LEAF DISEASE DETECTION USING DEEP LEARNING

A MINI PROJECT REPORT

18CSC305J - ARTIFICIAL INTELLIGENCE

Submitted by

Kangkhita Pujari
[RA2011003010444]
Abhinav Kumar
[RA2011003010453]
Shatakshi Dubey
[RA2011003010461]
Susmeet Mitra
[RA2011003010467]

Under the guidance of

Aruna M

Assistant Professor, Department of Computer Science and Engineering

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE & ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

S.R.M. Nagar, Kattankulathur, Chengalpattu District

MAY 2023

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled **"Leaf Disease Detection"** is the bona fide workof **Kangkhita Pujari (RA2011003010444), Abhinav Kumar (RA2011003010453), Shatakshi Dubey (RA2011003010461) and Susmeet Mitra (RA2011003010467)** who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
GUIDE NAME	Dr. M. Pushpalatha
GUIDE	**HEAD OF THE DEPARTMENT**
Assistant Professor	Professor & Head
Department of Computing Technologies	Department of Computing Technologies

ABSTRACT

Plant diseases are a global problem that can cause significant damages and losses in crops. To prevent these problems and minimize losses, appropriate measures for disease identification must be introduced. Machine learning and computer vision are actively researched technical approaches that can achieve intelligent farming through early detection of plant diseases. A desirable application would be an aid for farmers or garden enthusiasts to diagnose plant diseases. However, most current applications achieve this function by submitting an image to a team of plant pathologists or expert garden advisers to get possible identification results and advice. Developing a more accessible and convenient application could significantly benefit the agricultural industry.

TABLE OF CONTENTS

ABBREVIATIONS

IOT	Internet of Things
PCA	Principal Component Analysis
RGB	Red Green Blue
YIQ	Luminance-In phase-Quadrature

CHAPTER 1

INTRODUCTION

Leaf detection is a field of study under the image recognition field of computer vision. Recognizing leaves is of utmost importance in biodiversity conservation. The major project, detection of diseases in leaves, is also another important milestone in conserving not just biodiversity but also saving crops from disease spread. The algorithm PCA has aided the process of leaf detection, by the monitoring of some basic features of leaves and then comparing the values obtained with the available data set. Leaf disease detection is an emerging field that uses computer vision and machine learning techniques to identify plant diseases based on images of their leaves. This approach can provide early detection of diseases, which is critical for preventing their spread and minimizing losses in crop yields. Leaf disease detection has significant potential to aid farmers and agriculturalists in diagnosing diseases quickly and accurately, and providing targeted treatments. The field is still in its early stages, and ongoing research is focused on improving the accuracy and reliability of detection methods, as well as developing new techniques for detecting a wider range of diseases.

