers related to this thesis.
- Chapter 4 will describe the methodology of our proposed algorithm, will describe the dataset and evaluation methods.
- Chapter 5 presents the results and discussion of detection and segmentation algorithm.
- Chapter 6 presents the conclusion and the future work of this research.

2 Background

2.1 Aerial Imaging

The images used in this research are aerial images. If an image is taken from an aircraft or other flying object then we call it aerial imaging. Platforms for aerial photography can be fixed wing aircraft, UAVs, helicopters or even parachutes. Cameras mounted on these platforms can be triggered remotely and automatically. We have used images captured by cameras mounted on UAVs for our research. Figures. 1.1 and 1.2 show aerial images of lentil plots used in this research.

2.2 Multi-spectral Image

In our research, we have used multi-spectral aerial images. A multi-spectral image is one that catches image information at two or more electromagnetic wavelengths. It is a combination of several monochrome images. Two or more monochrome images can be stacked up to make a multi-spectral image. A good example of a multi-spectral imaging are the 5-channel images acquired from a Micasense Red-Edge camera (Micasense Inc., Seattle, WA). Figure 2.1 shows plot images of 5 different channels captured by Micasense camera.

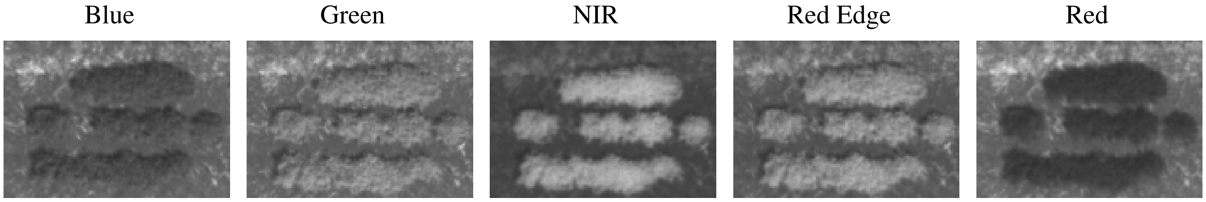


Figure 2.1: Multi-spectral cropped plot images from original orthomosaic captured by Micasense camera; (Left to right) blue, green, NIR, RE, and red channel images. Original image from Professor Kirsten E. Bett and team, P2IRC.

2.3 Down-sampling of an Image

When an image has very high resolution, it takes time to process. Down-sampling that image reduces processing time at the cost of reduced spatial resolution. A digital image can be down-sampled with a certain sampling factor which performs a re-sampling resulting in a shrinked image. Down-sampling of an image can be done by replacing a group of pixel values by one value within a neighborhood. That certain value within the neighborhood can either be the minimum value in the neighborhood which is known as down-sampling with single pixel selection or the average pixel value of the neighborhood known as down-sampling with interpolation. Figure 2.2 shows an example of down-sampling.

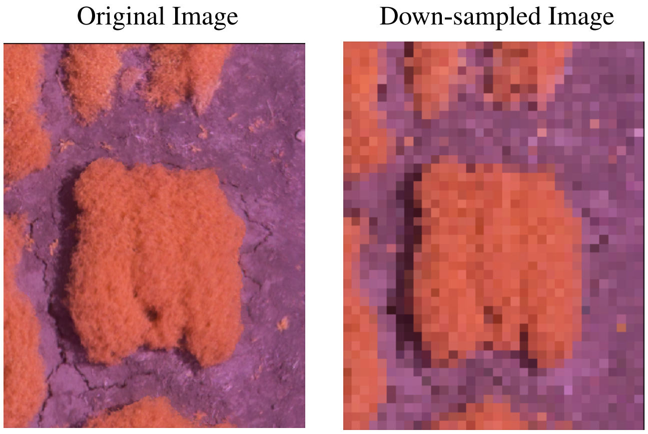


Figure 2.2: Original image and corresponding down-sampled image. Original image from Professor Kirsten E. Bett and team, Plant Phenotyping and Imaging Research Centre (P2IRC).

2.4 Contrast Enhancement

Contrast enhancement is an image enhancement technique for improving the contrast by stretching the range of intensity values to a larger range of values. An 8-bit gray-level high contrast image contains the full range of intensity values. So, a low contrast image can be transformed into a high contrast image by stretching the intensity values in such a way that the intensity histogram of the image spans the full range. In a normal contrast enhancement of an image the lowest gray value GL_{min} is mapped to 0 and the highest gray value GL_{max} is mapped to 255 for an 8-bit image. The remaining gray values are re-mapped linearly between 0 to 255. If I_0 is output pixel value, I_i input pixel value, Min_i is minimum pixel value in the input image, Max_i is maximum pixel value in the input image, Min_0 is minimum pixel value in the output image, and Max_0 is maximum pixel value in the output image then following is the formula for linearly contrast stretching of an image.

$$\mathrm{I_0} = (I_i - \mathrm{Min_i}) * (\mathrm{Max_0} - \mathrm{Min_0})/(\mathrm{Max_i} - \mathrm{Min_i}) + \mathrm{Min_0} \tag{2.1}$$

The ultimate goal of contrast enhancement is to have an image where details are more clearly visible. But linear contrast enhancement is insufficient when the image has a lot of noise and/or when different regions in the image require different amounts of contrast enhancement. In such case, linear contrast enhancement fails to enhance important details from the image and/or over-enhances noise. To overcome the problem of over-enhancement, non-linear methods have evolved which are briefly described below.

2.4.1 Local Contrast Enhancement

Local contrast enhancement of an image is done to extract finer detail from that image. In local contrast enhancement, the image is divided into sub-images for block processing. Based on the statistical properties, suitable contrast enhancement such as that described above in Section 2.4 is applied to each block of the image.

Histogram Equalization

Histogram equalization (HE) is an image-dependent contrast enhancement technique. Mathematically the best possible contrast of an image is when there are an equal number of pixels of each possible intensity which is a uniform histogram. Histogram equalization transforms an image such that the new intensity of pixel $g(x, y)$ is determined by the cumulative density function of the original image's histogram. Histogram equalization enhances the contrast of an image so that details are visible that were not before [16].

Adaptive Histogram Equalization (AHE)

In AHE, the HE mapping is applied to small pixel neighborhoods. In AHE, each pixel is mapped to an intensity proportional to its rank in the pixel neighborhood. Under certain conditions, AHE over-enhances an image in a way that the enhanced image shows undesirable features and can be a slow process [40].

AHE is different from ordinary HE in that AHE does enhancement to each distinct constituent section of an image. That is why AHE is suitable for local contrast improvement of the image. But when an image has fairly homogeneous region causing the pixels intensity values to be similar, AHE over-amplifies noise in such cases. A variant of AHE named Contrast Limited Adaptive Histogram Equalization (CLAHE) prevents and controls over-enhancement, that is why CLAHE has been used for image enhancement in this research. A brief description of CLAHE is given in next section [28].

Contrast Limited Adaptive Histogram Equalization

CLAHE is used to enhance the contrast of an image while limiting the enhancement of noise. In CLAHE, the slope of the mapping function which maps input intensity values to output intensity values defines contrast enhancement. CLAHE limits the height of a histogram by clipping it at a predefined value [29]. Figure 2.3 shows contrast enhancement of an image using both classic Histogram Equalization and CLAHE.

In CLAHE, an image is divided into several non-overlapping regions of almost equal size. First, histogram of each region is calculated, then based on the desired limit of contrast expansion, a clip limit for clipping the histogram is obtained. Second, the histogram of each region is redistributed in such a way that the height does not go beyond the clip limit. Finally, the cumulative distribution functions of the resultant contrast-limited histograms are determined for grayscale mapping. The estimate of the CDF is the sum of histogram of each grayscale scaled by the dimension of the image. [33].

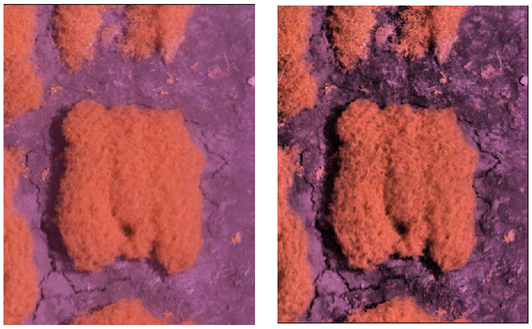


Figure 2.3: (Left) Original image, (Right) Corresponding contrast enhanced image of (a) using CLAHE. Original image from Professor Kirstin Bett and Team, Plant Phenotyping and Imaging Research Centre (P2IRC).

2.5 gNDVI

The normalized difference vegetation index (NDVI) and green NDVI (gNDVI) are graphical indicators. They are used to analyze remote sensing measurements and evaluate whether the object being observed contains live green vegetation or not. NDVI evaluates vegetation by estimating the contrast between near infrared and red light.

$$NDVI = \frac{NIR - Red}{NIR + Red} \tag{2.2}$$

gNDVI specifies vegetation by measuring the difference between near infrared and green light.

$$gNDVI = \frac{NIR - Green}{NIR + Green} \tag{2.3}$$

We used both NDVI and gNDVI in our proposed algorithm, depending on the source of the input image. For the Micasense camera we used NDVI, but if the input image was from Sony $\alpha500$ camera, then we used gNDVI because the Sony camera does not capture a red channel (our Sony $\alpha500$ camera is modified to capture NIR instead of red).

2.6 Object Detection

In computer vision, object detection is a set of techniques which deals with the detection of instances of semantic objects of a certain class in digital images or videos.

2.6.1 Blob Analysis

In computer vision, blob is a region of an image where the region has some constant properties such as brightness or color, compared to surrounding regions. Blob analysis is used to obtain any region of interest [19, 20, 22]. Blob detection methods detect areas from an image which have different brightness or color than the rest of the image. Objects which appear as bright on dark or dark on bright regions having a compact shape in an image are considered blobs. The regions obtained using blob analysis are used for further processing.

The Laplacian of Gaussian

This is a simple approach for blob detection. The idea of this approach is to filter the image with a Gaussian filter with standard deviation σ, and then by a Laplacian filter. A Laplacian filter estimates the second derivative of an image function. A Gaussian function G can be represented as:

$$G\left(x, y, \sigma\right) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.4}$$

where σ is the standard deviation. The Gaussian scale-space volume $L\left(x, y; \sigma\right)$ of image $f\left(x, y\right)$ is

$$L\left(x, y; \sigma\right) = G\left(x, y, \sigma\right) * f\left(x, y\right) \tag{2.5}$$

where * is the convolution operator. Then, the Laplacian operator $\bigtriangledown^2$ can be applied to the Gaussian scale-space representation of an image. So the scale-space representation will be-

$$\bigtriangledown^2 L\left(x, y; \sigma\right) = \left[\bigtriangledown^2 G\left(x, y, \sigma\right)\right] * f\left(x, y\right) \tag{2.6}$$

Here, $L\left(x, y; \sigma\right)$ is the scale-normalized response at point $\left(x, y\right)$ to the LoG filter with scale σ. The Gaussian filtering step before the Laplacian is to smooth an image and to reduce noise. Points of extreme intensity change are highlighted by the Laplacian filter. The scale σ of LoG controls the convergence of blob-like structures to local extrema. LoG filter exhibits strong positive response to bright blobs of radius $\sigma\sqrt{2}$, and a strong negative response to dark blobs of the same size. To automatically capture blobs of different size, a multi-scale approach is adopted. A three-dimensional scale-space volume $L\left(x, y; \sigma\right)$ shown in Equation 2.6 is computed to detect multi-scale blobs with automatic scale selection. Blobs are detected at the 3D extrema in this scale-space volume which means if a voxel $\left(x, y, \sigma\right)$ is larger than any of its 26 neighbors, then $\left(x, y\right)$ is the centre of a blob in the original image. 3D extrema of scale space determine the position and scale of blobs and the magnitude of the extrema can be thresholded to eliminate insignificant blobs [19].

This method has been used in this thesis as a part of the detection algorithm because it works well when objects of interest are compact, blob-like regions. The lentil plots from aerial view looked like blobs and it seemed they had similar color, brightness, texture, size and shape. Moreover, the soil region had different brightness and color than the plot regions. LoG blob detection technique performs well when there exists a good contrast between a foreground and a background region.

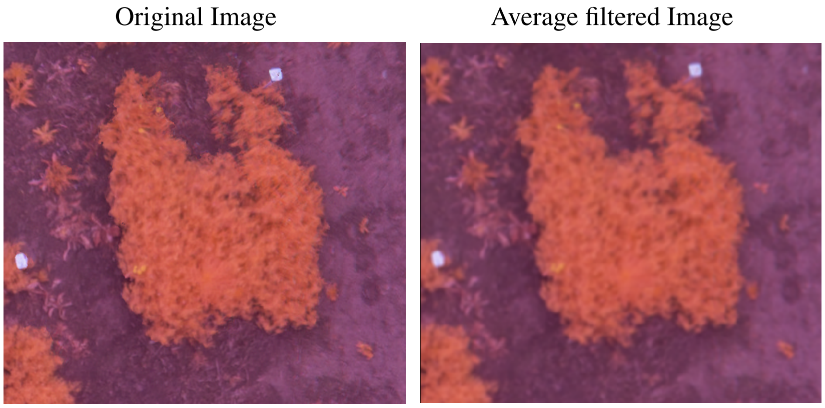
2.7 Linear Filtering

In linear filtering, the new value of a pixel is a linear combination of its neighbors. A linear combination of x_1, x_2, and x_3 is $ax_1 + bx_2 + cx_3$ for real numbers a, b, and c. Coefficients are specified by a filter mask which also defines the neighborhood. The neighborhood is also called mask, or kernel. The filter mask can be represented by an $m \times n$ matrix. Linear filtering operates in a ’sliding window’ fashion. The mask of the linear filter is centred over each pixel, and the sum of the products of mask coefficients and underlying pixels are computed. In this research, an averaging filter has been used as there was no prominent noise in the images and this filter was fast which could serve the purpose of noise attenuation.

Average Filtering

Average filtering is used to smooth an image. The primary use of smoothing an image is to suppress noise. Image smoothing blurs sharp edges of objects that are not noise. In average smoothing, each pixel is replaced with the average intensity of its neighbors. Usually, the neighborhood is an $n \times n$ matrix. For a 3×3 neighborhood, a filter mask h can be defined as shown below. Figure 2.4 shows an image and its corresponding average filtered image. The sharp edges of the plot region are blurred in the average filtered image.

$$\text{M} = \frac{1}{9} \times \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$$



Original Image

Average filtered Image

Figure 2.4: Original image and corresponding average filtered image. Original image from Professor Kirsten E. Bett and team, Plant Phenotyping and Imaging Research Centre (P2IRC)

2.8 Image Segmentation

When an image is partitioned into semantically meaningful regions then it is called image segmentation. We say an image segmentation is successful or failure by determining the segmentation accuracy [32]. There are many types of image segmentation, such as:

- **Edge based segmentation:** In this method, segmentation is done by detecting and linking edge pixels to form a contour. For example, the Hough transformation can be used for implementing edge-based segmentation [43].
- **Graph Theoretic Approach:** In this segmentation technique, images are considered as graphs, pixels are represented as nodes and adjacent pixels are connected by an edge. Then the segmentation is done by determining

which nodes correspond to foreground/background pixels with edge weights and node weights encoding image pixel information. Graph cuts [3] and random walker are [26] graph theoretic methods for image segmentation.

