From Pixels to Pathogens: A MATLAB Based Approach to Plant Disease Identification

L Shanmukha Sai Srivalli, Annem Venkareddy
1,2 Undergraduate Student
1,2 VIT-AP University

ABSTRACT: Agriculture plays a crucial role in feeding the growing population, particularly in Asia where over 70% of the population depends on it. However, crop quality and quantity can be significantly affected by diseases, leading to agricultural losses. Detecting diseases early is essential to prevent such losses. This project aims to develop a software solution that can automatically detect and classify plant diseases. The process involves loading an image, pre-processing it, segmenting it to isolate the relevant parts, extracting features, and finally classifying the disease based on the features. This software can be particularly useful in detecting diseases in plant leaves. The use of image processing techniques in agriculture can help to improve crop yields, prevent losses, and ensure food security.

INTRODUCTION:

Agriculture is the backbone of our country, and farmers carefully select crops to maximize their profits and yields. However, to ensure standard production, scientific techniques must be employed. This can be achieved through technical assistance to help manage diseases that can significantly impact crop yields and profitability. Image processing is an excellent technique for detecting plant diseases in agricultural applications. However, disease identification is often restricted by human visual capabilities, making it a tedious and time-consuming process. Therefore, there is a need for an automated system that can recognize, classify, and quantitatively detect disease symptoms.

Diseases are typically detected on the leaves or stems of plants, and their quantification can be challenging due to the complexity of visual patterns. Developing a computer vision system to identify and classify diseased crops can help farmers make unbiased decisions about disease infection and valuation. This system can also help to avoid the need for human intervention and provide accurate and efficient detection of diseases.

In this project, digital high-resolution images of healthy and unhealthy plants are captured and pre-processed for image enhancement. The captured leaf images are then segmented using the k-means cluster technique to create clusters, and features are extracted before applying the K-means and Random Forest Classifier for training and classification. The diseases can then be recognized using this method.

This paper provides introduction to the importance of disease detection in agriculture, followed by a brief overview of plant diseases. A literature review of leaf disease detection techniques is provided, and the methodology of the proposed system based on MATLAB image processing is described. The results and discussion are presented, and the paper concludes with future work.

METHODOLOGY:

- IMAGE ACQUISITION
- IMAGE PRE-PROCCESSING
- IMAGE SEGMENTATION
- FEATURE EXTRACTION
- CLASSIFICATION

1. Preprocessing: The first step is to preprocess the image to remove noise and enhance contrast. This can involve techniques such as image filtering, histogram equalization, and contrast stretching.

2. Segmentation: The next step is to segment the image to identify the regions of interest (ROI), such as the leaf and the disease spots. This can involve techniques such as thresholding, region growing, and edge detection.

3. Feature extraction: Once the ROI has been identified, relevant features can be extracted from the image. Some common features used for plant disease detection include color-based features, texture-based features, and shape-based features. For example, color-based features can include the mean and standard deviation of color channels such as red, green, and blue. Texture-based features can include statistical measures such as entropy and contrast, while shape-based features can include measures such as area, perimeter, and circularity.

4. Feature selection: After extracting the features, it is important to select the most relevant features that can discriminate between different classes of images. This can involve techniques such as principal component analysis (PCA) or linear discriminant analysis (LDA).

5. Classification: Finally, the selected features can be used to train a classifier such as a support vector machine (SVM) or a random forest (RF) to classify the images into different categories such as alternaria alternata, bacterial blight, antracnose, cercospora leaf spot, or healthy.
In MATLAB, there are several built-in functions and toolboxes that can be used for feature extraction and classification, such as the Image Processing Toolbox, the Computer Vision Toolbox, and the Statistics and Machine Learning Toolbox. These tools can greatly simplify the process of detecting plant diseases from leaf images and provide a wide range of options for feature extraction and classification.

1. IMAGE ACQUISITION:
   - The initial step of digital image processing is the loading of an image, which involves capturing an image using a digital camera and saving it in a digital format for subsequent processing using MATLAB. This can also refer to the process of retrieving an image from hardware for further processing. In our project, we captured images of both healthy and diseased leaves and fruits using a camera, as depicted in figure, to be used in our MATLAB-based image processing system.