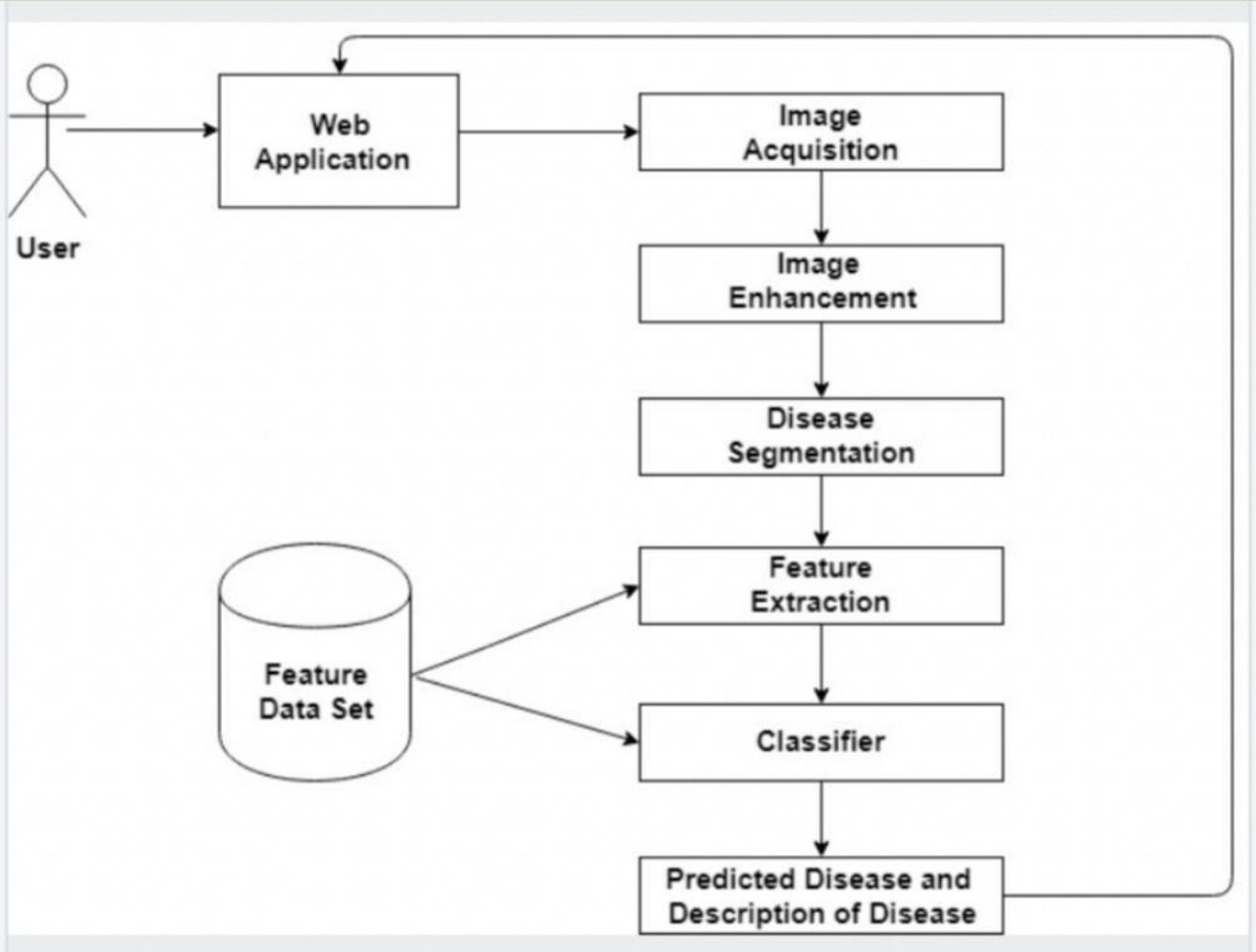
CHAPTER 2

LITERATURE SURVEY

Leaf disease detection is an essential aspect of precision agriculture as it enables early detection of diseases that can significantly impact crop yield and quality. With the increasing demand for food, it is essential to develop effective methods for disease detection to improve crop management practices and reduce economic losses due to disease outbreaks. In recent years, computer vision and machine learning techniques have gained significant attention for leaf disease detection. Feature extraction is a crucial step in this process, where specific characteristics are identified in the image to distinguish between healthy and diseased leaves. Various techniques, such as RGB, HSV, YIQ, and Dithered Images, have been proposed for feature extraction. Among them, RGB color space is widely used as it separates images into their red, green, and blue components, making it easier to identify changes in color intensity that may indicate disease symptoms. The HSV and YIQ color spaces provide an alternative to RGB that can be useful in specific applications**.**

CHAPTER 3

SYSTEM ARCHITECTURE AND DESIGN



CHAPTER 4

METHODOLOGY

Data collection: This involves acquiring images of plant leaves that can be used for disease detection. These images can be captured using various devices, such as smartphones or cameras.

Image preprocessing: Before the images can be analyzed, they need to be preprocessed to remove any noise or artifacts that may affect the accuracy of the disease detection algorithm. This can include steps such as cropping, resizing, and color correction.

Feature extraction: This is a critical step in the process that involves identifying specific characteristics in the image that can help distinguish between healthy and diseased leaves. Various techniques, such as RGB, HSV, YIQ, and Dithered Images, can be used for feature extraction.

Disease detection: Once the features have been extracted, a classification algorithm is applied to determine if the leaf is healthy or diseased. Various machine learning algorithms, such as Support Vector Machines (SVM) or Convolutional Neural Networks (CNN), can be used for this purpose.

