Project Description

Background

In recent years, the integration of technology in agriculture has gained significant attention. One of the key areas of focus is the monitoring of plant growth in controlled environments, such as greenhouses. This project aims to leverage image processing techniques to analyze and monitor plant growth over time, providing insights into growth patterns, leaf count, and overall plant health.

Problem

The primary challenge addressed by this project is the need for an efficient and accurate system to monitor plant growth in controlled environments. Traditional methods of monitoring are often labor-intensive and subjective, leading to inconsistencies in data collection and analysis. This project seeks to develop an algorithm that can automatically capture and analyze time-lapse images of plants, segmenting plant regions to highlight changes in growth patterns, leaf count, and health indicators.

OBJECTIVES

To establish a robust image acquisition model that captures time-lapse images.

To develop algorithms for image pre-processing and segmentation to analyze growth patterns and health indicators.

01. Image Acquisition Model

- Set up a mobile camera in a fixed position
- Use time-lapse photography techniques to capture images at regular intervals.

02. Pre Processing Techniques

1. **Used Libraries**
   - **OpenCV**
     - ⇨ Image processing, contour detection, and morphological operations.
   - **Numpy**
     - ⇨ Numerical operations and array manipulations
   - **Matplotlib**
     - ⇨ Visualizing the output Images and plotting the graph.
   - **OS**
     - ⇨ File and directory operations

2. **Image Loading & Validation**
   - cv2.imread()
   - File extension check (.png , .jpg , .jpeg)

3. **Color Space Conversion (BGR → HSV)**
   - cv2.cvtColor()

4. **Green Color Masking**
   - cv2.inRange()

5. **Morphological Operations (Noise Removal)**
   - Opening (cv2.MORPH_OPEN)
   - Closing (cv2.MORPH_CLOSE)

6. **Contour Detection & Filtering**
   - cv2.findContours()
   - Area-based filtering

03.Segmentation Techniques

1. **Color-Based Segmentation (HSV Thresholding)**
   - **Technique:**
     - Convert the image from BGR to HSV for better color segmentation.
     - Use cv2.inRange() to create a binary mask where green pixels are white (255) and others are black (0).
   - **Purpose:**
     - Isolates green leaves from the background.

2. **Morphological Operations (Noise Removal)**
   - **Techniques**:
     1. **Opening (cv2.MORPH_OPEN)**
        - Removes small noise in the background.
     2. **Closing (cv2.MORPH_CLOSE)**.
        - Fills small holes in the detected leaves.
   - **Purpose**:
     - Refines the binary mask to improve contour detection accuracy.

3. **Contour-Based Segmentation**
    - **Technique**:
        - Use cv2.findContours() to detect boundaries.
    - **Purpose**:
        - Extracts individual leaf regions for area measurement and counting.

04.Error Cases & Causes

1. **False Edge Detection**
    - Cause:
        - Low contrast between plant and background (dark leaves on dark soil).
        - Overlapping leaves creating complex edges.
    - Effect:
        - Broken contours or incorrect plant segmentation.

2. **Incorrect Largest Contour Selection**
    - Cause:
        - Background objects (pots) mistaken as part of the plant.
    - Effect:
        - Overestimated height and area.
        - Showing incorrect leaf counts

3. **Lighting & Shadows**
    - Cause:
        - Uneven lighting creates artificial edges.
        - Shadows merge with the plant contour.
    - Effect:
        - Incorrect edge detection and height and area calculation.