- **Region based segmentation:** In this group of methods, segmentation is done by constructing a region by grouping pixels with similar properties such as color, grey levels or texture into regions [18].
- **Split and Merge:** In this method, an image is recursively divided into a quadrants. The image blocks are split into quadrants if they are not similar; they are merged if they are similar. Quadtree decomposition method for image segmentation can be considered as split and merge technique [27].
- **Watershed Segmentation:** This is a hybrid method described with topographic notions of 'watersheds' and 'catchment basins'. In this method, a gradient magnitude image is interpreted as a topographic surface. Local maxima are 'watersheds' and are the region boundaries and areas with low gradient are catchment basins [7].

Random Walker for Image Segmentation

Random Walker is a method where the user has to give certain seeds and based on these seeds one can have a very good segmented image. Grady et. al. proposed the random walk approach for image segmentation. In his approach, he treated an image like a weighted graph [11].

Let G be a weighted graph where $G = (V, E)$ consists of node set V and edge set $E \subset V \times V$ where each edge $e = (i, j)$ has a weight w_{ij}. The outgoing weights of a node in the graph can be treated as probabilities if, for each node $v \in V$, the weights of the outgoing edges are non-negative and sum to 1.

In random walk, the sequence of nodes visited in the graph G is determined by the probabilities of the outgoing edges from one vertex to another. A random walk starting at node $v_0 \in V$ on the above-mentioned graph G is a sequence of nodes $v_0, v_1, \ldots, v_n$ where each successive node is connected by edges (i.e. $(v_i, v_{i+1}) \in E, i = 0 \ldots n-1$) and each subsequent vertex starting from v_0 is determined by previous node's outgoing edge probabilities. If p_1 is the weight of one outgoing edge e_1 involving vertices (v_0, k_1) and p_2 is the weight of another outgoing edge e_2 involving vertices (v_0, k_2) then $v_1 = k_1$, and $v_2 = k_2$ with probabilities p_1 and p_2 respectively. The v_2 is determined by the probabilities of the outgoing edges. The same procedure is followed for other vertices as well.

Grady et. al [11] solved image segmentation by encoding the problem as random walks on a weighted graph. He considered each pixel as a node, and an created an edge between each pair of adjacent pixels in either a 4-connected or 8-connected fashion. The weight w_{ij} is assigned to each node according to the similarity between two adjacent pixels i and j. The weights are assigned based on equation 2.7.

$$w_{ij} = exp(-\beta(g_i - g_j)^2) \tag{2.7}$$

Equation 2.7 is a Gaussian function of intensity difference between pixel i and j, where β is inversely proportional to the standard deviation of the Gaussian and controls how quickly the weight decreases with increasing intensity difference. Larger β implies a more rapid decay of the weight function with increasing intensity difference thus larger difference between g_i and g_j means lower edge weight which means a random walker is less likely to traverse that edge. Intuitively, β controls how different the intensity of two regions must be to be considered belonging to different segmented regions.

The random walker algorithm requires input labels of certain pixels as foreground and background for a binary segmentation. Let, F and B be the foreground- and background-labeled sets of pixels respectively. F and G are disjoint. For each pixel v, the random walker algorithm calculates the probability p that a random walk starting from that pixel v and will arrive at a pixel in foreground F. If $p > 0.5$ then pixel v is assigned the foreground label otherwise it is assigned the background label in the segmentation.

Grady et. al. [11] in his implementation of random walker did not simulate random walks for computational efficiency. The author simulated the algorithm using electric potential. He computed probability p for each pixel by computing electric potential. He let the pixels with initial foreground label have electric potential 1.0, and background-labeled pixels are ground with potential 0.0. The edge weights were interpreted as electrical conductivity. Then the potential (coinciding with the probability p) at all of the unlabeled nodes can be computed quickly and efficiently. Details of this reformulation and proof of equivalence to the problem of finding the probability of random walk arrival, as well as generalization to more than two segmentation labels, can be found in [11]. The random walker implementation in the Python scikit-image package that was used in this thesis uses the electric potential problem formulation and solution.

2.9 Unsupervised Learning

Unsupervised learning is a kind of machine learning algorithm which can learn the possible outcomes (classes) from the unlabeled data, and also can predict outcomes on new samples. In unsupervised learning, a system is given unlabeled and uncategorized data; but the system will still be able to group that unsorted information according to similarities and differences of the given data.

K-means

K-means clustering is an unsupervised learning algorithm used to categorize unlabeled data. It segments n data points into exactly K groups in which every data point lies in a cluster or group with the closest mean which is the main idea of a cluster. The goal of this algorithm is to find the groups of the input data provided. There are two results of the K-means clustering algorithm:

1. The centroids of clusters and

2. Labels for the input data.

If $[x_1, x_2, ..., x_n]$ is a given set of data points then the goal of K-means clustering is to partition the n data points into $k \leq n$ sets.

There are few ways to implement the K-means algorithm. Lloyd's algorithm is one of the popular techniques to implement K-means clustering. In Lloyd's algorithm [21], if an initial set of K means $[m_1, ..., m_k]$ is given (which can be generated or selected randomly), then the algorithm proceeds by alternating between following two steps.

1. **Assignment Step:** In this step, each data point is assigned to the closest mean vector.

2. **Update step:** Update each cluster mean by recomputing the mean from the vectors assigned to that cluster in step 1.

The algorithm converges when a stopping criterion is met. The stopping criteria can be no data points change clusters, the sum of the distances is minimized, or maximum number of iterations is reached. In this thesis, minimization of the sum of the distances was considered to be the stopping criteria. This K-means algorithm is easy to implement and computationally much faster than hierarchical clustering. K-means provide tighter cluster than hierarchical clustering.

2.10 Basic Morphological Operations

This section discusses the basic morphological operations. These operations process images based on shapes. In morphological operations, a structuring element (a matrix used to identify the pixels in the image being processed and defines the neighborhood used in the processing) is applied to an input image and generates an output image of the same size of the input image. The value of each individual pixel in the output image is determined by comparing a pixel with its corresponding neighbors. Based on the size and shape of both the structuring element and neighborhood, a morphological operation can be done which will be sensible to specific shapes in the input image.

2.10.1 Morphological Dilation and Erosion

Morphological dilation and erosion are done based on a structuring element (SE). A structuring element is a binary matrix defining a neighborhood. The structuring element can be of any shape and the centre of the structuring element is usually the actual centre of the shape. In dilation, the structuring element scans over the binary input image. Dilation "rubber stamps" the SE shape in the output image when the centre of the SE is on a white pixel in the input image. In this process, the input image remains unchanged, the output image gets stamped. Dilation thickens the foreground region in an image. In erosion, the pixel of the output image under the centre of the structuring element changes to black if there is at least one black pixel in the structuring element's neighborhood. Erosion shrinks foreground regions and removes small holes.

2.10.2 Morphological Opening

If an image A is eroded by a structuring element B, and that output is dilated with the same structuring element, then it is called opening.

$$A \circ B = (A \ominus B) \oplus B, \tag{2.8}$$

where $\ominus$ and $\oplus$ denote erosion and dilation respectively.

The morphological opening is used to remove the small objects from the image that do not completely enclose the structuring element while not moving or changing the properties of objects. It is mainly used to remove the connection between two objects in an image.

2.10.3 Morphological Closing

In morphological closing, the dilated image A with structuring element B is eroded with the same structuring element.

$$A \bullet B = (A \oplus B) \ominus B, \tag{2.9}$$

where $\ominus$ and $\oplus$ denote erosion and dilation respectively.

Closing is used to connect the objects that are closer, and fill in small holes protecting the properties of the object. The closing operation fills in the parts of background holes which are smaller than the structuring element.

3 Literature Review

For this research, the literature review was divided into two phases. First, the pieces literature on object detection from aerial and other images were reviewed. Then, we reviewed unsupervised approaches to segment regions of interest from an image. The reason for reviewing such kinds of literature was to find a solution to automatically detect and segment lentil plots from multi-spectral images. We were not able to use any deep learning based models for our problem because of lack of data. We had only 7 images from 3 different growing seasons which is not enough to train a deep model. There are also no public datasets of such kind. So, we focused our literature review on traditional approaches.

3.1 Review on Object Detection

In image processing, object detection means identifying and locating objects of interest from images or videos. In our research, we had to identify lentil plots from multi-spectral aerial images. There were many pieces of literature found on object detection but we prioritized the pieces literature using aerial imagery and having agricultural applications.

Popescu et. al. proposed an approach for detection, localization, segmentation, and size evaluation of flood areas from aerial images which combined a sliding box technique and texture feature analysis [30]. In their method, they acquired the input images with the help of a hybrid system consisting of a mobile part and a fixed part. The mobile part is a UAV and the fixed part consists of a ground control station, a ground data transmitter, and a launcher. The authors classified their input images in two classes: flood and non-flood areas. They used a sliding box technique with 64×64 box dimension to check if each of the boxes has flood or non-flood region with the help of texture analysis. They used mean intensity in red, energy on hue, homogeneity on saturation, and mass fractal dimension on green features for the texture analysis. They calculated the performance of each tested feature for measuring the correctness of classification.

A novel algorithm for detection of buildings using local feature vectors and probabilistic frameworks from very high resolution aerial and satellite images was proposed by Sirmacek and Unsalan [38]. The authors' first local feature vector was Harris-corner-based local feature vectors [15]. Secondly, they used gradient magnitude based support regions (GMSR)-based local feature vectors [42]. They used this feature to extract structural and conditional statistical features to classify land. Thirdly, the authors have used Gabor-filtering based local feature vectors. The authors used a median filter to smooth the image and remove small noises. Then they applied Gabor filtering in different orientations to obtain their local feature vector. Finally, they used FAST (Features from Accelerated Segment Test) based local feature vectors. FAST is a corner detection method, also used for feature extraction [34]. They picked the corner pixels extracted using the FAST algorithm to make a local feature vector. Each of these local feature vectors help to identify a building to be detected. Lastly, they detect tentative building locations as a discrete joint random variable, and estimate their probability density function by those local feature vectors. The authors here used Kernel-based density estimation.

Barrero et. al. [2] have used neural networks (NN) to detect weed plants in rice fields from stitched aerial images captured by UAV at an altitude of 50 meters. The rice field used for this paper was approximately 10 hectares. The authors built a knowledge database based on texture and color descriptors. For texture descriptors, they have used GLCM (Gray Level Co-occurrence Matrix) along with the following Haralick descriptors: Contrast, Correlation, Energy, Entropy, and Homogeneity. For color descriptors, they have used NDI (Normalized Difference Index), where $NDI = \frac{Green-Red}{Green+Red}$. For their neural network (NN), they have used one hidden layer containing a sigmoid activation function, and one linear output layer. They trained the neural network with an increasing number of neurons. They used 70% data for training, 15% for validation, and 15% for testing. They got 99% accuracy in detecting weeds.

Rabatel et. al. [31] introduced a recursive process using Fast Fourier Transform (FFT) along with a selective Gabor filtering-based algorithm for frequency analysis to detect vineyards in aerial images. To make the Gabor filter work efficiently on large aerial images, the authors have done some extra processing on the input image. First, they have normalized the original input image and then Gabor-filtered the image which was then partitioned into 500×500 sub-images for FFT computation. Finally, to find the vine plot in the input image, they iteratively searched for frequency peaks which indicates the presence of vine plots.

Guerrero et. al. [14] automatically detected crop rows of maize fields from images acquired by a mobile agricultural vehicle by segmenting the image by identifying greenness and then applying Otsu's thresholding. Their image segmentation was focused on the separation of green plants. First, they normalized the input RGB image and then they simply fragmented with Otsu's method. For tracing the crop lines, they have adjusted a straight line for specific pixel alignment that are expected to be as crop rows. They verified the correctness of their detection by matching the equations of an expected straight line and an adjusted straight line. They also used a Theil-Sen estimator to correct the expected crop line. This estimator robustly fits a line to some sample points in a plane by choosing the median of the slopes of all the lines through pairs of points.

Gée et. al. [9] did blob coloring analysis and region-based segmentation from perspective wide-view images from ground-fixed cameras to distinguish between weed and crop. Their algorithm was based on the double Hough transform. They identified a vanishing point using perspective geometry of the scene to detect crop rows. The authors in this study have developed a discrete statistical analysis with the assumption that the weed spatial distribution is a random process. They color-transformed the input image. To detect only the crop rows they have run a double Hough transform.

Yamamoto et. al. [44] proposed an approach for the detection of intact on-plant tomatoes using machine learning on images from a conventional RGB digital camera. In their training phase of the proposed method, they classified the pixels into four different classes: fruits, leaves, stems, and backgrounds. Then they calculated 15 color features and generated a decision tree. The generated decision tree was then applied to the input image. Then they extracted the pixels of fruit class; extracted blobs and classified them into 3 classes: single fruit, multi-fruit, and non-fruit. Finally, they calculated the color, texture, and size feature of the blobs and generated decision trees. In the testing phase, they tried to find the class of the blob.

From our review of pieces of literature on object detection, we have seen that many applications using detection of certain plants from aerial images have been found, but no work was found to detect lentil plots from multi-spectral images captured by UAV-mounted camera.

3.2 Review on Image Segmentation

Segmentation is the process of breaking an image into its meaningful constituent regions. It is an important task in computer vision in a sense to uniquely separate and identify an image region so that the region can further be analyzed or post-processed. Segmentation can be done using both the supervised and unsupervised approach. In our problem, we did not have any labeled images of lentil plots. So, we had to segment images in this research using the unsupervised approach.

Eduardo et. al [23] detected and segmented grapes from images of cluster grapes in vineyard. They considered individual grape shape as a sphere to apply Hough transform for detecting circles. They placed a solid background behind the cluster of interest to avoid confusion from other clusters. They also used a circular reference which was detected by the Hough transform. The Hough transform was also used to measure the radius of the reference and all the existing circles were scaled using the radius. Before applying the Hough transform, the authors enhanced the contrast of the image using contrast-limited adaptive histogram equalization (CLAHE).

Frank Y. Shih and Shuoxian Cheng [36] proposed a method for color image segmentation. Their method was based on automatic seeded region growing algorithm. Their entire algorithm can be divided into 4 steps. First, they converted the input RGB images into YC_bC_r color space. Second, they automatically select seeds; seed pixels had similarity to their neighbor and seeds for different regions were disconnected. Third, the color image was segmented in such a way where each segmented portion corresponded to a seed. Finally, region-merging was used to merge similar segmented regions.