2. IMAGE PREPROCESSING:
   - The primary goal of image pre-processing is to improve the quality of an image by removing any unwanted distortions and enhancing specific features before further processing. Various techniques are employed in pre-processing such as changing the size and shape of the image, filtering out noise, converting images, enhancing image quality, and performing morphological operations. In this particular project, MATLAB code was used to modify the size of the image, enhance contrast, and convert RGB images to grayscale. These pre-processing techniques were applied to the captured images, as shown in figures, in order to prepare them for further operations such as segmentation and cluster creation.
   1. MEDIAN FILTERING
   2. HISTOGRAM EQUALIZATION
   3. HSI IMAGE - Hue, Saturation, and Intensity

3. IMAGE SEGMENTATION
   - In digital image processing, segmentation involves dividing a digital image into multiple parts or regions for easier analysis. It helps in identifying the objects and boundaries of the image. To achieve this, the K-means clustering technique is commonly used. In this method, the image is partitioned into clusters, with at least one cluster containing a significant area of the unhealthy portion of the image. The algorithm aims to classify the objects into a specified number of categories based on their features, while minimizing the total square of distances between the data entities and the respective clusters. To carry out this segmentation, the RGB color space is converted to the L*a*b* color space, which consists of a luminosity layer 'L*', chromaticity 'a*', and 'b*'.

4. FEATURE EXTRACTION

Feature extraction in image processing is the process of selecting and extracting relevant information (features) from an image for further analysis or classification. In the context of disease detection in leaves, feature extraction is an important step that involves selecting informative features that can distinguish between healthy and diseased leaves or between different types of diseases. There are many different types of features that can be extracted from images for disease detection, depending on the specific requirements of the problem. Some common features used in disease detection in leaves include:
1. Color-based features: The color of a leaf can provide important information about its health status. Color-based features can be extracted from an image by converting it to different color spaces (e.g., RGB, HSV, or LAB) and analyzing the distribution of colors.
2. Texture-based features: The texture of a leaf can also provide important information about its health status. Texture-based features can be extracted by analyzing the spatial arrangement of pixels in an image, such as the frequency of edges or the distribution of gradients.
3. **Shape-based features:** The shape of a leaf can also be indicative of its health status. Shape-based features can be extracted by analyzing the contour of the leaf or its geometric properties, such as its area, perimeter, or circularity.

4. **Statistical features:** Statistical features can be extracted by analyzing the distribution of pixel intensities in an image, such as the mean, standard deviation, or entropy.

**5. Classification**

- So, if a image of leaf with disease is uploaded in matlab and run the code, we find a part of the leaf highlighted in dark color which represents that the leaf is suffering with a disease.

**Examples:**

1. Alternaria alternata
2. Bacterial Blight
3. Antracnose
4. Cercospora Leaf Spot

- And if an image of leaf without disease is uploaded in matlab and run the code, we find entire part of the leaf in the even color which represents that the leaf is not suffering with any kind of disease.

- In this way, classification of the images of leaves with and without diseases is done.

**MATLAB CODE FOR IMAGE PRE-PROCESSING:**

```matlab
% Load the image
originalImage = imread('normalleaf.jpg');
% Convert the image to grayscale
grayImage = rgb2gray(originalImage);
% Apply a median filter to remove noise
medianFiltered = medfilt2(grayImage);
% Apply histogram equalization to enhance contrast
eqImage = histeq(medianFiltered);
% Convert the image to HSI format
hsiImage = rgb2hsv(originalImage);
% Display the original and processed images
subplot(2, 2, 1);
imshow(originalImage);
title('Original Image');
subplot(2, 2, 2);
imshow(medianFiltered);
title('Median Filtered');
subplot(2, 2, 3);
imshow(eqImage);
title('Histogram Equalization');
subplot(2, 2, 4);
imshow(hsiImage);
title('HSI Image');
```

**EXPLANATION OF CODE**

**IMAGE PRE-PROCESSING**

This MATLAB code performs a series of image pre-processing techniques on the input image 'normalleaf.jpg'. The steps involved in the code are as follows:

- The input image is loaded using the 'imread' function and assigned to a variable called 'originalImage'.
- The 'rgb2gray' function is used to convert the original color image to grayscale and the resulting image is stored in a variable called 'grayImage'.
- A median filter is applied to remove any noise present in the grayscale image. The 'medfilt2' function is used to apply the median filter to the image and the resulting image is stored in a variable called 'medianFiltered'.
- Histogram equalization is applied to enhance the contrast of the image. The 'histeq' function is used to apply the histogram equalization to the median filtered image and the resulting image is stored in a variable called 'eqImage'.
- The 'rgb2hsv' function is used to convert the original color image to the HSI format and the resulting image is stored in a variable called 'hsiImage'.
Finally, the original image and the pre-processed images (median filtered, histogram equalized, and HSI) are displayed using the 'subplot' and 'imshow' functions.

In summary, this MATLAB code performs various pre-processing techniques to enhance the input image for further analysis.

