Sustainable Precision Agriculture: An Autonomous Approach to Soil Sensing and Sapling Tree Management

by

Tajwar Al Haque Robin

21101312

Khondaker Ashef Hossain

21101193

Fatin Ishtiaque

21101206

Alimun Al Mikdad

21301219

A thesis submitted to the Department of Computer Science and Engineering in partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering

Department of Computer Science and Engineering

Brac University

January 2025

Declaration

It is hereby declared that

1. The thesis submitted is our own original work while completing a degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

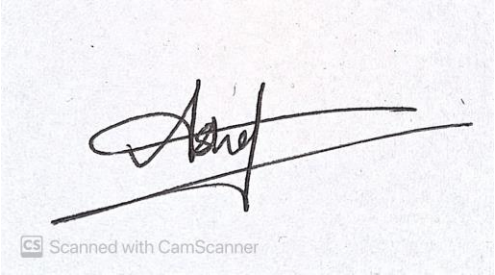
Student's Full Name & Signature:

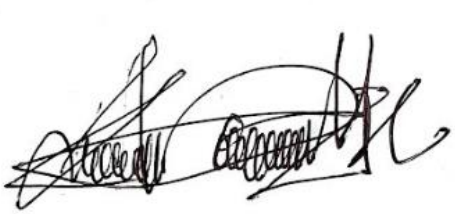
TAJWAR AL HAQUE ROBIN

21101312

Fatin Ishtiaque

21101206





KHONDAKER ASHEF HOSSAIN

21101193

ALIMUN AL MIKDAD

21301219

Approval

The thesis titled “Sustainable Precision Agriculture: An Autonomous Approach to Soil Sensing and Sapling Tree Management” submitted by

1. Tajwar Al Haque Robin (21101312)
2. Khondaker Ashef Hossain (21101193)
3. Fatin Ishtiaque (21101206)
4. Alimun Al Mikdad (21301219)

Of Spring 2024 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science on May, 2024.

Examining Committee:

Supervisor:

(Member)

Riad Ahmed

Lecturer

Department of Computer Science

Brac University

Table of Contents

Declaration 2

Approval 3

Abstract 9

1. Introduction 10

2. Research Purpose 12

2.1 Objectives

2.1.1 On-site Soil Inspection and Sampling 13

2.1.2 Autonomous Robotic Precision Planting 14

2.1.3 Solutions in Precision Agriculture 14

2.1.4 Automated System for In-Situ Evaluation 14

2.1.5 Unique Agriculture Technological innovations 14

2.1.6 Improved Disease Management 15

2.1.7 Enhanced Soil Health Monitoring 15

2.1.8 Optimal Use of Resources 15

2.2 Research problems and solutions 16

3. Literature Review 18

3.1 Vision-Based Perception 19

3.2 Soil Analysis 19

3.3 Ion-Selective Electrode (ISE) 21

3.4 C3088 Camera with MATLAB GUI 22

3.5 pH Sensor 24

3.6 Laser-Based Arrays for Height Measurement 24

3.7 An improved pistachio detection approach using YOLO-v8 Deep Learning Models 26

3.7.1 Object Detection Evolution 26

3.7.2 YOLO Model Series 26

3.7.3 Applications in Agriculture 26

3.7.4 Challenges in Pistachio Detection 27

3.7.5 Previous Work on Pistachio Detection 27

3.7.6 Methodological Improvements 27

3.7.7 Performance Metrics 28

4. Robot Specifications **29**

5. Methodologies **32**

5.1 Methodology for YOLOv8 Model Training for Leaves Detection 33

4.1.1 Model Overview 33

4.1.2 Training the Model 35

4.1.2.1 Data Preparation 35

4.1.2.2 Model Setup 36

4.1.2.3 Training **36**

5.2 Methodology for Plant Disease Classification using Xception and DenseNet121 37

4.2.1 Environment Setup 37

4.2.2 Data Loading and Preprocessing 38

4.2.3 Data Inspection and Visualization 38

4.2.4 Image Data Augmentation 38

4.2.5 Train-Validation Split 39

4.2.6 Creating Data Generators 39

4.2.7 Model Architecture 40

4.2.8 Model Ensemble 41

4.2.9 Learning Rate Scheduling 41

4.2.10 Model Checkpointing 42

4.2.11 Model Training 42

4.2.12 Saving and Visualizing Training History 42

4.2.13 Test Data Prediction 43

5. Work Plan

6. Results

7. Future Work **67**

8. Conclusion **68**

Bibliography **70**

Abstract:

In modern urban forestry and precision agriculture, the successful adaptation of seedlings and saplings to evolving soil and microenvironmental conditions is crucial for their survival and growth. This research investigation introduces an autonomous agricultural robot that uses artificial intelligence (AI) and machine learning (ML) techniques to monitor soil quality and identify plant illnesses, with the goal of improving crop management and sustainable practices. The suggested system collects and processes real-time data on essential soil factors such as pH, conductivity, humidity, temperature, and nutrient levels. A normalization-based methodology is used to calculate a soil damage level (SDL), allowing for accurate assessment of soil health by comparing real-time data against predefined reference values for crops such as rose, guava, and tomato. The system's AI-based plant disease detection module, leveraging the YOLOv11 deep learning model, effectively classifies plant leaves into categories—healthy, scab, rust, and multiple disease— and generates corresponding health scores. The results of the study illustrate the system's capacity to identify soil deficiencies and diagnose plant diseases with high accuracy, allowing for early intervention and precision treatment techniques. The long-term objective is to optimize plant growth environments by addressing nutrient deficiencies and improving planting practices through the integration of AI-driven automation. This research contributes to the advancement of precision agriculture by providing a robust, autonomous system capable of delivering actionable insights for soil and crop management, ensuring enhanced productivity and sustainability. The ultimate goal is to optimize the growth environment by identifying and rectifying soil deficiencies while generating a health score for the plants, ensuring the quality of saplings through precise planting methods.

Keywords— Precision Agriculture, Plant Disease Detection, Soil Quality Monitoring, Machine Learning, Artificial Intelligence, Autonomous Robot, YOLOv11.

Chapter 1

Introduction

Bangladesh remains one of the most agriculture-intensive countries globally, characterized by extensive crop diversity and a large farming workforce. With the rising demand for productivity and sustainability, smart farming—the convergence of precision agriculture, automation, and machine learning—has garnered significant attention for optimizing agricultural management. According to Singh et al. (2020), precision agriculture (PA) not only enhances yields but also minimizes operational expenses by tailoring farm inputs to localized needs. This paradigm shift in agriculture leverages an array of technologies, including advanced sensors, robotics, image processing, and artificial intelligence (AI), to improve overall efficiency and foster environmentally sustainable practices (Bhatnagar et al., 2022; Liakos et al., 2018).

A cornerstone of modern agricultural management is soil health monitoring, which directly affects crop yield and resource utilization. Automated systems using multi-parameter soil sensors can measure pH, conductivity, humidity, temperature, and nutrient levels—factors integral to informed decision-making (Liakos et al., 2018). When integrated with AI-based models, such systems enable real-time alerts for anomalies in soil conditions, such as nutrient deficiencies or excessive temperature, thereby expediting countermeasures before significant crop damage occurs. Further, machine vision and deep learning methods enhance plant health diagnostics by detecting visual symptoms of disease on leaves, stems, and fruits. Early recognition of

diseases—especially those manifesting through leaf discolorations or deformations—can lead to prompt treatments and significantly reduce yield losses (Khan et al., 2021; Sishodia et al., 2020).

Recent advancements in convolutional neural networks (CNNs), such as the YOLO (You Only Look Once) family of architectures, have made it feasible to identify plant diseases, pests, and nutrient deficiencies with high accuracy (Redmon & Farhadi, 2018). By leveraging robust image processing modules, an agricultural robot can autonomously roam the field, capture images, and classify plant anomalies on-site. Additionally, the same platform can measure critical soil parameters—via sensors employing protocols like RS485—for immediate feedback on soil quality. The fusion of these hardware and software components closes the loop between problem detection and corrective action, an integral concept in precision agriculture (Liakos et al., 2018; Singh et al., 2020).