John R. Smith and Shi-Fu Chang segmented images using quad-tree decomposition [39]. The authors extracted texture features using quad-tree decomposition. Their proposed method performs segmentation on the frequency domain. Qingbing Zeng et. al. segmented overlapping plant fruit images using marker-controller watershed transform [46]. The authors extracted markers automatically using K-means clustering. Then they extracted contour of the fruits using watershed transform.

There exist so many kinds of literature on segmenting different images but we did not find any specific solution to segment lentil plots from multi-spectral aerial images.

3.2.1 Review on K-means Clustering

We came to know from our literature review that, K-means clustering is the most frequently used unsupervised approach for image segmentation for its speedy convergence. It is one of the most popular techniques among unsupervised clustering algorithms because of its robustness and simplicity [6]. Moreover, we have found that the random walker algorithm results in high quality image segmentation with proven accuracy if good seed labels of the image can be found [11]. So, we wanted to incorporate both the approaches to solve our problem which lead us to specifically focus our literature review on K-means clustering and random walker for image segmentation.

Dhanachandra et. al [6] segmented images using the K-means algorithm, but they used subtractive clusters to generate the initial centres to be used in the K-means algorithm for the image segmentation. To get rid of unwanted regions, a median filter was applied to the segmented region. After loading the input image, they applied partial contrast stretching, then initialized the number of clusters k. Then they calculated the maximum potential pixel of the image to be used as the first cluster centre. They updated the potential pixel of other remaining pixels based on the first cluster centre. Then again found the second maximum potential point based on the first technique and continue the process until a certain number of iterations. Then they separated the pixels using k-means. They repeated these steps until a certain error value. Finally, they applied median filter into the reshaped cluster to get the segmented region. They evaluated their segmentation by analyzing the values of Root Mean Square Error (RMSE) and Peak to Signal Noise Ratio (PSNR).

Ng et. al. [24] used K-means and watershed algorithm on the gradient magnitude and they reduced the number of false edges and over-segmentation for medical image segmentation. In their proposed method, they clustered the input image using K-means clustering; then they used Sobel filter in the clustered image to get an edge map. They used their own automated thresholding [25] to create an improved edge map. To get an initial segmentation map they used an improved watershed transform. Their segmentation algorithm was on the gradient magnitude image, which is different from the watershed algorithm. The initial segmentation map was obtained using rainfall simulation. Their most important part of the proposed method is their post-segmentation merging. From the initial partitions obtained from watershed segmentation R_i whose size is N_i, they calculated the mean intensity of each partition. Then they calculated the difference between mean intensities of partition i and j; and then the difference. Finally they measure the similarity using a similarity measure criterion. They evaluated their proposed algorithm by comparing the number of partitions in the segmentation map. They ran their method on 44 images which obtained 90-95% accuracy when the initial partitions were merged.

Dubuisson-Jolly, and Gupta in [8] segmented aerial images using maximum likelihood classification combined with a certainty based fusion criterion. In their proposed method, they first considered color features based on SAR model and texture feature separately. They computed the likelihood for each class in each feature space independently. The candidate segmentation classes a and b for each pixel was defined based on a machine learning rule. They evaluated their proposed method with three measures. First, τ_1 the percentage of pixels classified as unknown; second, τ_2 the percentage of correctly classified pixels among the known pixels and third, τ_3 the average distance between ground truth and segmented image.

Deng and Clausi [5] tried to avoid estimating parameters for training data. They proposed a new implementation scheme of two-component Markov Random Field (MRF) model by introducing a function-based weighting parameter between the two components. They were able to segment different images using their MRF model which is even able to estimate its parameters. The authors found that the most important criteria for implementing a MRF model is the maximum a posteriori (MAP) criterion. To maximize the posteriori conditional probability distribution, they minimized a certain energy function. The Gibbs sampler and the Metropolis sampler are the two sampling methods used to implement the MAP criterion. They used an annealing scheme in the Gibbs sampler. In their implementation, they estimated four parameters: mean, and standard deviation of each class, weighting parameter for determining contribution of energy function, and a constant for specifying a priori. They used three methods for image segmentation.

1. Simple MRF model with a variable weighting parameter.

2. Simple MRF model with a constant weighting parameter.

3. the K-means clustering method.

Jain and Farrokhnia [17] proposed a segmentation algorithm based on texture analysis and multi-channel filtering theory. The channels are described by a bank of Gabor filters. They proposed a deliberate filter determination scheme which depends on the recreation of the input image from the filtered images. Each of the filtered images was obtained using a blob detector, even different texture was appeared in different regions lying within a blob. A statistical approach was used in a small window within a blob region. Finally, to incorporate the feature images and to produce the segmentation, they utilized an unsupervised square-error clustering algorithm.

3.2.2 Review on Random Walker

Yen et. al. proposed a method to compute a new distance metric called "Euclidean Commute Time" based on a random walk model to be used in the K-means algorithm to generate separate data clusters [45]. They modeled a random walk based on a weighted graph. He represented each observation with a node and connected to k nearest neighbors according to the Euclidean distance. Moreover, they computed the minimum spanning tree to obtain a connected graph [41]. They computed the adjacency matrix A from that graph where $a_{ij} = w_{ij}$ if node i is connected to node j. Then they calculated the Euclidean Commute Time (ECT) distance. They used the derivation of the Laplacian matrix of the graph to get the ECT distance. Finally, they used this ECT distance in K-means clustering, each data point was assigned to a certain cluster using this distance. They compared their result with classical k-means.

Grady et. al. in his another work [13] used random walks for interactive organ segmentation in two and three dimensions. In this work, they mainly focused on the time required to perform the interactive segmentation in CPU and GPU. Grady also showed that, though the random walker algorithm for image segmentation requires a user to specify labels, inclusion of a prior nonparametric probability density model can be used instead of user-labeled pixels [10]. This algorithm can also be represented in four steps.

- Use a simple kernel estimation to produce the probability densities to label each node based on intensities.

- Generate edge nodes.

- Solve the system equation defined in [11].

- Finally, assign each node to the label with highest probability.

Sinop and Grady presented a new image segmentation technique where a certain norm is minimized using a certain method which is related to graph cuts and random walker but different from them [37]. In their work, they explore the segmentation algorithm with a newly defined l_∞ norm, provided a method for optimization and showed that their algorithm produces accurate segmentation. For detailed equations and calculations, see [37]. They have proved in their work that their proposed algorithm produces tighter segmentation than Random Walker.

From our literature review on image segmentation, we have found many authors have adopted many image segmentation techniques but no specific solution on lentil plot segmentation from multi-spectral aerial images.

4 Methodology, Dataset, and Evaluation Methods

Our solution to the problem of segmenting lentil plots has two parts: a detection step and a segmentation step. We designed a generalized algorithm which can detect and segment lentil plots from aerial view even if the input image is from either Micasense camera or Sony $\alpha 500$ camera. We measured the success of the detection algorithm by comparing the number of detected plots using our algorithm with manually counted number of plots. Segmentation accuracy was measured by comparing the segmented image by our segmentation algorithm with hand drawn ground truths.

4.1 Our Proposed Method

Our proposed algorithm is a generalized algorithm to work on the images captured by both the Sony $\alpha 500$ and Micasense cameras. Our algorithm can detect and segment lentil plots from images captured at different growth stages. The blob detection technique was adopted in the detection algorithm. A combination of unsupervised clustering and the random walker algorithm was used to develop the segmentation algorithm. First, we created a normalized difference vegetation index (NDVI) image if the input image is from the Micasense camera or a green NDVI (gNDVI) image if the input image is from the Sony $\alpha 500$ camera, because the Sony camera does not capture a red channel which is mandatory to make an NDVI image. The contrast between near infrared channel and a red channel is estimated to evaluate vegetation by NDVI. gNDVI quantifies vegetation by measuring the contrast between near infrared and green channels.

Our proposed detection algorithm was run on a specified region of interest within the orthomosaic images consisting only the lentil trials so that the different plots from other neighboring trials around the periphery of the orthomosaic are not detected. Once the plots are detected, the detected plots were cropped from the orthomosaic as a rectangular patch based on the scales of the corresponding blobs. Finally, our proposed segmentation algorithm was run on each of the cropped patches. The workflow of our entire algorithm can be divided into two phases:

- Use of LoG blob detection technique to determine the location of the plots and crop the plot patches using the scale of the blobs.
- Segmentation of plots from cropped patches using K-means clustering and random walker algorithm for image segmentation.

The above-mentioned phases are discussed in detail in the next two sections. The brief overview of our proposed algorithm has been shown in figure 4.1.

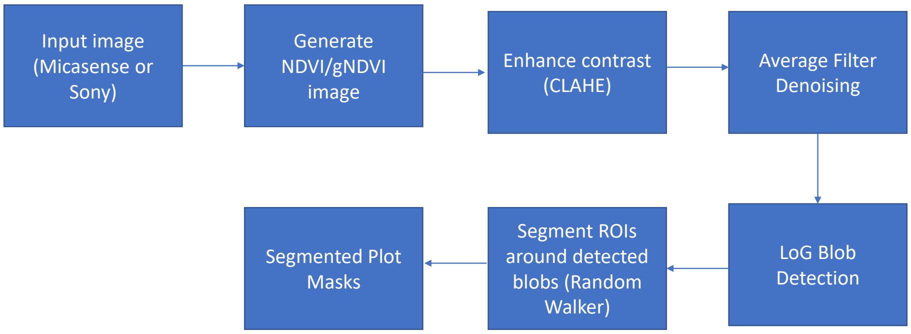

Input image (Micasense or Sony)
Generate NDVI/gNDVI image
Enhance contrast (CLAHE)
Average Filter Denoising
LoG Blob Detection
Segment ROIs around detected blobs (Random Walker)
Segmented Plot Masks

Figure 4.1: Overview of the proposed method.

4.1.1 Lentil Plots Detection

The datasets we used for this research had lentil trials with plots of different breeds having different size and shape. The plots were fairly compact in shape and semi-regularly spaced. We used Laplacian of Gaussian (LoG) blob detection to detect the plots because this algorithm excels at locating compact blob-like regions by detecting extrema in a Gaussian scale-space. In LoG blob detection, the input image $f(x, y)$ is convolved by a Gaussian kernel,

$$G(x, y, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{4.1}$$

at a specific scale σ which brings about a scale-space representation $L(x, y; \sigma) = \bigtriangledown^2[G(x, y, \sigma)] * f(x, y)$. The Laplacian operator exhibits strong positive responses to dark blobs of radius $r = \sigma\sqrt{2}$ and strong negative response for bright blobs. [19]. The overview of our detection algorithm is shown on figure 4.2.

The step by step procedure for our detection algorithm is given below:

- Load the input image.
- Check the type of the input image by checking the number of channels. If the input image has more than 3 channels then the image is from Micasense camera because the Sony $\alpha 500$ camera does not capture more than 3 channels. We check the type of the input image because based on the type we either make NDVI or gNDVI for further processing of the input images.
- After getting the type of the input image we break the image into its constituent channels.
- We down-sampled the Sony images to $\frac{1}{16}$ of its original size and Micasense images to $\frac{1}{4}$ of its original size. The reason for down-sampling the two different images from two different cameras is their different sizes. Both types

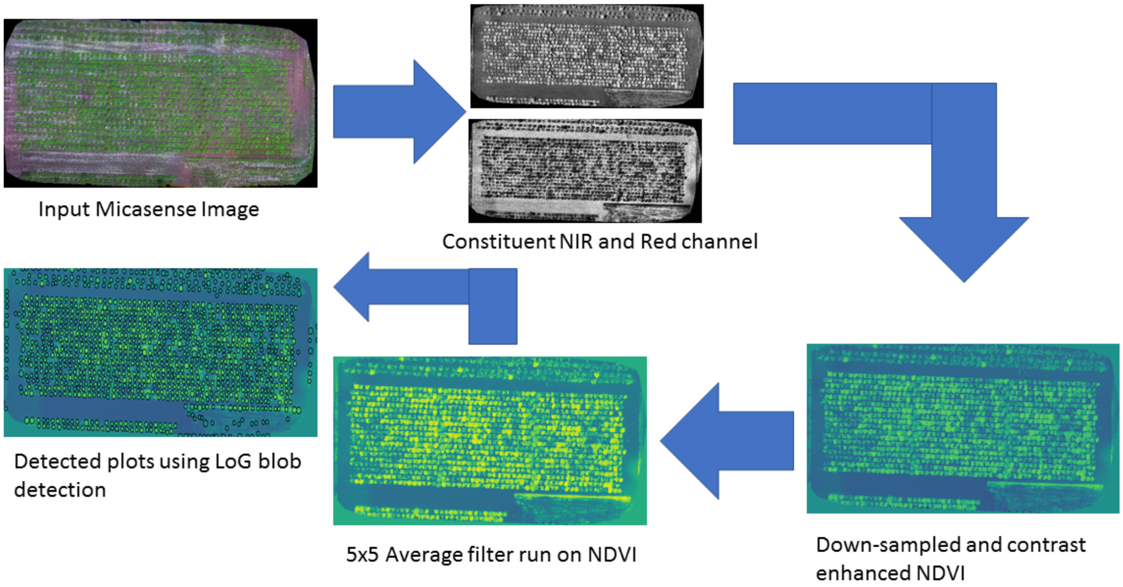

Input Micasense Image
Constituent NIR and Red channel
Down-sampled and contrast enhanced NDVI
5x5 Average filter run on NDVI
Detected plots using LoG blob detection

Figure 4.2: Overview of the detection algorithm.

of images were down-sampled with different ratio so both type of images down-sample to a similar size image so that any post-processing with some fixed parameters can be performed.

- The contrast of the channels needed to compute NDVI was enhanced using Contrast Limited Adaptive Histogram Equalization (CLAHE) to intensify the local detail of the image [28]. The reason for using CLAHE is, it improves the contrast of an image while avoiding over enhancement. Figure 2.2 showed an example of cropped plot patch enhanced contrast using CLAHE. We wanted to enhance local contrast so that we can obtain tiny detail of the plot region to be included in the segmented plot.

- The NDVI or gNDVI was created based on the input image. Figure 4.4 shows an NDVI image made from using NIR and red channels shown in figure 2.1. The image shown here is a false color representation of 2D NDVI.

- Then to reduce noise, a 5×5 averaging filter on the gNDVI was applied because the images are only lightly corrupted by noise and the averaging filter is fast.

- The LoG blob detection algorithm was run to detect the plots. The minimum and maximum standard deviation for the Sony camera was 6 and 12; for Micasense 8 and 12. Table 4.1 shows the parameters used for the LoG detection. Figure 4.5 shows a sample LoG detection output.