**MATLAB CODE FOR IMAGE SEGMENTATION:**

```matlab
% Load the image
img = imread('damagedleaf.jpg');
% Convert from RGB color space to L*a*b* color space
lab_img = rgb2lab(img);
% Reshape the image into a 2D array of pixels
lab_pixels = reshape(lab_img, [], 3);
% Perform K-means clustering to segment the image
k = 2; % number of clusters
[idx, centers] = kmeans(lab_pixels, k);
% Reshape the clustering indices to the original image size
segmented = reshape(idx, size(lab_img, 1), size(lab_img, 2));
% Display the segmented image
imshow(segmented, []);
% Color the segmented regions with different colors
segmented_img = zeros(size(img));
for i = 1:k
    mask = repmat(segmented == i, [1 1 3]);
    color = repmat(centers(i, :), [sum(mask(:)) 3 1]);
    segmented_img(mask) = color(:);
end
% Display the colored segmented image
imshow(segmented_img);
```

**EXPLANATION OF CODE (IMAGE SEGMENTATION)**

This MATLAB code performs image segmentation using the K-means clustering technique. The steps involved in this process are as follows:

- The input image 'image.jpg' is loaded using the 'imread' function and stored in the variable 'img'.
- The 'rgb2lab' function is used to convert the RGB color space of the input image into the L*a*b* color space. This is stored in the variable 'lab_img'.
- The 'reshape' function is used to convert the 3D 'lab_img' array into a 2D array of pixels, stored in 'lab_pixels'.
- The 'kmeans' function is used to perform K-means clustering on the pixel values. The number of clusters is set to 'k=2'. The resulting clustering indices are stored in 'idx', and the cluster centroids are stored in 'centers'.
- The clustering indices 'idx' are reshaped to the size of the original image using the 'reshape' function, and stored in 'segmented'.
- The segmented image is displayed using the 'imshow' function.
- The segmented regions are colored differently using a loop that iterates over the number of clusters. For each cluster, a binary mask is created for the corresponding segmented region, which is then used to select the pixel values of the color of the corresponding cluster centroid. The colored segmented image is stored in 'segmented_img'.
- The colored segmented image is displayed using the 'imshow' function.

Overall, this code segment uses K-means clustering to partition an input image into segments, and then colors the segments with different colors for visualization.

**MATLAB CODE TO DETECT ALTERNARIA ALTERNATA DISEASE:**
MATLAB CODE TO DETECT BACTERIAL BLIGHT DISEASE:

```matlab
% Load leaf image
im = imread('bacterialblight.jpg');
% Convert image to grayscale
imGray = rgb2gray(im);
% Preprocess image to enhance contrast and remove noise
imPreprocessed = imadjust(imGray);
% Segment image using Otsu's method
level = graythresh(imPreprocessed);
imSegmented = imbinarize(imPreprocessed, level);
% Remove small objects from binary image
imCleaned = bwareaopen(imSegmented, 100);
% Calculate the Euler number of the binary image
eulerNumber = bweuler(imCleaned);
% Threshold Euler number to identify bacterial blight disease
if eulerNumber < 1
    disp('Bacterial blight disease detected');
else
    disp('Leaf is healthy or has a different disease');
end
% Display original image with binary image overlay
overlay = imoverlay(im, imCleaned, [1 0 0]);
figure;
imshow(overlay);
```
MATLAB CODE TO DETECT ANTRACNOSE DISEASE:

```matlab
% Load leaf image
im = imread('antracnose.jpg');
% Convert image to grayscale
imGray = rgb2gray(im);
% Preprocess image to enhance contrast and remove noise
imPreprocessed = imadjust(imGray);
% Segment image using Otsu's method
level = graythresh(imPreprocessed);
imSegmented = imbinarize(imPreprocessed, level);
% Remove small objects from binary image
imCleaned = bwareaopen(imSegmented, 100);
% Dilate image to connect close spots or lesions
se = strel('disk', 5);
imDilated = imdilate(imCleaned, se);
% Calculate the ratio of white pixels in the image
imSize = size(imDilated);
numPixels = imSize(1) * imSize(2);
numWhitePixels = sum(sum(imDilated == 1));
ratio = numWhitePixels / numPixels;
% Threshold ratio to identify anthracnose disease
if ratio > 0.3
    disp('Anthracnose disease detected');
else
    disp('Leaf is healthy or has a different disease');
end
% Display original image with binary image overlay
overlay = imoverlay(im, imDilated, [1 0 0]);
figure;
imshow(overlay);
```

