PROJECT REPORT

Project title:Crop Disease Identifier using Image Processing
Course:Fundamentals in AI and ml
Student Name:ADARSH KUMAR RAIWAL
Registration Number:25BCY10183

1. Introduction

The Crop Disease Identifier is a Python-based image processing project designed to assist farmers and agricultural workers in identifying common plant leaf diseases. Instead of manually inspecting crops, this tool analyzes leaf images and predicts the disease category using histogram comparison and basic computer vision techniques. Since even small farms cannot afford costly inspection tools or AI-based models, this project aims to provide a lightweight, offline, and beginner-friendly solution that utilizes only fundamental Python and OpenCV concepts.

2. Problem Statement

Farmers often struggle to detect plant diseases early, leading to reduced crop yield and financial losses. Manual inspection is time-consuming, error-prone, and highly dependent on expertise.
There is a clear need for a simple, low-cost, digital system that can:

- Analyze uploaded leaf images
- Identify basic disease patterns
- Work without internet or sensors
- Remain easy to use for non-technical users

This project addresses the above challenges using basic computer vision concepts suitable for IPSP-level programming skills.

3. Objectives

The primary objectives of this project are:

- To create a tool that detects common leaf diseases using simple image comparison techniques.
- To apply Python concepts such as lists, functions, loops, and modular programming.
- To use OpenCV for extracting image features like color histograms and edges.
- To enable a file-based dataset that can be expanded easily.
- To provide meaningful disease detection along with suggested remedies.

4. Methodology & Algorithm

The project uses a modular design where separate components handle loading images, extracting features, and comparing them with a sample dataset.

General Algorithm

1. Start the program.
2. Read the input leaf image uploaded by the user.
3. Load reference disease images from dataset folders:
   - healthy
   - rust
   - blight
4. Convert both images to HSV color space.
5. Extract HSV histogram from each image.
6. Compare histograms using Bhattacharyya distance.
7. Choose the disease category with the minimum distance.
8. Display the detected disease and recommended actions.
9. Stop.

5. Technical Implementation (Concepts Used)

This project implements many core Python concepts relevant to IPSP.

A. List & Nested Lists

Definition:

Lists store multiple values, while nested lists allow hierarchical storage.

Application:

The project stores dataset paths and category lists in simple Python lists:

```
categories = ["healthy", "rust", "blight"]
```

Feature scores and comparison results are stored as list entries for iteration and selection.

B. File Handling (Image Reading)

Definition:

File handling refers to reading and writing data from storage.

Application:

The system reads image files using OpenCV:

```
img = cv2.imread("dataset/rust/1.jpg")
```

Instead of text files, the project handles image files as input data.

The dataset folders serve as a persistent storage mechanism.

C. Functions (Modularity)

Definition:

Functions break down code into smaller tasks.

Application:

Major functions include:

- feature_extract(img) – extracts histogram features
- compare(img1, img2) – compares two images
- identify(path) – identifies disease by scanning dataset

This modularity ensures readable and maintainable code.

D. Control Structures (Loops & Conditionals)

Definition:

Loops iterate over data, while conditionals manage decisions.

Application:

- A for loop iterates through each category folder.
- Nested loops compare input image with every sample image.
- if statements check for minimum score and determine disease.

These structures help automate classification without human intervention.

6. Testing & Validation

The model was tested through multiple scenarios:

Case 1 – Healthy Leaf

Input: Clear green leaf image

Output: Detected as "Healthy" → Suggest "Normal Growth"

Result: Correct

Case 2 – Rust Disease Leaf

Input: Brownish patches on leaf

Output: Classified as "Rust"

Result: Successful

Case 3 – Blight Leaf

Input: Black circular spots

Output: "Blight Detected"

Result: Accurate

Case 4 – Noisy / Poor Quality Image

Result: Still provides closest category due to histogram comparison logic

Case 5 – Dataset Expansion

7. Conclusion

The Crop Disease Identifier successfully demonstrates how Python and basic computer vision techniques can be used to solve real-world agricultural problems. The project does not rely on any advanced machine learning model, making it lightweight and suitable for beginner programmers.

It applies essential IPSP concepts like lists, loops, modular coding, and file handling while introducing basic image processing fundamentals. This system can be upgraded with machine learning models, mobile integration, or cloud-based datasets in the future, expanding its real-world impact.