  In table 4.1, the values of minimum and maximum standard deviation, and the mask size of noise removal filter which we used are mentioned. These values were selected based on the scale of the original image, the height of the capturing distance of the image with drone, known plot size, known inter-plot distance, and the image

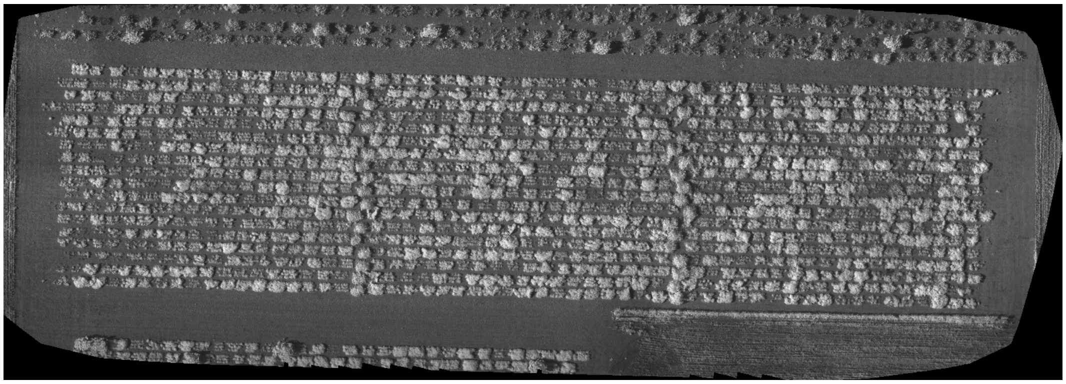


Figure 4.3: NIR channel of a lentil field captured using Micasense camera on 27th July, 2016.

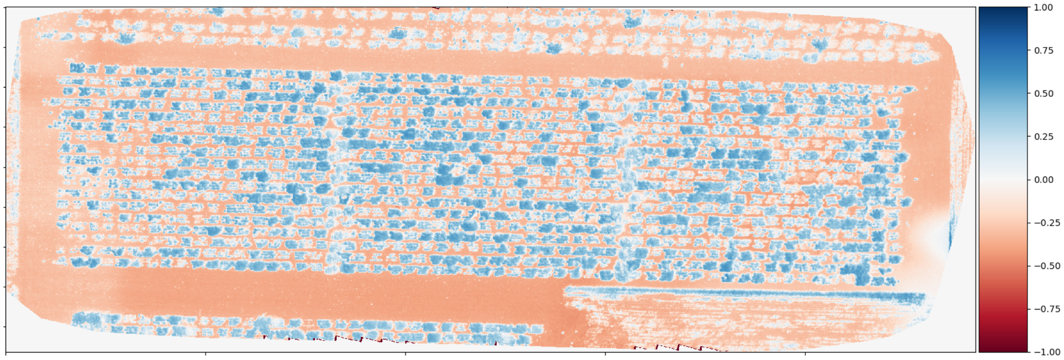


Figure 4.4: NDVI of image in figure 4.3 and its corresponding red channel.

Table 4.1: Table of parameters for detection algorithm.

Parameter Name	**Sony**	**Micasense**
Down-sampling factor	0.0625	0.25
Average filter mask size	5x5	5x5
Minimum sigma	6	8
Maximum sigma	12	12
Threshold	0.6	0.6
Overlap	0.1	0.1

down-sampling factor. These static quantities let us obtain a very good estimate of the approximate number pixels contained within a plot and hence the expected blob scale. The down-sampling factors for the images from both the cameras were selected in a way so that images are down-sampled in such dimension on which static values for different parameters can be used for the proposed algorithm. The motivation for selecting the mentioned down-sampling factors was to keep the images smaller to perform faster post-processing operations without losing much detail of the images. Once the images were down-sampled, the rest of the parameter values were static for images from both cameras. The reason for choosing a smaller filter mask for the averaging filter was the presence of a low amount of noise in the images. Minimum sigma is the minimum standard deviation and maximum sigma is the maximum standard deviation for Gaussian kernel in the scale space. We chose the values shown on table to capture all the plots with minimum number of false positives. Threshold is the absolute lower bound for the magnitude of scale space extrema. Any extrema with magnitude smaller than that threshold are ignored. Overlap defines the maximum amount of overlap of two detections which must have to be considered different detections. We kept the overlap small to avoid any blobs overlapping each other by more than 0.1%. Before we crop the plot patches using the scale of the blobs, we scale up the blob by the down-sampling factor so that we have the plot patches from the original dimension of the image and we do not loose any detail of the patch.

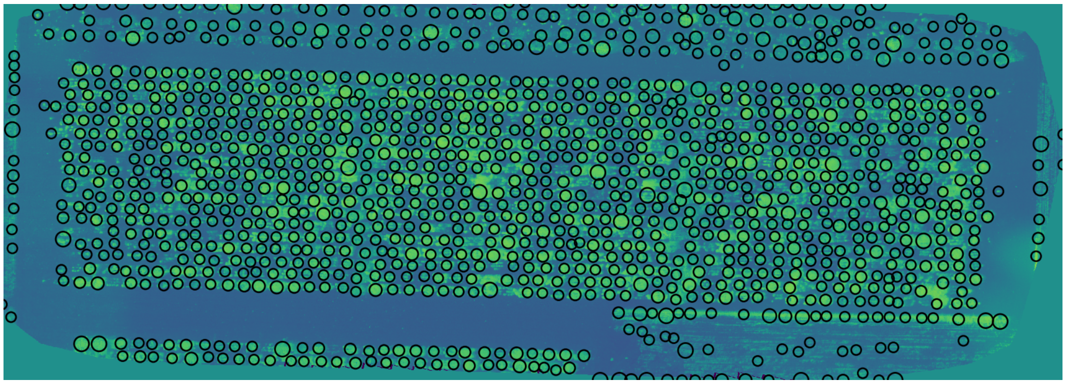


Figure 4.5: Detected plots using our proposed algorithm on image shown in figure 4.4

4.1.2 Lentil Plots Segmentation

For high quality segmentation, we used the random walker for segmenting the plots as random walker for image segmentation results in high quality segmentation even if the image is a low contrast image [12]. If a random walker can be provided with good seed labels, it results in extremely good segmentation, which the authors in [12] proved in their work. We automated the process of labeling the seed pixels using the simple and fast K-means clustering. This algorithm in our research problem was considered suitable because there was a good contrast between the foreground which is the plot and the background which is the soil in the gNDVI images. So, it was reasonable for us to expect intensities of the lentil images to form two reasonably well separated clusters which led us to choose $K = 2$ for K-means clustering. The centroids of the 2-clusters are used to label the pixels for random walker algorithm as they are not likely to be drawn from the range of any overlap of the clusters' distribution. In K-means algorithm, the distance between observations and centroids are minimized.

The step by step procedure of segmentation algorithm is given below:

- For a given cropped patch obtained using the detection algorithm from original input image, the patch is then contrast enhanced using CLAHE and again converted either to the NDVI or gNDVI image based on the camera captured the original image. Figure 4.6 shows a cropped patch and its corresponding contrast enhanced gNDVI. The gNDVI thus gets normalized to have values between -1 to $+1$. gNDVI is a 2D grayscale image. The figure shows a false color representation of the gNDVI.

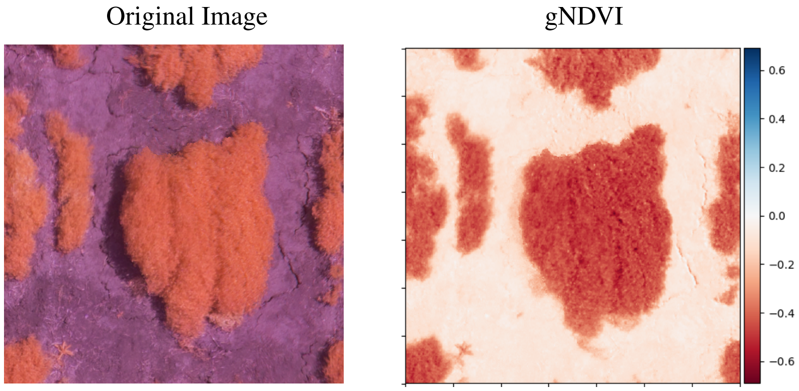


Figure 4.6: Cropped patch from Sony camera and its' corresponding contrast enhanced gNDVI.

- K-means clustering with $K = 2$ is then run on the resulting image with 50 different random initializations. The selection of initial mean strongly influences the performance of K-means. So, we tried different initial mean values to enhance the accuracy of K-means clustering. The initial mean values were randomly chosen 50 times and we made sure no initial mean is repeated more than twice. After running K-means clustering with 50 different initializations. The cluster centroids which resulted in the smallest within-cluster variance, are retained as the final centroids.

- The centroids are then used to choose the brightest and darkest pixels markers. The pixels in the gNDVI image having value larger of two centroids is used as foreground and value smaller of two centroids is used as background seed. Figure 4.7 shows markers obtained using darkest and brightest pixels in each cluster.

- The random walker segmentation algorithm is run to get a binary segmentation. We chose $\beta = 130$ for the random walker. Figure 4.8 shows the output of random walker.

- We extract the largest connected component from the segmentation we get using random walker. The largest connected component was extracted by finding the largest cluster of pixels with the same value which are connected to each other through 4-pixel connectivity.

- We decided the size of the structuring element (SE) for subsequent morphological operations based on the number of pixels in the largest connected component. If the number of white pixels is smaller than $34,000$ then we

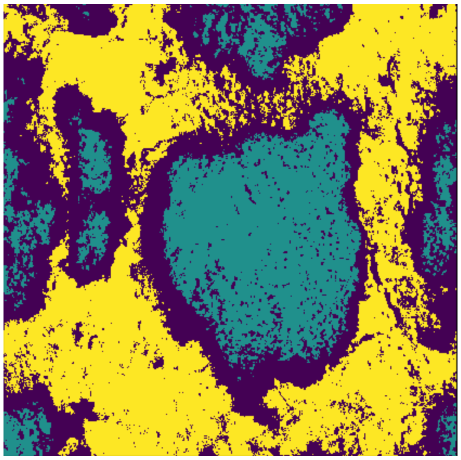


Figure 4.7: Markers of figure 4.6

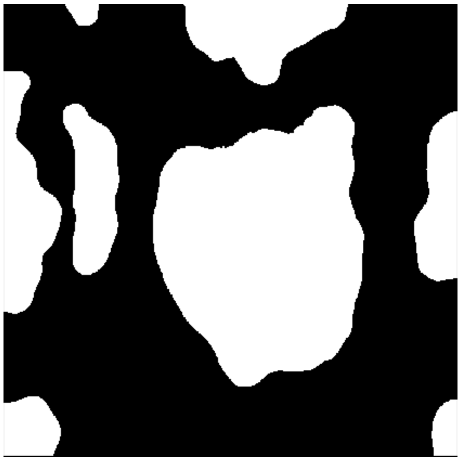


Figure 4.8: Foreground after random walker.

take a radius 27 disk structuring element regardless of the camera that acquired the image. Otherwise a radius 9 disk structuring element is used for Sony camera images. This number of pixels was chosen empirically. On the other hand, we have used a radius 5 disk structuring element for all the Micasense camera images. These radii of structuring elements were derived from the known spatial resolution or scale of the image.

- We run morphological opening and closing with the selected structuring element on the result of random walker. Figure 4.9 shows the output after morphological operation.

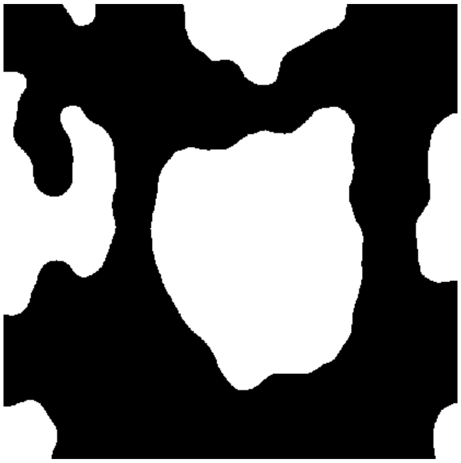


Figure 4.9: Output after morphological operation.

- Any remaining holes within the region is filled.
- Finally, we kept the largest connected component which is our final segmentation. Figure 4.10 shows final segmentation.

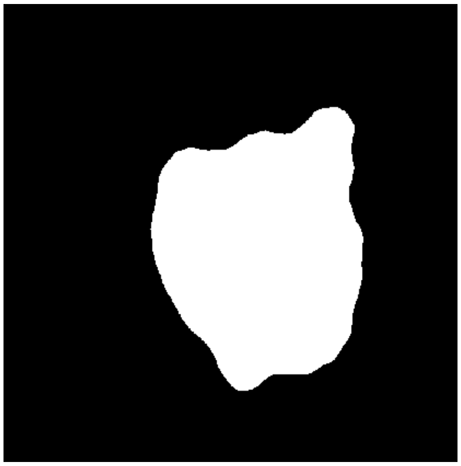


Figure 4.10: Final Segmentation

- The parameters used in this segmentation algorithm are shown in the table 4.2.

Table 4.2: Table of parameters for segmentation algorithm.

Parameter Name	Sony	Micasense
No of inits (K-means)	50	50
β (Random Walker)	130	130
Size of SE	27 and 9	5

In the segmentation phase, we used different sized structuring element for different camera images and for different stages to aid in the processing of plots from early-season images where canopy closure was not confirmed. The main purpose of the post-processing steps in the segmentation phase was to remove regions that were not part of lentil plots and eradicate unwanted plot regions from the neighborhood which were included within the bounding box while cropping the plot patches using the scale of the blobs. The complete flow chart of our proposed detection and segmentation algorithm is shown in figure 4.11.

4.2 Dataset Description

Images used in this research were acquired using Draganfly Commander and X4P models (Draganfly Innovations Inc. Saskatoon, SK, Canada). The UAV carried a consumer-grade Sony $\alpha5100$ (Sony Corporation, Tokyo, Japan) camera which is actually an RGB camera with a modified sensor to capture NIR instead of red channel for image acquisition in 2017 and a Micasense Red-Edge camera (Micasense Inc. Seattle, WA, USA) to gather blue, green, red, red edge, and near infrared bands for image acquisition in 2016. The cameras were mounted on a stabilizing gimbal. Draganfly surveyor software was utilized to create a pre-programmed flight-plan which produced a minimum of 70% front and side image overlap. The 2017 images were then stitched into orthomosaics using Agisoft (Agisoft, St. Petersburg, Russia) and 2016 images were stitched using Pix4D software (Pix4D, Lausanne, Switzerland) by the Research Assistant Mr. William van der Kamp.

We had 7 orthomosaic images of lentil plot fields from 2 locations. Among those 7 images, 2 were captured in 2016 with Micasense camera and the remaining 5 images were captured in 2017 with Sony $\alpha500$ camera from the locations of Sutherland and Rosthern, Saskatchewan, Canada. Table 4.3 shows the dates on which images were collected, on which growth period the lentils were at the time of capture, number of images from each capturing date, number of plots on each captured field, camera used to capture each image, and the field locations of each image.