MATLAB CODE TO DETECT CERCOSPORA LEAF SPOT DISEASE:

```matlab
% Load leaf image
im = imread('cercospora.jpg');
% Convert image to grayscale
imGray = rgb2gray(im);
% Preprocess image to enhance contrast and remove noise
imPreprocessed = imadjust(imGray);
% Segment image using Otsu's method
level = graythresh(imPreprocessed);
imSegmented = imbinarize(imPreprocessed, level);
% Remove small objects from binary image
imCleaned = bwareaopen(imSegmented, 100);
% Dilate image to connect close spots or lesions
se = strel('disk', 5);
imOpened = imopen(imCleaned, se);
% Calculate the ratio of black pixels in the image
imSize = size(imOpened);
numPixels = imSize(1) * imSize(2);
numBlackPixels = sum(sum(imOpened == 0));
ratio = numBlackPixels / numPixels;
% Threshold ratio to identify cercospora leaf spot disease
if ratio > 0.1
    disp('Cercospora leaf spot disease detected');
else
    disp('Leaf is healthy or has a different disease');
end
% Display original image with binary image overlay
overlay = imoverlay(im, imcomplement(imOpened), [1 0 0]);
figure;
imshow(overlay);
```
RESULTS AFTER RUNNING MATLAB CODES

IMAGE PREPROCESSING OF LEAF WITH DISEASE

IMAGE PREPROCESSING OF LEAF WITHOUT DISEASE: (HEALTHY LEAF)
IMAGE SEGMENTATION OF A LEAF WITH DISEASE:

SNAPSHOT OF A LEAF SUFFERING WITH ALTERNARIA ALTERNATA DISEASE:

IMAGE SEGMENTATION OF A LEAF WITHOUT DISEASE:
MATLAB COMMAND WINDOW:

```
% Load the leaf image
leaf = imread('alternaria_alternata.jpg');

% Convert the image to grayscale
gray_leaf = rgb2gray(leaf);

% Perform image segmentation using Otsu's thresholding method
level = graythresh(gray_leaf);
binary_leaf = imbinarize(gray_leaf, level);

% Fill any holes in the binary image
filled_leaf = imfill(binary_leaf, 'holes');

% Remove any small objects from the binary image
```

SNAPSHOT OF A LEAF SUFFERING WITH BACTERIAL BLIGHT DISEASE:
MATLAB COMMAND WINDOW:

```matlab
1/ % Calculate the Euler number of the binary image
eulerNumber = bwuler(inCleaned);

20 % Threshold Euler number to identify bacterial blight disease
21 if eulerNumber < -1
22 disp('Bacterial blight disease detected');
23 else
24 disp('Leaf is healthy or has a different disease');
25 end

27 % Display original image with binary image overlay
28 overlay = imoverlay(in, inCleaned, [1 0 0]);
29 figure;
30 imshow(overlay);
```

Command Window
Warning: Function filter has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
Warning: Function gamma has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
>> bacteri

Warning: Function filter has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
Warning: Function gamma has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
Bacterial blight disease detected

SNAPSHOT OF A LEAF SUFFERING WITH ANTRACNOSE DISEASE:

MATLAB COMMAND WINDOW:

```matlab
44 % Threshold ratio to identify anthracnose disease
45 ratio = numWhitePixels / numPixels;
46 if ratio > 0.3
47 disp('Anthracnose disease detected');
48 else
49 disp('Leaf is healthy or has a different disease');
50 end

54 % Display original image with binary image overlay
55 overlay = imoverlay(in, imFiltered, [1 0 0]);
56 figure;
57 imshow(overlay);
```

Command Window
Warning: Function filter has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
Warning: Function gamma has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
>> anthracnose

Warning: Function filter has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
Warning: Function gamma has the same name as a MATLAB built-in. We suggest you rename the function to avoid a potential name conflict.
Anthracnose disease detected

>>
SNAPSHOT OF A LEAF SUFFERING WITH CERCOSpora LEAF SPOT DISEASE:

MATLAB COMMAND WINDOW:

CONCLUSION:

- In conclusion, the use of image processing techniques in identifying plant diseases has been shown to be effective in this study. The methodology used involves image acquisition, image pre-processing, image segmentation, and classification techniques.
- Image acquisition involves capturing images of plant leaves using a digital camera. Image pre-processing involves enhancing the images by removing noise and equalizing their histogram. Image segmentation is performed to separate the regions of interest from the background using the K-means clustering algorithm. Finally, the classification technique is used to classify the segmented regions into healthy or diseased based on their texture and color features.
- This methodology has shown promising results in accurately identifying the plant diseases from the images. The use of image processing techniques has also provided a non-destructive and non-invasive method for diagnosing plant diseases, which can potentially save time and resources. Further research can be done to improve the accuracy and efficiency of this method, including the use of more advanced machine learning algorithms for classification.