CNNs consist of several layers, including convolutional layers, pooling layers, and fully connected layers. The convolutional layer applies a set of learnable filters to the input image to generate a set of feature maps. These feature maps represent local patterns of the input image, such as edges, corners, and textures. The pooling layer downsamples the feature maps to reduce their spatial size and computational complexity. The fully connected layer takes the

output of the previous layer and computes the final classification or regression output.

One of the main advantages of CNNs is their ability to learn hierarchical representations of input images, with each layer learning more abstract and complex features than the previous layer. This makes CNNs well-suited for tasks such as object recognition, where the network can learn to recognize simple features such as edges and gradually build up to recognizing more complex objects.

CNNs have been very successful in a wide range of computer vision tasks, including image classification, object detection, segmentation, and image generation. They have also been applied in other domains such as natural language processing and speech recognition.

SVMs are based on the idea of finding the hyperplane that maximally separates two classes of data points in a high-dimensional feature space. The hyperplane that maximizes the margin between the two classes is called the maximum-margin hyperplane. SVMs seek to find this hyperplane by solving an optimization problem.

In addition to linear classification, SVMs can also be used for nonlinear classification by using kernel functions. A kernel function maps the input data points to a higher-dimensional feature space, where a linear hyperplane can be used to separate the data points. Common kernel functions include the polynomial kernel, Gaussian kernel, and sigmoid kernel.

SVMs have several advantages, including their ability to handle high-dimensional data and their robustness to overfitting. They are widely used in a variety of applications, including image classification, text classification, and bioinformatics. However, SVMs can be sensitive to the choice of kernel function and the value of the regularization parameter.

User interface: The results of the disease detection algorithm are typically presented to the user through a graphical user interface (GUI), which can provide information about the type and severity of the disease.

The SIFT (Scale-Invariant Feature Transform) algorithm is a computer vision algorithm used for detecting and describing local features in images. It was first introduced by David Lowe in 1999.

The SIFT algorithm is widely used for image recognition, object detection, and 3D reconstruction. It works by identifying key points in an image that are invariant to scale, orientation, and illumination changes. These key points are then described using a set of local image features that can be matched across different images.

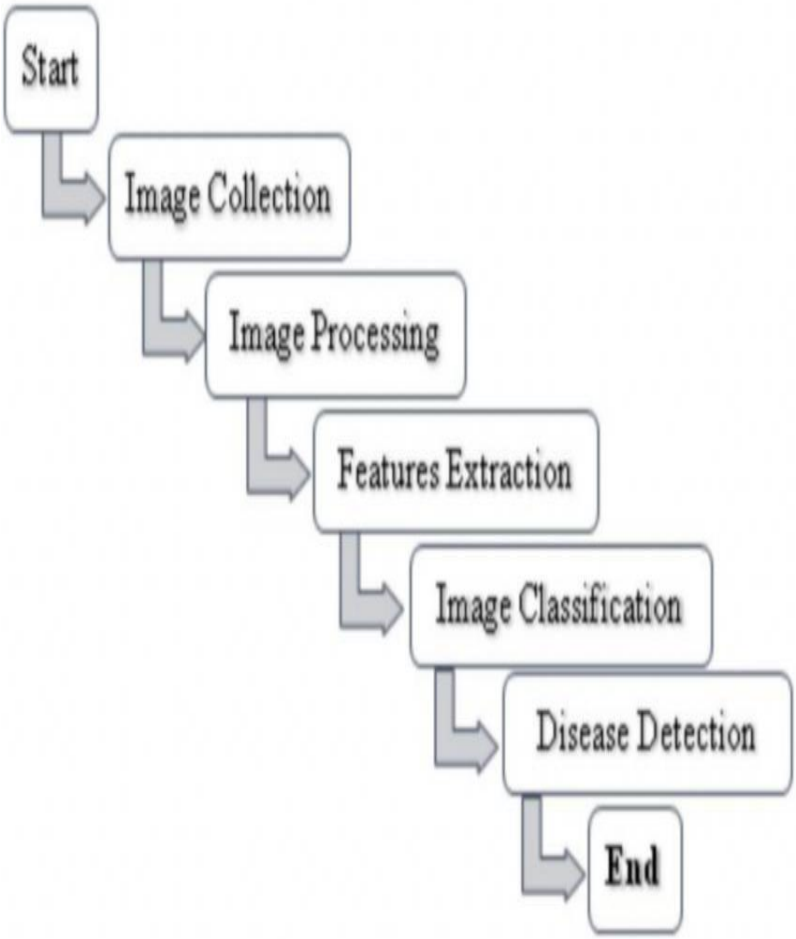
The SIFT algorithm has several steps, including scale-space extrema detection, keypoint localization, orientation assignment, and keypoint descriptor generation. The algorithm uses a Gaussian scale space to detect keypoints at multiple scales and orientations. Keypoint descriptors are generated by computing a histogram of gradient orientations around each keypoint.