We had 2 images from early season, 3 from mid-season and 2 from late-season. In this research, the images captured on 24 June and 4 July, 2017 were considered early season images, images captured on 13 July, 2016, 18 July and 20 July, 2017 were considered mid-season images and images captured on 26 July, 2017 and 27 July, 2016 were considered as late season images because lentils on those dates were on those corresponding growth periods. Orthomosaic images from the Micasense camera had dimension of 3000×9000 on average and the images from the Sony $\alpha500$ had dimension of 30000×20000 on average. Micasense camera captured 972 plots altogether lying in the same lentil field and the 2017 images captured with the Sony camera had 1410 lentil plots. Figure 4.12 and figure

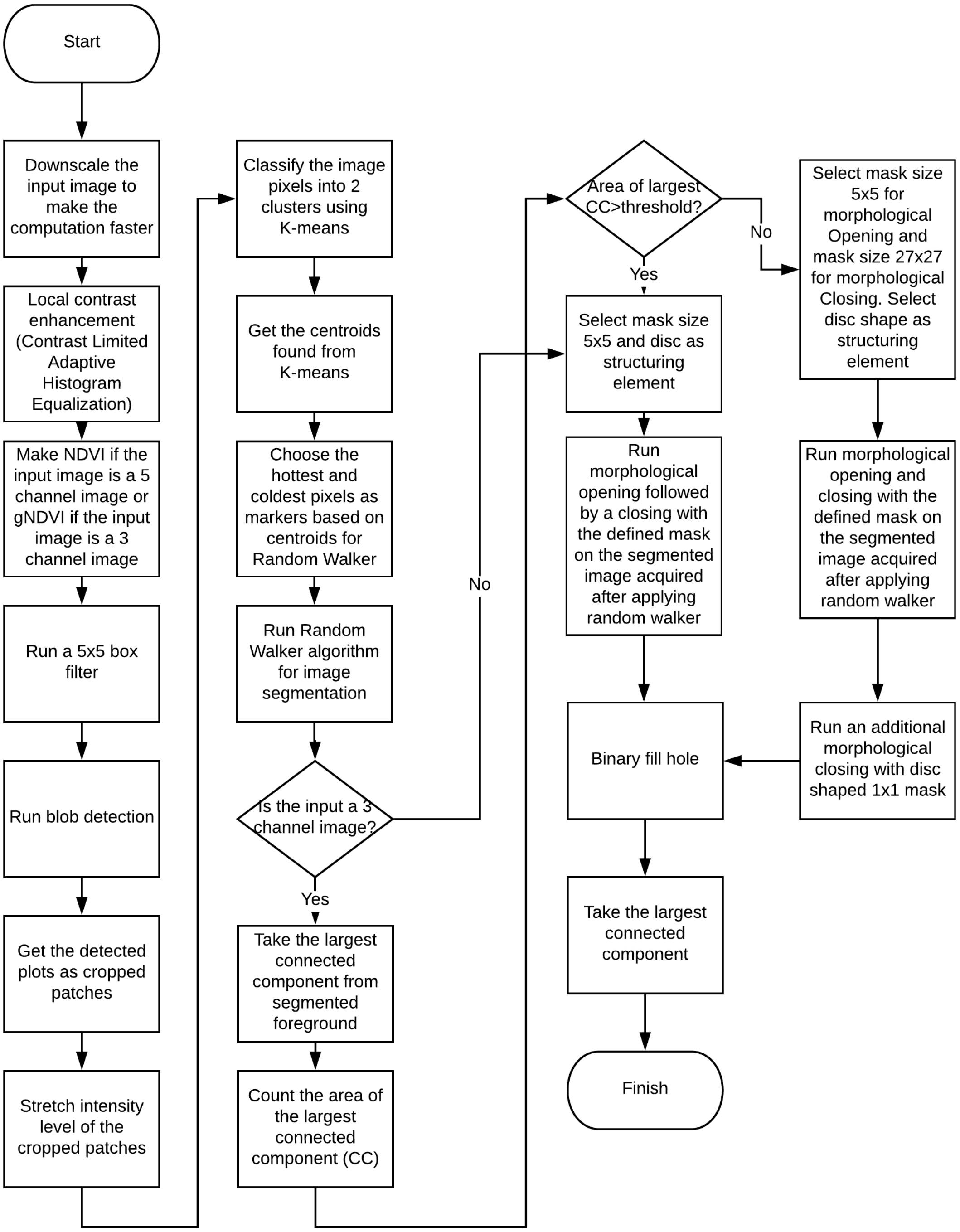

Start
Downscale the input image to make the computation faster
Local contrast enhancement (Contrast Limited Adaptive Histogram Equalization)
Make NDVI if the input image is a 5 channel image or gNDVI if the input image is a 3 channel image
Run a 5x5 box filter
Run blob detection
Get the detected plots as cropped patches
Stretch intensity level of the cropped patches
Classify the image pixels into 2 clusters using K-means
Get the centroids found from K-means
Choose the hottest and coldest pixels as markers based on centroids for Random Walker
Run Random Walker algorithm for image segmentation
Is the input a 3 channel image?
Yes
No
Take the largest connected component from segmented foreground
Count the area of the largest connected component (CC)
Area of largest CC>threshold?
Yes
No
Select mask size 5x5 and disc as structuring element
Run morphological opening followed by a closing with the defined mask on the segmented image acquired after applying random walker
Select mask size 5x5 for morphological Opening and mask size 27x27 for morphological Closing. Select disc shape as structuring element
Run morphological opening and closing with the defined mask on the segmented image acquired after applying random walker
Run an additional morphological closing with disc shaped 1x1 mask
Binary fill hole
Take the largest connected component
Finish

Figure 4.11: Workflow of the proposed detection and segmentation algorithm.

Table 4.3: Data collection details

Capture Date	Growth Season	No. of Image(s)	No. of Plots	Name of the Camera	Field Location
24 June 2017	Early	1	1410	Sony	Sutherland
4 July 2017	Early	1	1410	Sony	Sutherland
13 July 2016	Mid	1	972	Micasense	Rosthern
18 July 2017	Mid	1	1410	Sony	Sutherland
20 July 2017	Mid	1	1410	Sony	Sutherland
26 July 2017	Late	1	1410	Sony	Sutherland
27 July 2016	Late	1	972	Micasense	Rosthern

4.13 shows orthomosaic images captured by Sony camera on early and late growth stage of lentils. Figure 1.2 shown in chapter 1 shows a mid-season orthomosaic image.

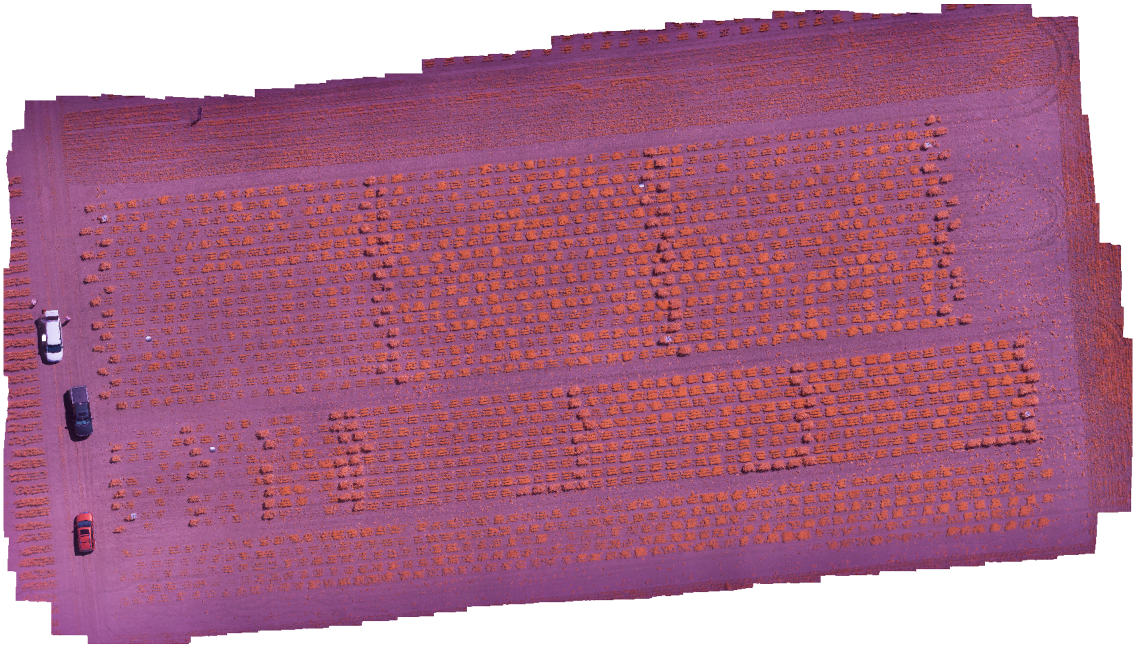


Figure 4.12: Early seaosn orthomosaic image (NIR-G-B) of lentil plots captured by Sony α5100 camera.

4.3 Evaluation Methods

The performance of our detection algorithm was evaluated using precision and recall. We manually counted the number of detected plots detected by the proposed detection algorithm. The known number of rows and plots seeded per row worked as ground truths for the detection algorithm to be evaluated. We computed precision and recall to measure the accuracy of our detection algorithm. Precision is the proportion of objects detected that are relevant. So, in our case, precision was the proportion of detected regions that were actually plots. When an actual lentil plot was detected using

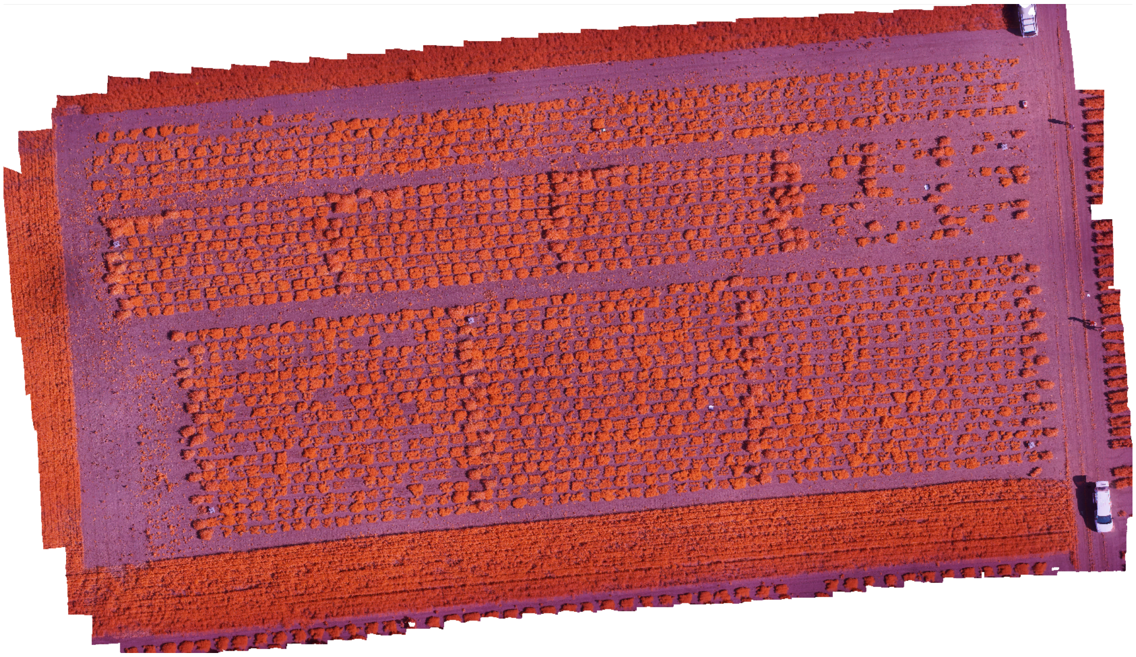


Figure 4.13: Late seaosn orthomosaic image (NIR-G-B) of lentil plots captured by Sony α5100 camera.

our detection algorithm we considered that as a true positive. If the algorithm detected any place other than the actual plots, then that was taken as false positive. Any plot remaining undetected was considered as false negative. Recall is the proportion of lentil plots that were actually detected. We have shown a Precision-Recall (PR) curve for evaluating the performance of our detection algorithm which was created by varying the extrema magnitude threshold parameter.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}} \tag{4.2}$$

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}} \tag{4.3}$$

The overall performance of the segmentation algorithm is characterized by computing the average Dice similarity coefficient (DSC), mean squared distance (MSD) of the plot boundary, sensitivity, and specificity. DSC is used to compare the similarity of two samples. If B is a set of pixels making up the region resulting from the segmentation algorithm and G be the set of pixels making up the ground truth region then the DSC is:

$$\text{DSC} = \frac{2|B \cap G|}{|B| + |G|} \tag{4.4}$$

Mean squared distance measures the average minimum deviation of the segmented region's boundary from the ground truth region's boundary. If $G = g_1, g_2, ..., g_N$ the points in ground truth boundary and $B = b_1, b_2, ...b_M$ the points in the segmented boundary then

$$\text{MSD} = \frac{\sum_{b \epsilon B}(\min_{g \epsilon G} d^2(b, g))}{M} \tag{4.5}$$

Sensitivity is the same as recall and in this context is the proportion of detected foreground pixels that are correctly labeled as foreground. Specificity is proportion of detected background pixels correctly labeled as background:

$$\text{Specificity} = \frac{\text{True Negative}}{\text{True Negative} + \text{False Positive}} \tag{4.6}$$

Greater DSC means the segmentation overlaps more with the ground truth; MSD indicates how closely the boundaries of the segmented and ground truth regions agree. Sensitivity and specificity are measures of an algorithm's performance to correctly classify a plot (in this research) having plot pixels only and discard non-plot pixels. Sensitivity refers to the algorithm's ability to designate all the plot pixels as plot. A high sensitivity means there are few false negative pixels in the segmentation. The specificity indicates the ability of an algorithm to designate pixels which are not part of plot. A high specificity indicates there are few false positive pixels in the segmentation.

4.3.1 Ground Truth Generation Process

We used 'Color Range' and 'Refine Edge' tools of Adobe Photoshop CC 2017.1.1 (18.1.1) to generate ground truth (GT) label images from lentil plot images captured by multispectral UAV-mounted Sony $\alpha500$ and Micasense cameras. Ground truths of the Micasense camera images were made from NDVI image and ground truths of the Sony $\alpha500$ camera images were made from gNDVI images because (g)NDVI image gives exposure to the vegetation.