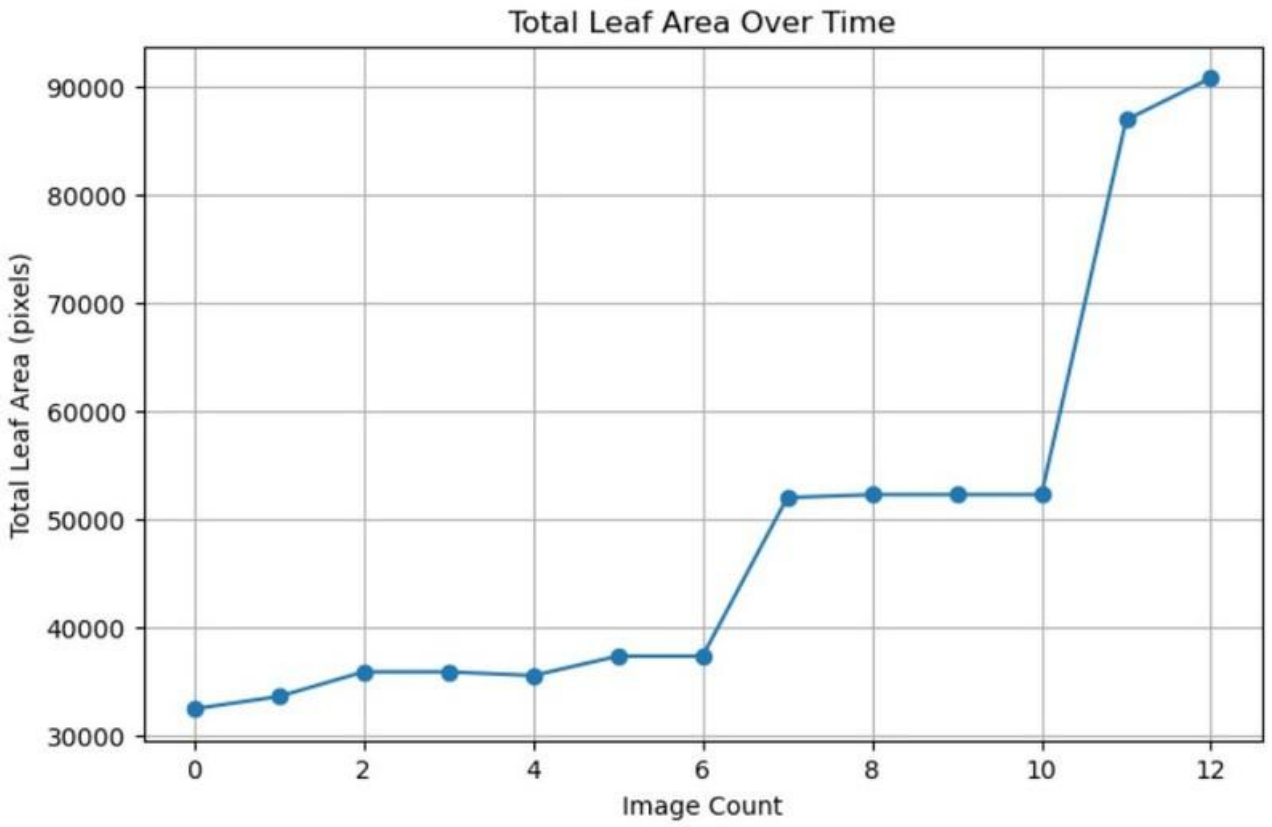
Sample pre processed Image

Original Image

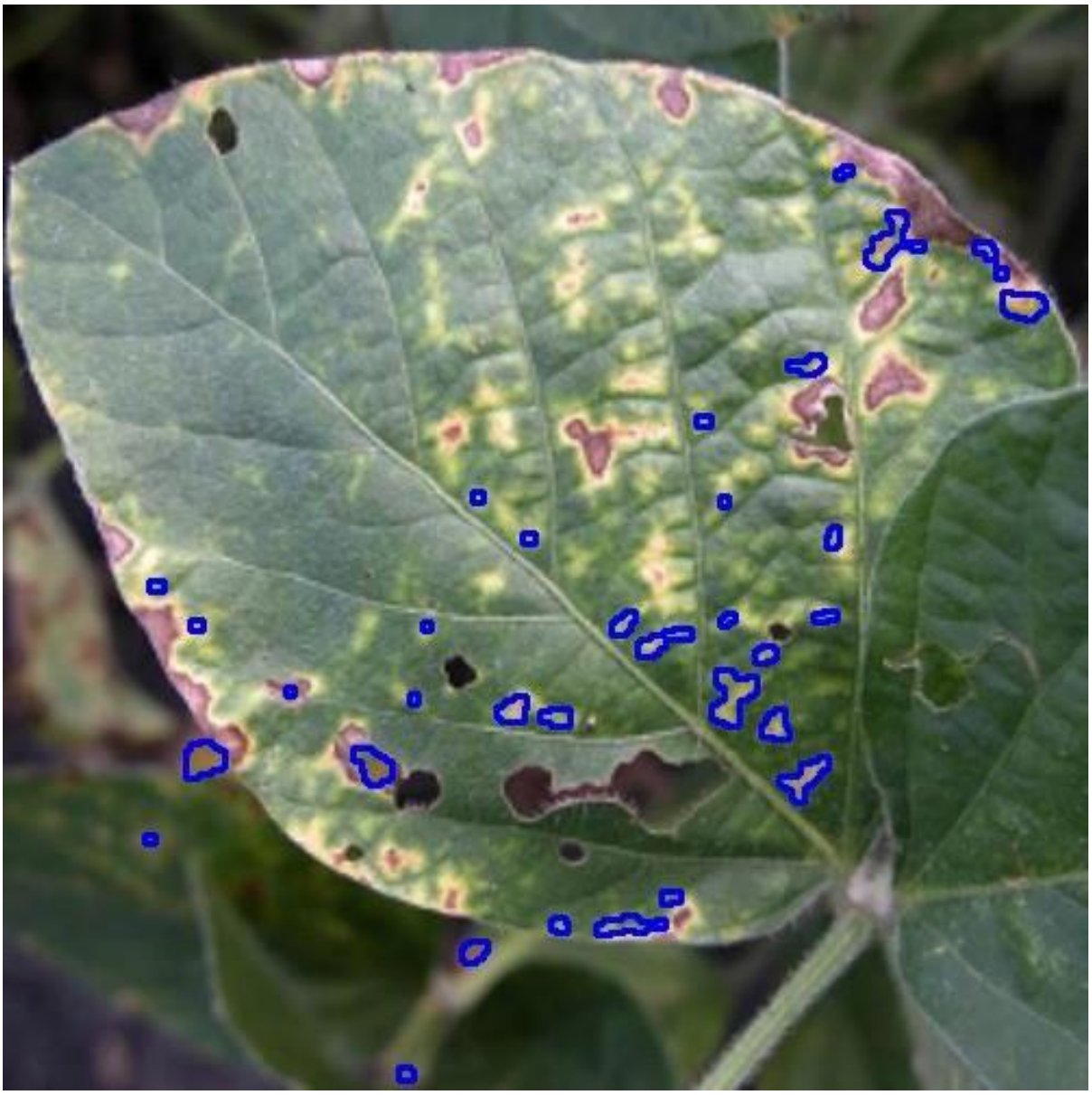
Segmented Image (Leaf Mask)

Leaf Count: 1, Total Area: 268245 px

Graphical Representation of Leaf Area Growth



Detecting Unhealthy Leaves



Graphical Representation of Monitoring Height