SIFT is a robust and widely used feature detection algorithm, but it can be computationally expensive and may not work well on low-contrast or blurry images.

Matplotlib is a popular data visualization library for Python. It provides a wide range of plotting tools for creating high-quality graphs, charts, and figures in Python.

With Matplotlib, you can create a variety of different types of plots, including line plots, scatter plots, bar plots, histograms, and more. Matplotlib provides a wide range of customization options, including color, line style, marker style, axis labels, legends, and more.

Matplotlib has a simple and intuitive interface, making it easy to create basic plots quickly. For example, you can create a simple line plot with the following code:


Start
Image Collection
Image Processing
Features Extraction
Image Classification
Disease Detection
End

CHAPTER 5

CODING AND TESTING

```
import cv2
import numpy as np
import matplotlib.pyplot as plt
import numpy as np
import cv2
import matplotlib.pyplot as plt
from scipy import ndimage
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler
from matplotlib.colors import hsv_to_rgb
img = cv2.imread('./image.jpg')
plt.imshow(img)
plt.show()
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(img)
plt.show()
hsv_img = cv2.cvtColor(img, cv2.COLOR_RGB2HSV)
plt.imshow(hsv_img)
lower_green = np.array([25,0,20])
upper_green = np.array([100,255,255])
mask = cv2.inRange(hsv_img, lower_green, upper_green)
result = cv2.bitwise_and(img, img, mask=mask)
plt.subplot(1, 2, 1)
plt.imshow(mask, cmap="gray")
plt.subplot(1, 2, 2)
plt.imshow(result)
plt.show()
lower_brown = np.array([10,0,10])
upper_brown = np.array([30,255,255])
disease_mask = cv2.inRange(hsv_img, lower_brown, upper_brown)
disease_result = cv2.bitwise_and(img, img, mask=disease_mask)
plt.subplot(1, 2, 1)
plt.imshow(disease_mask, cmap="gray")
plt.subplot(1, 2, 2)
plt.imshow(disease_result)
```

```
plt.show()
final_mask = mask + disease_mask
final_result = cv2.bitwise_and(img, img, mask=final_mask)
plt.figure(figsize=(15,15))
plt.subplot(1, 2, 1)
plt.imshow(final_mask, cmap="gray")
plt.subplot(1, 2, 2)
plt.imshow(final_result)
plt.show()
surf = cv2.xfeatures2d.SURF_create(400)
# Find keypoints and descriptors directly
kp, des = surf.detectAndCompute(final_result,None)
len(kp)
print(kp)
print(des)
```

CHAPTER 6

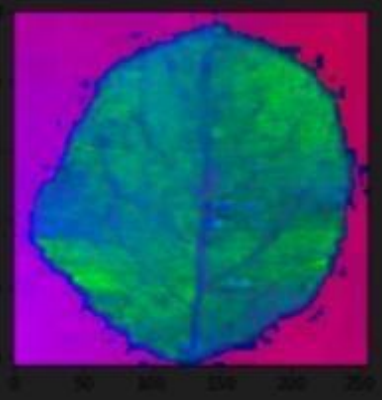
SCREENSHOTS AND RESULTS

```
img = cv2.imread('./image.jpg')
plt.imshow(img)
plt.show()
```