The 'Color Range' tool lets us automatically contour the plots using a mouse click. This automatic selection was not 100% precise as the plots are heterogeneous in size and shape. So, human interaction was mandatory to define the GT. The 'Refine Edge' tool let manually correct the selection of plot region. To generate the ground truths, following steps were followed by the ground truth raters:

- Opening the input image using Adobe Photoshop (CS6 Extended 13.0) for which the rater wants to generate the GT.
- Choosing 'Select', and then 'Color Range'.
- Select any plot region by mouse click and the 'Color Range' command automatically selects all identical colors, partially select similar colors and completely ignore dissimilar colors.
- Use the 'Shift Edge' control bar of the 'Refine Edge' tool to shift the automatically selected boundary points inwards or outwards based on need. Adjustment of edge shifting was based on the plant pixels. We did not want to miss any plant pixel and liked to keep the largest connected component. The goal was to try to keep the entire plot region and avoid background region as much as possible.
- Save the image.

Finally, the saved plot images were passed to a Python script to make a labeled Ground Truth. These GT images were used to evaluate segmentation algorithm. However, segmentation performed solely by one person is too much work for a single person. So, we involved three people with good domain experience to make the ground truths of randomly selected individual plots. Since segmentations done by individual people will have individual biases, to find out how close the different groups of manual segmentations are we measured Generalized Tanimoto Coefficient (GTC). We had fields of lentil plots from 2 locations. One from Sutherland and other from Rosthern. Images of each field were captured approximately at minimum once every two weeks. We had seven orthomosaic images in total. We used all the images from fields of both the locations to generate ground truths. A Python script was written to generate random (x, y) coordinates located within the source aerial image. If the randomly generated coordinates fell within a plot region, we took a square patch consisting of that plot into consideration for making ground truth from that. We picked at least 29 plot images from each image. Each observer was given 200 plot images to generate ground truths. Among the 200 patches, 100 plot images were common to each observer to measure inter-rater-agreement. We measured the GTC of the common 100 ground truths generated by each individual person. Specifically, GTC was chosen because it is not sensitive to segmentation group size [4]. GTC has a range of 0 to 1 with 1 representing perfect agreement of all segmentations. The acceptance criteria for ground truths was: the inter-rater GTC will have to be greater than 0.75. The ground truths which were used in this research had average inter-rater GTC of 0.96.

5 Results and Discussion

We used Python to run our algorithm and executed both the detection and segmentation algorithms on an Intel Core i7-2600 CPU @ 3.40 GHz processor with 16.0GB RAM. The detection algorithm took 0.03 seconds on average to run on the entire orthomosaic image of its original dimension. The detection algorithm includes the steps of taking input image, breaking the image into its constituent channels, down-sampling them, enhancing their local contrast, generating gNDVI, average filtering, and running the blob detection algorithm. This average time was calculated over 7 orthomosaic images. The average time for running the segmentation algorithm on each plot patch was 1.39 seconds. The average time of running the segmentation algorithm was calculated over 400 plot patches. Making gNDVI from the cropped patch, re-sampling the gNDVI, running K-means, running random walker, determining the mask size using pixel count of largest connected component, and finally the post-processing which includes morphological operations are the steps included in the segmentation phase. The evaluation of our algorithm had two aspects. First, to test the number of lentil plots correctly and incorrectly detected, second, to test the closeness of the automatically segmented plots with the segmentation drawn by the raters.

5.1 Results

5.1.1 Results of Detection Algorithm

The precision values for the 2 early season images were 0.971 and 0.968, and the recall values were 0.989 and 0.978. The average precision of the detection algorithm on early season image is 0.97 and average recall value is 0.984. For the 3 mid season images, the precision values were 0.99, 0.97, and 0.973; and recall values were 0.98, 0.983, and 0.978. The average precision and recall for the mid season images are 0.98. The precision and recall values for late season images are comparatively low. The precision values for 2 late season images were 0.93 and 0.94, the recall values were 0.965 and 0.95. Their average precision value is 0.94 and average recall value is 0.95. The average precision over 7 orthomosaic images was 0.963 and recall was 0.973. Table 5.1 shows the results of our detection algorithm with information of capturing camera, date and growth season.

Figure 5.1 and figure 5.2 shows that the detection algorithm worked for both Sony and Micasense camera images captured at different stages of the growing season. The figures show detected plots as black circles super-imposed over the plots. The figure is cropped portion from an original NDVI or gNDVI image. From the figures it can be seen that, our proposed detection algorithm could detect plots of various shapes and sizes.

We have also run our algorithm on other images captured by the Sony camera mounted on a UAV from different height and captured with a different angle than the actual setup used in this research. These images did not capture

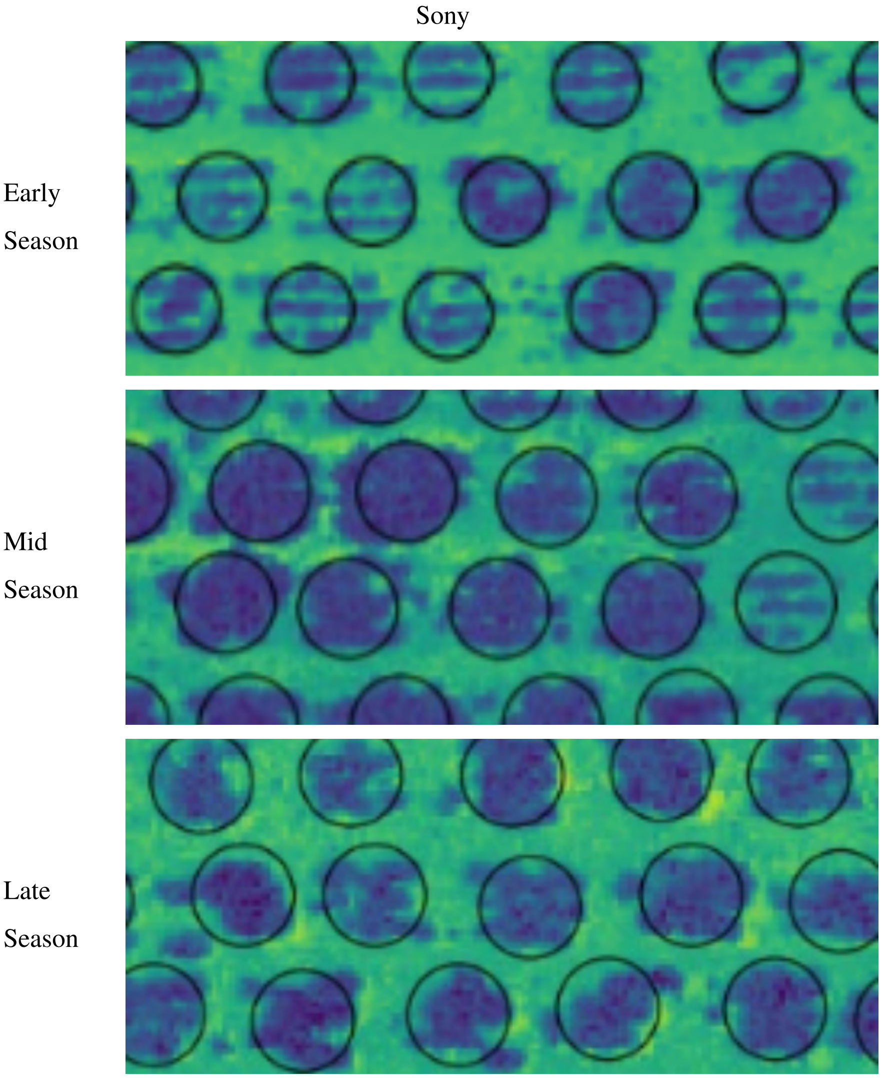


Figure 5.1: Output of detection algorithm, black circles overlaid on plots are detected using proposed algorithm. Image is a cropped gNDVI from Sony orthomosaic.

Table 5.1: Precision-Recall Table for Detection Algorithm

Growth Season of Image	**Capture Date**	**Camera**	**Precision**	**Recall**
Early Season	24 June 2017	Sony	0.971	0.989
Early Season	4 July 2017	Sony	0.968	0.978
Mid Season	13 July 2016	Micasense	0.99	0.98
Mid Season	18 July 2017	Sony	0.97	0.983
Mid Season	20 July 2017	Sony	0.973	0.978
Late Season	26 July 2017	Sony	0.93	0.965
Late Season	27 July 2016	Micasense	0.94	0.94
AVERAGE			0.963	0.973

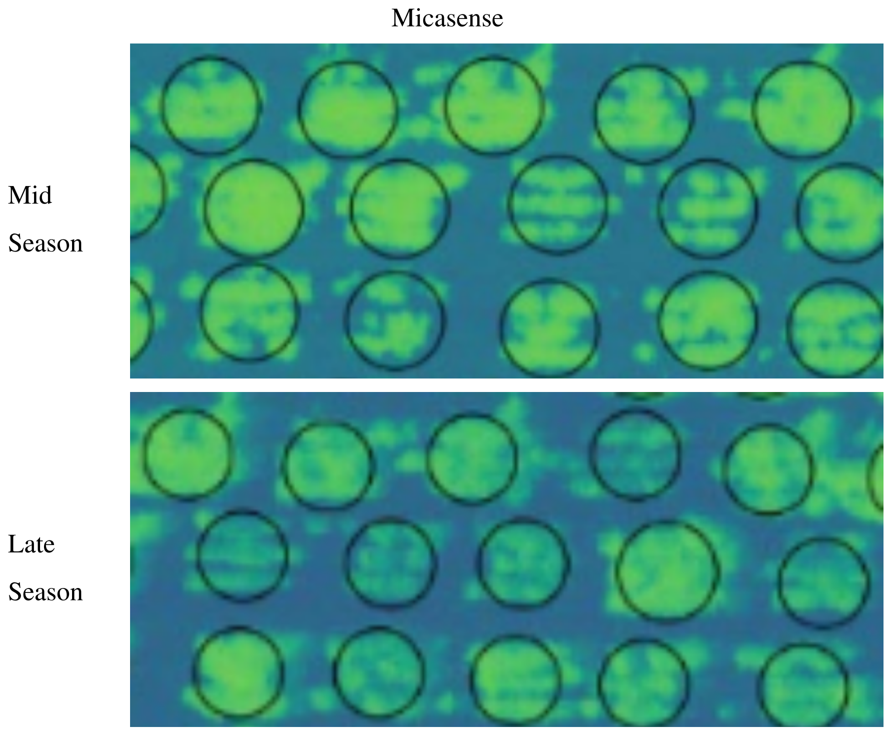


Figure 5.2: Output of detection algorithm, black circles overlaid on plots are detected using proposed algorithm. Image is a cropped NDVI from Micasense orthomosaic.

the entire lentil field rather it captured a certain portion of the field from lower height and from different angle. Our algorithm performed well in these images. The overall accuracy of our proposed detection algorithm sustained with average 97% precision and recall over 10 images. Figure 5.3 shows that our detection algorithm captured almost all the plots in these images as well.

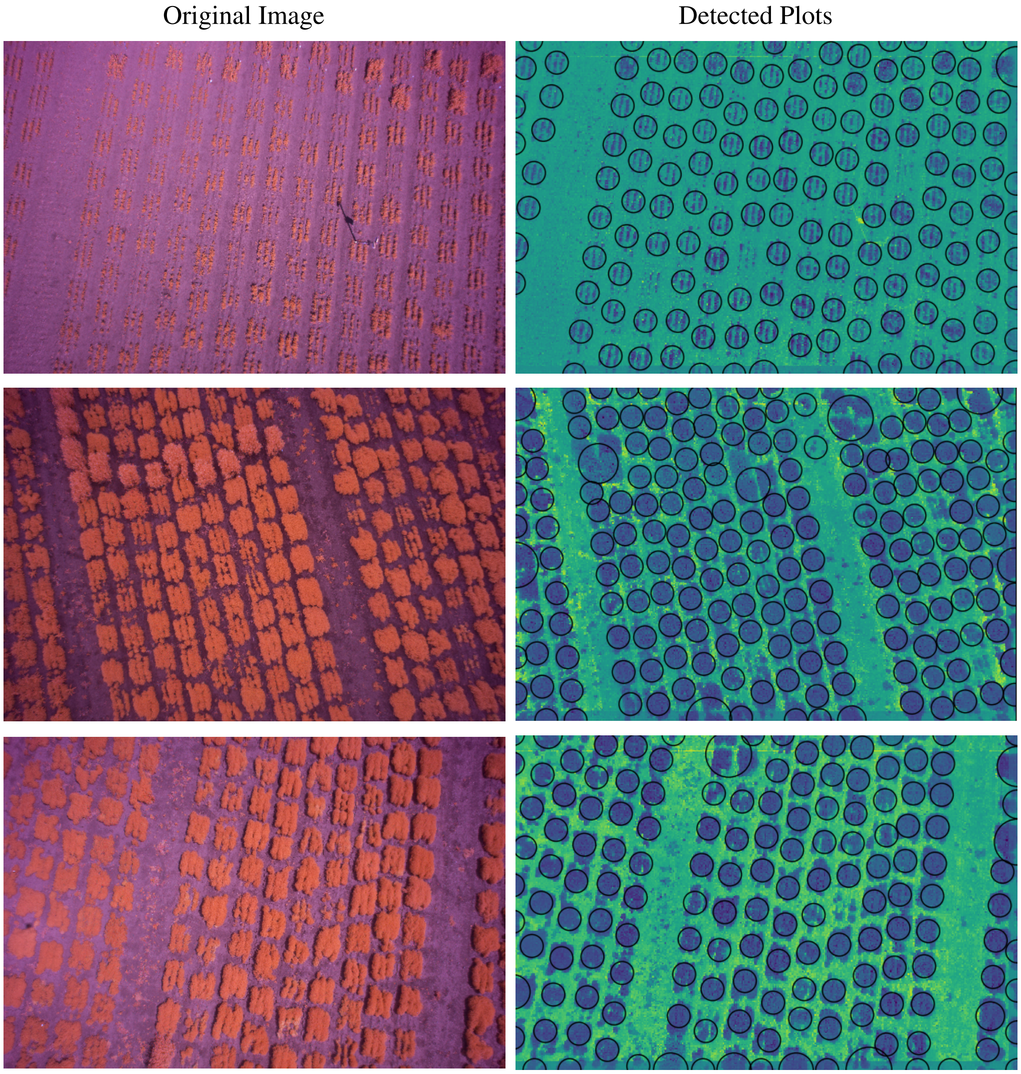


Figure 5.3: Output of detection algorithm on new images.

5.1.2 Results of Segmentation Algorithm

We evaluated the overall performance of the segmentation algorithm by measuring average Dice similarity coefficient (DSC), Mean squared distance (MSD), sensitivity, and specificity. When we designed our segmentation algorithm, we prioritized higher specificity rate while at the same time maintaining the DSC over 85%. We prioritized specificity because for plant phenotyping, it was preferable by the plant breeders to have segmentation include only plants at the cost of small part of the plot not being included compared to soil being included as part of the plot. The average DSC, MSD, sensitivity, and specificity for early season images were 0.876, 10.243, 0.863, and 0.974 respectively over 132 plot images. The average DSC, MSD, sensitivity and specificity for mid season images were 0.92, 6.875, 0.94, and 0.96 respectively over 135 plot images. The average DSC, MSD, sensitivity, and specificity for late season images were 0.89, 10.374, 0.93, and 0.95 respectively over 132 images.

Over all growth stages, the average DSC was 0.90, MSD was 8.13, sensitivity was 0.919, and specificity was 0.948 pixels which demonstrated the high accuracy of our proposed segmentation algorithm. These averages were made on individual 400 plot patches and these 400 patches were unique and randomly selected from different lentil fields captured in different growth stages. The table 5.2 shows the result of our segmentation algorithm.

Table 5.2: Results of the Segmentation Algorithm

Growth Season	**Capture Date**	**Camera**	**No. Plots**	**Avg. DSC**	**Avg. MSD**	**Avg. Sens.**	**Avg. Spec.**
Early	24 June 2017	Sony	66	0.899	8.556	0.886	0.966
Early	4 July 2017	Sony	66	0.856	11.93	0.840	0.982
Mid	13 July 2016	Micasense	45	0.905	8.086	0.915	0.967
Mid	18 July 2017	Sony	45	0.925	6.669	0.940	0.956
Mid	20 July 2017	Sony	46	0.92	5.87	0.95	0.946
Late	26 July 2017	Sony	66	0.89	9.01	0.943	0.937
Late	27 July 2016	Micasense	66	0.889	11.738	0.912	0.957
AVERAGE				0.898	8.837	0.912	0.959

5.2 Discussion

5.2.1 Discussion of Detection Results

From the result of our detection algorithm shown in table 5.1, it is apparent that our detection algorithm works well and performs consistently over all the growth seasons of the lentils when the images were captured with any of the two multi-spectral cameras. No image had a precision lower than 93% or recall lower than 96%.

The highest overall accuracy of the proposed detection algorithm was for a mid-season image captured in 2016 having precision and recall of 99% and 98% respectively. The average precision and recall for mid season images was better than for early or late season images, which can be seen in figure 5.4. However, the detection accuracy for images captured in the late growth season of lentils was less than both the mid and early-season images. The reason for worse detection accuracy in late season images was the overlapped growth of neighboring plots. It is often in late season images that lentils grow large and expand and the plots overlap, making it very hard for the detection algorithm to detect each lentil plot individually. The average precision and recall for late season images was 94% and 95.3% respectively. The precision and recall were 94% for the latest season image captured in 2016 and with Micasense

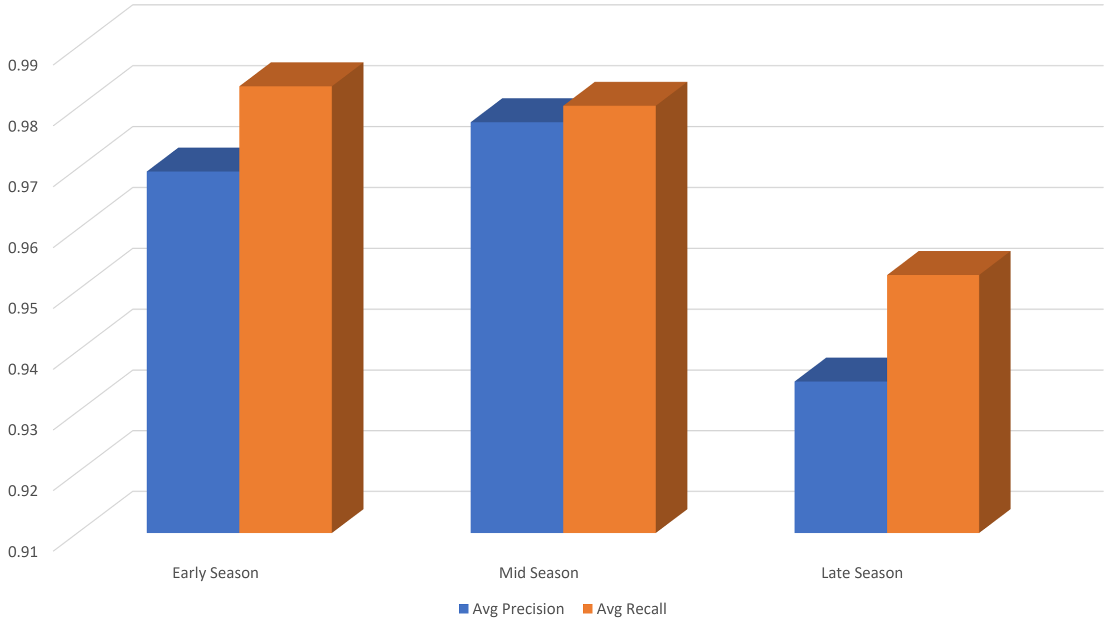


Figure 5.4: Bar plot of average Precision and Recall for different growth stages of images.

camera, this was the lowest accuracy among all the available orthomosaic images from every growth season. If the late season images are not considered, the overall accuracy of our detection algorithm is increased. The average precision is 97% and recall is 98% for early and mid season images. The average precision and recall may well be above 97% for a specific growth season but the accuracy for detection failing to capture 100% of the plots will not let the plant breeders track all the 1410 lentil plots. But with 97% accuracy, the plot monitoring becomes mostly automated and looking at the image of detected plots, the non-detected plots can be easily identified and can be monitored in person.

The 98% plot detection accuracy was sustained for images captured by Sony camera from a different height and angle. The camera being used for capturing images from different height and angle made the plots appear different from the setup used in this research. Angled lower height of the camera make the plots closer to the camera look large and the plots distant from camera look smaller than the actual size. But our proposed detection algorithm successfully detected most of the plots.

Figure 5.5 shows the PR curve of our proposed detection algorithm for different thresholds. This threshold is the absolute lower bound for scale space maxima. Table 4.1 shows the threshold we used for our detection algorithm. The PR curve will allow us to show the rate of detected plots that are relevant plots and the rate of detected plots to all relevant plots. We have ignored any extrema below 0.64 for precision in the curve because below that value the curve does not change. From the figure 5.5, it can be seen that the curve is steep and flat around 0.97 and 0.98 in precision axis against $1 - \text{recall}$ axis which is the x-axis. Moreover, a flat curve constantly nearing the value 1 in a PR curve means all the objects identified are consistently relevant as precision is the number of true positives over the

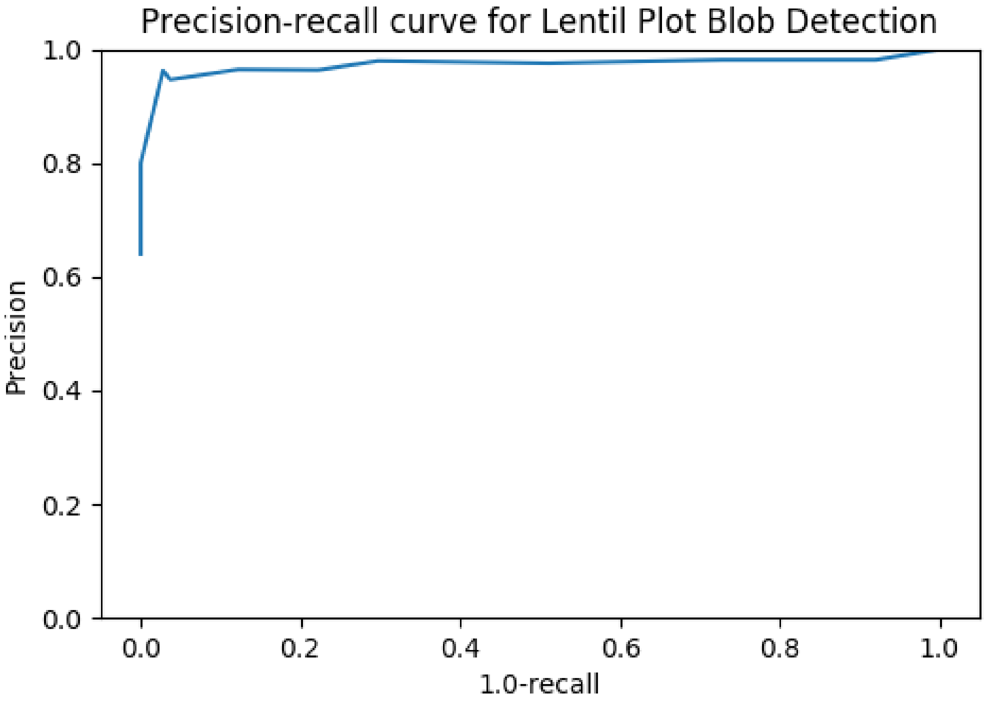


Figure 5.5: PR curve for different threshold values of blob detection of lentil plots.

summation of true positives and false positives; and recall is true positives over the summation of true positives and false negatives. So the curve being flat means all the plots captured are mostly relevant which indicates a very good and consistent classification.

Our proposed detection algorithm works for images captured in any season of the growth stage of the lentils. But sometimes, in the late season images it can be seen that quite a few plots have come closer to each other which made the detection harder. The Laplacian of Gaussian kernel while scanning like a sliding window, excels at the location of blob like regions and captures everything which falls within the kernel radius. If a plot is connected with another one, and one of the connected plots have less intensity and does not fit within the Gaussian kernel radius, then our detection algorithm cannot detect that plot. Such scenario is shown on figure 5.6.

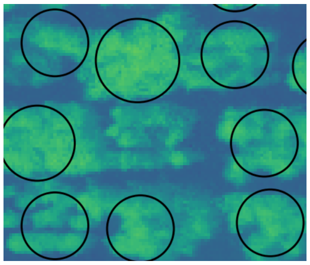


Figure 5.6: Scenerio when the algorithm failed to detect individual plot accurately. The plot in the middle, having no black circle indicates undetected plot.

5.2.2 Discussion of Segmentation Results

From figure 5.8 it is clear that, our segmentation algorithm works constantly well for the plot patches acquired from different times during the growth season. The average DSC, sensitivity and specificity was always more than 0.90, 0.91 and 0.94 respectively for mid and late season images. The highest segmentation accuracy was obtained for mid-season images. Because, in mid season, the plots are compact and well shaped and easier to segment. The accuracy for mid-season images was 0.91 average DSC, 0.96 average specificity, and 0.93 average sensitivity. Such sensitivity is good enough for bulk analysis of plots but 96% specificity means 4% of the plant pixels are labeled as background which may affect any further analysis but not much as non-plant pixels would lie on the boundary of the plots. We could not obtain good results for early season images because in early season, the plots are not matured enough, very little plant appears beyond soil, the color is almost the same as soil making it harder to segment them. Figure 5.7 shows such a case. The white boundary overlaid on the plot region indicates the segmentation boundary obtained using our algorithm. Our segmentation algorithm was unable to segment the yellow marked region. The region inside the yellow boundary is disjoint from the main plot region and is not matured enough to be recognized as part of the plot region. But the loosely connected plot region below the segmented plot was possible to include within the segmentation as that part is close to the main plot region and the morphological operation could easily capture it within the plot region.

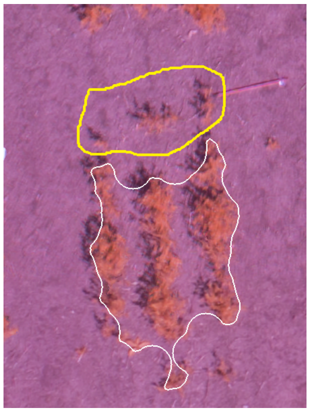


Figure 5.7: Scenerio when the algorithm failed to completely segment a plot.

Figure 5.9 shows our segmentation results. The results in the figure are shown for 5 different plots. The first 3 plots were captured by Sony camera and from 3 different growth stages. The last 3 plot images were captured by Micasense camera from 2 different growth stages. The plots shown in the figure are plots whose segmentation accuracy was best among all the plots. In the figure, the first segmentation was on early season plot image. As it can be seen, the plant pixels are scattered, not bound together which made challenging to segment images from early season images. The mid and late season images were comparatively easy to segment. As you will see on the figure 5.9, the mid and late season plots are compact and easy to segment but the problem with late season plots is they are very close to neighboring plots making it harder to segment individual plots.

In conclusion, our algorithm detects 96% of the plots accurately and also helps to identify the physical location of non-detected plots and segments each of the plot with more than 95% accuracy. Moreover, our algorithm detects

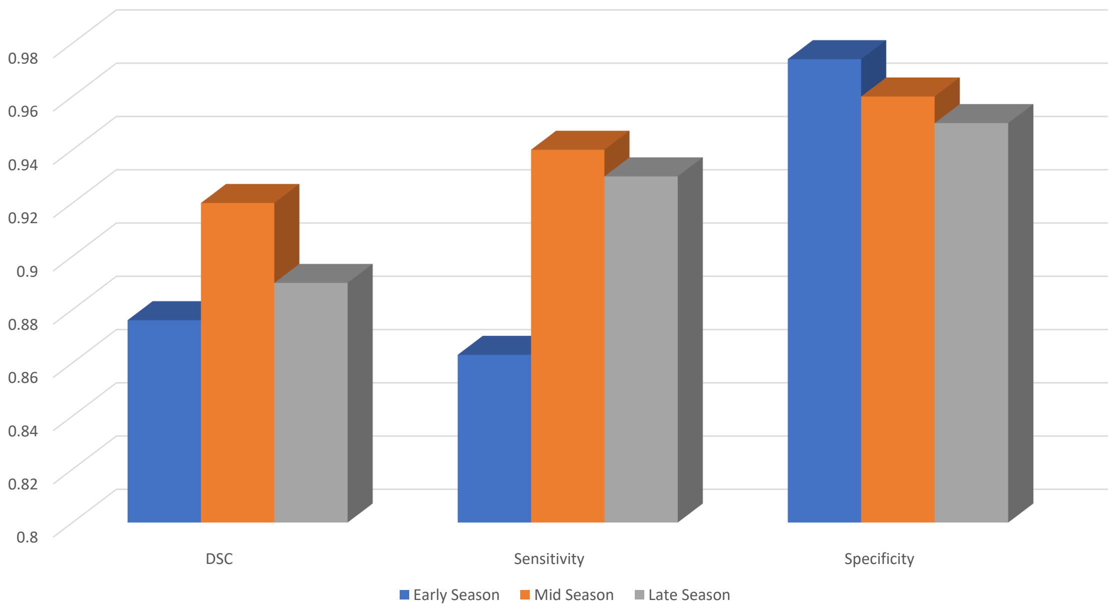


Figure 5.8: Average DSC, Sensitivity and Specificity of plot images from 3 growth stages.

and segments all the plots from a single multi-spectral image with less than couple of seconds to gather bulk of plots altogether for phenotypic analysis by plant breeders thus our proposed method will completely automate the process of monitoring the lentil plots though some scope of improvement in the method still exists.