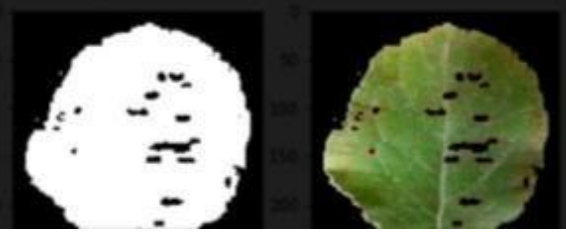
```
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
plt.imshow(img)
plt.show()
```

```
hsv_img = cv2.cvtColor(img, cv2.COLOR_RGB2HSV)
plt.imshow(hsv_img)
```

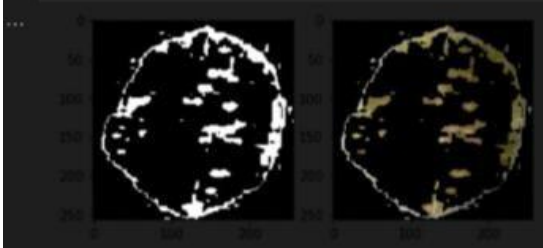
```
<matplotlib.image.AxesImage at 0x7f85f82aad90>
```



```
lower_green = np.array([25,0,20])
upper_green = np.array([100,255,255])
mask = cv2.inRange(hsv_img, lower_green, upper_green)
result = cv2.bitwise_and(img, img, mask=mask)
plt.subplot(1, 2, 1)
plt.imshow(mask, cmap="gray")
plt.subplot(1, 2, 2)
plt.imshow(result)
plt.show()
```



```
lower_brown = np.array([10,0,10])
upper_brown = np.array([30,255,255])
disease_mask = cv2.inRange(hsv_img, lower_brown, upper_brown)
disease_result = cv2.bitwise_and(img, img, mask=disease_mask)
plt.subplot(1, 2, 1)
plt.imshow(disease_mask, cmap="gray")
plt.subplot(1, 2, 2)
plt.imshow(disease_result)
plt.show()
```



```
final_mask = mask + disease_mask
final_result = cv2.bitwise_and(img, img, mask=final_mask)
plt.figure(figsize=(15,15))
plt.subplot(1, 2, 1)
plt.imshow(final_mask, cmap="gray")
plt.subplot(1, 2, 2)
plt.imshow(final_result)
plt.show()
```