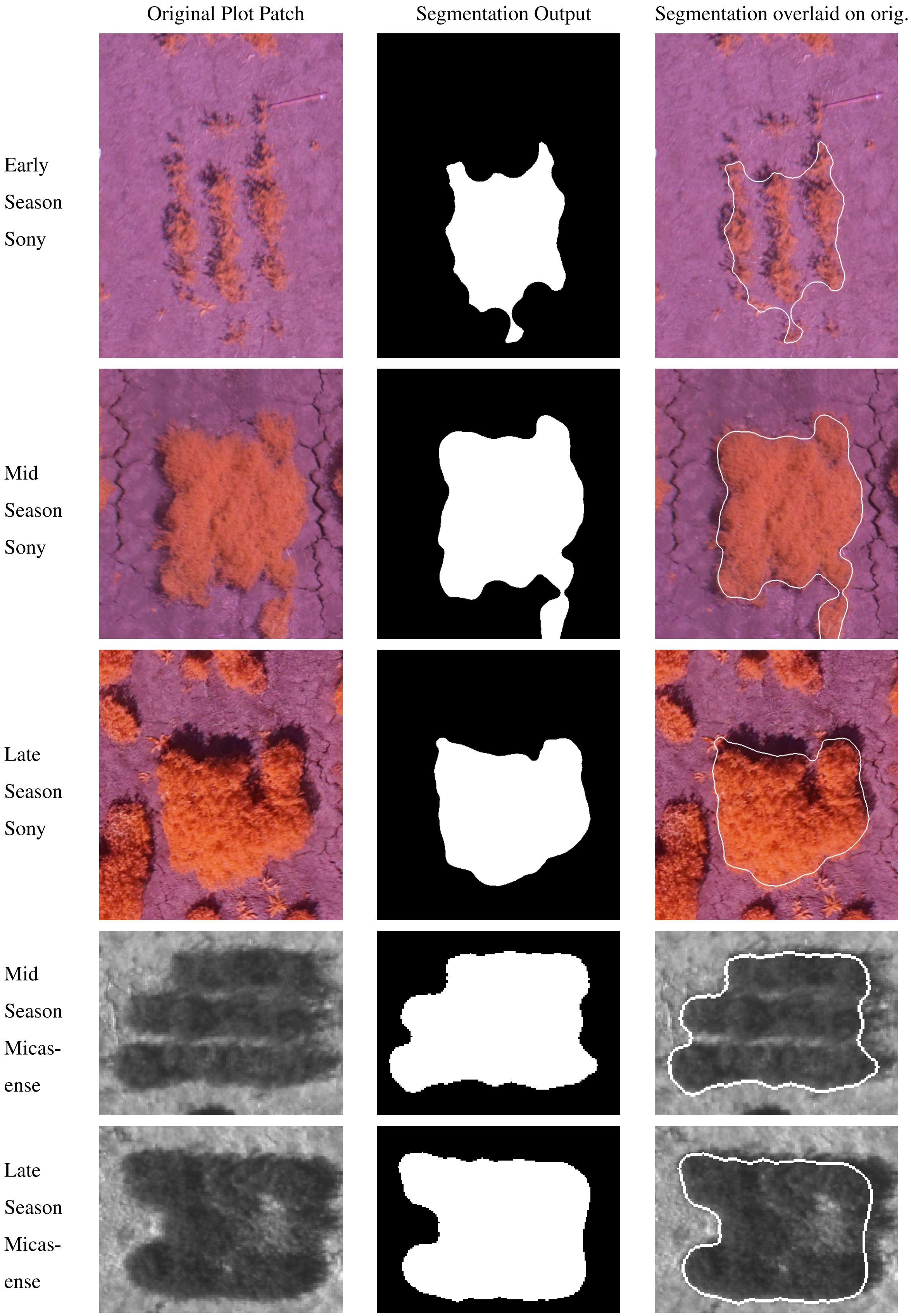


Figure 5.9: Segmentation Results of plots from 3 different seasons and 2 capturing cameras.

6 Conclusion

We have proposed an automatic detection algorithm based on blob detection and an automatic segmentation algorithm based on K-means clustering and random walker from aerial images captured by multi-spectral cameras mounted on drone. We derived contrast enhanced gNDVI from input images so that we have the apparent live green vegetation for better post-processing. Later, we processed the gNDVI to detect plots as blobs and segment cropped patches obtained using blob ratio. This entire process takes only 1.003 seconds on average. Our proposed method is the first detection and segmentation solution for lentil plots in aerial images. Our proposed detection method performed exceptionally well for images captured both in early and mid season of the crop's growth period. And our proposed segmentation method worked well for all 3 growth season but performed best for mid-season images. The advantage of our algorithm is that the algorithm works for multi-spectral images independent of the capturing camera and is an algorithm which works for image captured in every growth season of lentils. Moreover, our algorithm is unsupervised and requires no training to detect and segment the lentil plots. Automation of detection and segmentation of different lentil plots reduces the amount of time spent by plant breeders in the field to collect phenotypic information. Manually collecting a single trial in large research trials involves hundreds of man hours. Moreover, manual data collection is tedious, monotonous, and boring. Our algorithm is totally automatic and requires no human intervention.

The overall performance of our proposed detection algorithm depends on the maturity of the lentil plots. In this research, we have considered images captured with different angle and height other than the aerial images which were used to train and test the model. Our algorithm performed well for those images as well.

Since, gNDVI is commonly acquired in many breeding programs, our detection and segmentation method should be effective for other crops as well if the crops are blob-like in appearance. Our algorithm can detect and segment aerial lentil plot images with any number of channels of any multi-spectral camera if the camera captures the required channels to make NDVI or gNDVI. With the growing need of remote sensing and absence of detection and segmentation technique for automatically detecting and segmenting lentil plots from images, our proposed method will be proven most desired tool for helping in monitoring phenotypes of lentil breeding plots. This research of ours have already contributed to the literature as our work has been published in WACV 2019.

6.1 Future Work

Some of the potential future works of our proposed method are mentioned below:

- In later growth stage of lentils, the plots expand too much and grow together with neighboring plots. Though our detection algorithm can detect them uniquely, cropping them using blob ratio leads to inclusion of neighboring plot portions within the cropped patch. In future, this kind of scenario can be dealt with watershed segmentation.
- Our algorithm cannot distinguish between guard plots, which are in between two trials of lentils . In future, distinguishing between guard plot and lentil plot can be done.
- Sometimes, our detection algorithm failed to detect individual plots when two or three plots expanded so much and grew together. In that case, geospatial information can be used to uniquely identify each plot.
- Our algorithm works in a region of interest specified by user. In future, automatically determining the region of interest can be done and our proposed approach can be applied in other kind of crop images as well.

REFERENCES

[1] Eric Atkins. Saskatchewan takes lead in pulse crop exports as India's appetite grows, The Globe and Mail, April 2015.

[2] Oscar Barrero, Diana Rojas, Christian Gonzalez, and Sammy Perdomo. Weed detection in rice fields using aerial images and neural networks. In *Signal Processing, Images and Artificial Vision (STSIVA), 2016 XXI Symposium on*, pages 1–4. IEEE, 2016.

[3] Yuri Boykov and Gareth Funka-Lea. Graph cuts and efficient nd image segmentation. *International journal of computer vision*, 70(2):109–131, 2006.

[4] William R Crum, Oscar Camara, and Derek LG Hill. Generalized overlap measures for evaluation and validation in medical image analysis. *IEEE transactions on medical imaging*, 25(11):1451–1461, 2006.

[5] Huawu Deng and David A Clausi. Unsupervised image segmentation using a simple MRF model with a new implementation scheme. *Pattern recognition*, 37(12):2323–2335, 2004.

[6] Nameirakpam Dhanachandra, Khumanthem Manglem, and Yambem Jina Chanu. Image segmentation using k-means clustering algorithm and subtractive clustering algorithm. *Procedia Computer Science*, 54(2015):764–771, 2015.

[7] H Digabel and Christian Lantuéjoul. Iterative algorithms. In *Proc. 2nd European Symp. Quantitative Analysis of Microstructures in Material Science, Biology and Medicine*, volume 19, page 8. Stuttgart, West Germany: Riederer Verlag, 1978.

[8] M-P Dubuisson-Jolly and Alok Gupta. Color and texture fusion: application to aerial image segmentation and gis updating. *Image and Vision Computing*, 18(10):823–832, 2000.

[9] Ch Gée, J Bossu, G Jones, and F Truchetet. Crop/weed discrimination in perspective agronomic images. *Computers and Electronics in Agriculture*, 60(1):49–59, 2008.

[10] Leo Grady. Multilabel random walker image segmentation using prior models. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, pages 763–770. IEEE, 2005.

[11] Leo Grady. Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 28(11):1768–1783, 2006.

[12] Leo Grady, Thomas Schiwietz, Shmuel Aharon, and Rüdiger Westermann. Random walks for interactive alpha-matting. In *Proceedings of VIIP*, volume 2005, pages 423–429, 2005.

[13] Leo Grady, Thomas Schiwietz, Shmuel Aharon, and Rüdiger Westermann. Random walks for interactive organ segmentation in two and three dimensions: Implementation and validation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 773–780. Springer, 2005.

[14] José Miguel Guerrero, María Guijarro, Martin Montalvo, Juan Romeo, Luis Emmi, Angela Ribeiro, and Gonzalo Pajares. Automatic expert system based on images for accuracy crop row detection in maize fields. *Expert Systems with Applications*, 40(2):656–664, 2013.

[15] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, pages 10–5244. Citeseer, 1988.

[16] A. K. Jain. *Fundamentals of Digital Image Processing*. Englewood Cliffs, NJ, 1989.

[17] Anil K Jain and Farshid Farrokhnia. Unsupervised texture segmentation using gabor filters. *Pattern recognition*, 24(12):1167–1186, 1991.

[18] Hassana Grema Kaganami and Zou Beiji. Region-based segmentation versus edge detection. In *Intelligent Information Hiding and Multimedia Signal Processing, 2009. IIH-MSP'09. Fifth International Conference on*, pages 1217–1221. IEEE, 2009.

[19] Hui Kong, Hatice Cinar Akakin, and Sanjay E Sarma. A generalized laplacian of gaussian filter for blob detection and its applications. *IEEE transactions on cybernetics*, 43(6):1719–1733, 2013.

[20] Tony Lindeberg. Feature detection with automatic scale selection. *International journal of computer vision*, 30(2):79–116, 1998.

[21] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[22] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[23] Eduardo A Murillo-Bracamontes, Miguel E Martinez-Rosas, Manuel M Miranda-Velasco, Horacio L Martinez-Reyes, Jesus R Martinez-Sandoval, and Humberto Cervantes-de Avila. Implementation of hough transform for fruit image segmentation. *Procedia Engineering*, 35:230–239, 2012.

[24] HP Ng, SH Ong, KWC Foong, PS Goh, and WL Nowinski. Medical image segmentation using k-means clustering and improved watershed algorithm. In *Image Analysis and Interpretation, 2006 IEEE Southwest Symposium on*, pages 61–65. IEEE, 2006.

[25] HP Ng, SH Ong, KWC Foong, and WL Nowinski. An improved watershed algorithm for medical image segmentation. In *Proceedings 12th International Conference on Biomedical Engineering*, 2005.

[26] Vikramsingh R Parihar and Nileshsingh V Thakur. Graph theory based approach for image segmentation using wavelet transform. *International Journal of Image Processing (IJIP)*, 8(5):255–277, 2014.

[27] Matti Pietikäinen, Azriel Rosenfeld, and Ingrid Walter. Split-and-link algorithms for image segmentation. *Pattern Recognition*, 15(4):287–298, 1982.

[28] Stephen M Pizer, E Philip Amburn, John D Austin, Robert Cromartie, Ari Geselowitz, Trey Greer, Bart ter Haar Romeny, John B Zimmerman, and Karel Zuiderveld. Adaptive histogram equalization and its variations. *Computer vision, graphics, and image processing*, 39(3):355–368, 1987.

[29] Stephen M Pizer, R Eugene Johnston, James P Ericksen, Bonnie C Yankaskas, and Keith E Muller. Contrast-limited adaptive histogram equalization: speed and effectiveness. In *Visualization in Biomedical Computing, 1990., Proceedings of the First Conference on*, pages 337–345. IEEE, 1990.

[30] Dan Popescu, Loretta Ichim, and Traian Caramihale. Flood areas detection based on uav surveillance system. In *System Theory, Control and Computing (ICSTCC), 2015 19th International Conference on*, pages 753–758. IEEE, 2015.

[31] Gilles Rabatel, Carole Delenne, and Michel Deshayes. A non-supervised approach using gabor filters for vine-plot detection in aerial images. *Computers and electronics in Agriculture*, 62(2):159–168, 2008.

[32] Richard E. Woods Rafael C. Gonzalez. *Digital Image Processing*. Prentice Hall, second edition.

[33] Ali M Reza. Realization of the contrast limited adaptive histogram equalization (clahe) for real-time image enhancement. *Journal of VLSI signal processing systems for signal, image and video technology*, 38(1):35–44, 2004.

[34] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European conference on computer vision*, pages 430–443. Springer, 2006.

[35] Saskatchewan.ca. Crop planning guide and crop planner (saskatchewan), October 2018.

[36] Frank Y Shih and Shouxian Cheng. Automatic seeded region growing for color image segmentation. *Image and vision computing*, 23(10):877–886, 2005.

[37] Ali Kemal Sinop and Leo Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007.

[38] Beril Sirmacek and Cem Unsalan. A probabilistic framework to detect buildings in aerial and satellite images. *IEEE Transactions on Geoscience and Remote Sensing*, 49(1):211–221, 2011.

[39] John Smith and S-F Chang. Quad-tree segmentation for texture-based image query. In *Proceedings of the second ACM international conference on Multimedia*, pages 279–286. ACM, 1994.

[40] J Alex Stark. Adaptive image contrast enhancement using generalizations of histogram equalization. *IEEE Transactions on image processing*, 9(5):889–896, 2000.

[41] R. L. Rivest C. Stein T. Cormen, C. E. Leiserson. *Introduction to Algorithms*. Carnegie Mellon University, 2nd edition, September 2001.

[42] C Unsalan. Gradient-magnitude-based support regions in structural land use classification. *IEEE Geoscience and Remote Sensing Letters*, 3(4):546–550, 2006.

[43] M Arif Wani and Bruce G. Batchelor. Edge-region-based segmentation of range images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(3):314–319, 1994.

[44] Kyosuke Yamamoto, Wei Guo, Yosuke Yoshioka, and Seishi Ninomiya. On plant detection of intact tomato fruits using image analysis and machine learning methods. *Sensors*, 14(7):12191–12206, 2014.

[45] Luh Yen, Denis Vanvyve, Fabien Wouters, François Fouss, Michel Verleysen, and Marco Saerens. clustering using a random walk based distance measure. In *ESANN*, pages 317–324, 2005.

[46] Qingbing Zeng, Yu-bin Miao, Chengliang Liu, and Shiping Wang. Algorithm based on marker-controlled watershed transform for overlapping plant fruit segmentation. *Optical Engineering*, 48(2):027201, 2009.