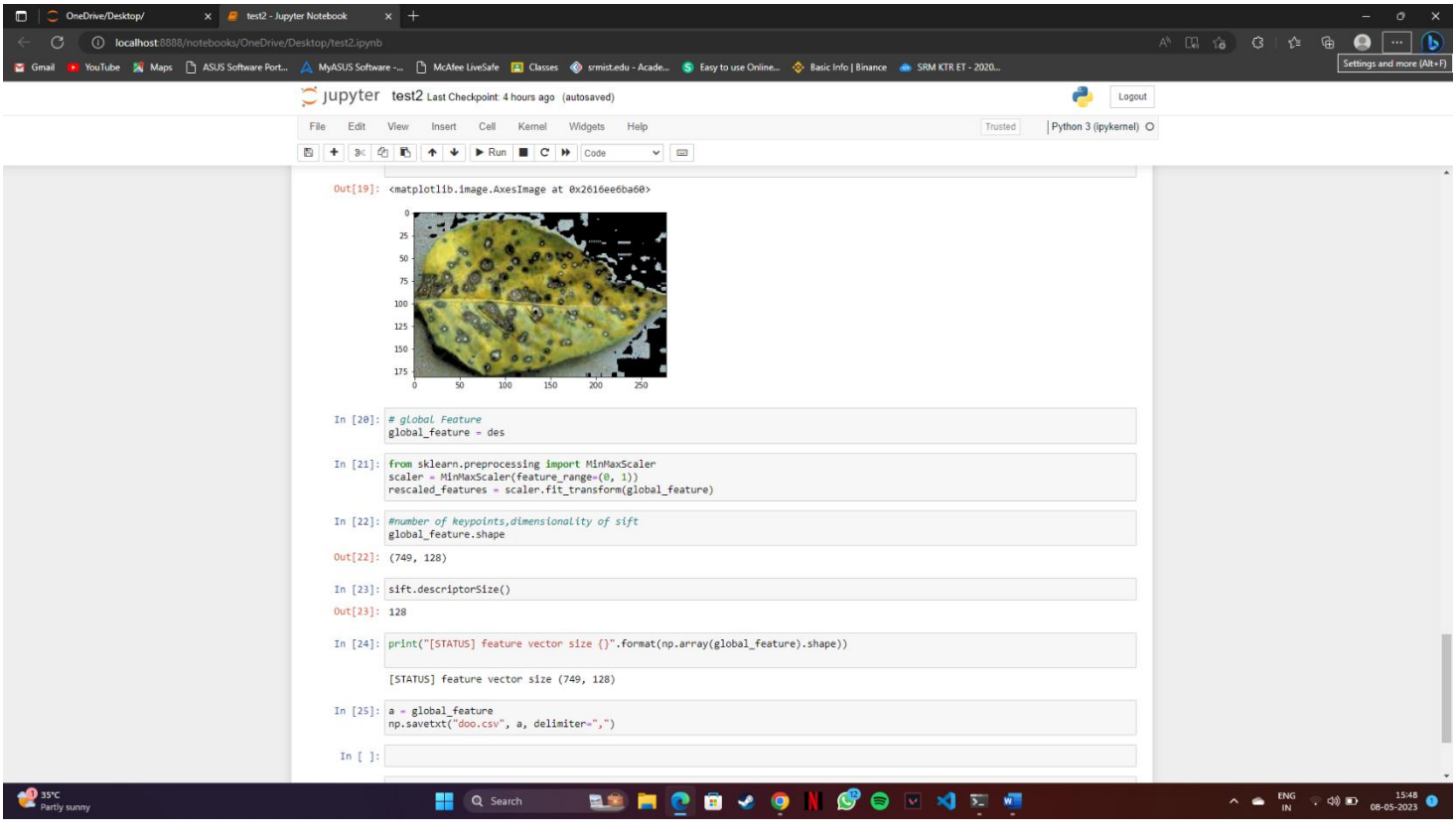
```
print(des)
```

```
[[ 0.          0.          0.         ... -0.00021652  0.00097847
  0.00114048]
 [ 0.          0.          0.         ...  0.00016049  0.00091302
  0.00066208]
 [ 0.00701571 -0.00964417  0.00778186 ...  0.00178146  0.00166531
  0.00231891]
 ...
 [-0.00191567  0.00205072  0.00200102 ...  0.00166039  0.00220279
  0.00291323]
 [-0.00528289  0.01501577  0.01313482 ... -0.00098651  0.00467307
  0.00252157]
 [-0.00095609 -0.00620609  0.02660246 ... -0.00146699  0.00247034
  0.00225198]]
```

```
img2 = cv2.drawKeypoints(final_result,kp,None,(255,0,0),4)
```

```
plt.imshow(final_result)
```

```
<matplotlib.image.AxesImage at 0x23acd6dff28>
```



CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENTS

The goal of the leaf-based plant disease detection project is to achieve an accurate classification of different plant diseases based on their leaf images, with a high level of precision. To accomplish this, the model should be able to generalize well on unseen data and be robust to variations in leaf appearance, such as lighting conditions, background noise, and scale. Additionally, the deployed model should demonstrate efficient inference time and compatibility with different hardware platforms. The project has the potential to contribute significantly to biodiversity monitoring, environmental conservation, and agricultural automation by providing a reliable tool for plant disease identification.

REFERENCES

• Camargo A. and J. S. Smith. 2008. An image-processing based algorithm to automatically identify plant disease Visual symptoms. Bio.Systematic. Engineering., 102: 9 – 21

• Camargo, A. and J. S. Smith. 2009. Image processing for pattern classification for the identification of disease-causing agents in plants. Com. Elect. Agr.66:121 –125.

• Guru, D. S., P. B. Mallikarjuna and S. Manjunath. 2011. Segmentation and Classification of Tobacco Seedling Diseases. Proceedings of the Fourth Annual ACM Bangalore Conference.

• Zhao, Y. X., K. R. Wang, Z. Y. Bai, S. K. Li, R. Z. Xieand S. J. Gao. 2009. Research of Maize Leaf Disease Identifying Models Based Image Recognition. Crop Modeling and Decision Support. Tsinghuauni.press. Beiging. pp. 317-324.