This research focuses on sapling tree management by creating an integrated framework that combines soil sensing, disease detection, and nutrient deficiency analysis to assist farmers and agronomists in making evidence-based decisions. Specifically, the system will monitor the soil conditions of newly planted saplings and detect potential foliar diseases at an early stage using image-based classification. By unifying these functionalities, farmers can react promptly to emerging threats such as suboptimal soil conditions or pathogen infections, ensuring a more robust and sustainable growth process. Ultimately, this holistic approach promises to improve yield, reduce resource wastage, and enhance environmental outcomes through judicious use of fertilizers, fertilizers, and water (Bhatnagar et al., 2022).

In summary, the proposed framework integrates soil parameter monitoring, deep learning–based disease recognition, and autonomous robotic navigation. It stands to contribute to the broader

field of precision agriculture and smart farming, helping to transform traditional agricultural practices into data-driven, efficient, and sustainable systems. This thesis further aims to promote the development of specialized robotic solutions for Bangladeshi agriculture, thereby aligning with global sustainability agendas while strengthening local food security and economic resilience.

Chapter 2

The purpose of this research is to design, implement, and evaluate an integrated precision agriculture framework that unifies multi-sensor soil health monitoring, deep learning–based disease detection, and autonomous robotic navigation for efficient sapling management, with the aim of enhancing productivity, reducing resource waste, and promoting sustainable agricultural practices (Liakos et al., 2018; Sishodia et al., 2020). The research objectives include developing a high-fidelity RS485-based multi-sensor system to capture key soil parameters—pH, conductivity, humidity, temperature, and nutrients (NPK)—ensuring accuracy and reliability under varying conditions to enable timely interventions and predictive analytics (Bhatnagar et al., 2022). It will also implement an advanced convolutional neural network (CNN)-based disease identification module, using models like YOLOv3 to detect early-stage diseases or nutrient deficiencies from leaf images in real time, enabling targeted and localized interventions (Khan et al., 2021; Redmon & Farhadi, 2018). Furthermore, the research aims to design a mobile robotic platform with line-following, obstacle avoidance, and soil sampling capabilities to collect consistent in-situ data from predefined grid points, reducing labor and error (Krishnaswamy & Bhat, 2021; Shamshiri et al., 2018). A data fusion and real-time alert system will integrate sensor metrics and visual disease

classifications to generate actionable notifications, combining threshold-based and machine learning–based triggers to ensure reliability and trust in automated alerts (Bhatnagar et al., 2022; Sishodia et al., 2020). The system will be evaluated through field trials for accuracy, scalability, cost-effectiveness, and sustainability, focusing on yield improvement, optimized resource use, and environmental impact, validating its potential to lower operational costs and align with eco-friendly farming practices (Zhang et al., 2019; Shamshiri et al., 2018). Finally, comprehensive documentation, training materials, and regional customization will ensure user adoption, scalability, and relevance across different soil types, climates, and farm sizes, with potential collaborations to strengthen its implementation in Bangladesh and promote sustainable precision agriculture (Krishnaswamy & Bhat, 2021; Singh et al., 2020). By achieving these objectives, this research aims to deliver a robust, data-driven solution to monitor, analyze, and manage factors critical to sapling health, minimizing losses, maximizing efficiency, and advancing sustainable agricultural goals (Bhatnagar et al., 2022; Liakos et al., 2018).

Chapter 3

Literature Review

Electrochemical plans have been used in farming for a number of decades. With the development of technology, outdated equipment gave way to mechatronics, the fundamental component of modern agriculture. Modern agriculture is built on precision agriculture (PA), a notion that originally appeared in the early 1980s. Precision farming is made possible by contemporary agriculture's mechatronic technology, such as the Internet of Things. These solutions are frequently employed to keep an eye on and manage every subprocess. More scientists are striving day by day to create vehicles for agricultural activities that are more sensible and adaptive.

Most researchers working on precision agriculture nowadays are creating mobile robots including autonomous vehicles (Burks et al., 2005; Blackmore et al., 2007). Three separate verticals comprise the designs to be merged in order to develop precision farming: sensor modules, and mobile robot navigation implements (Framework and Applications). This approach has been used by several nations, such as the US, the EU, Denmark, Australia, Finland, India, and others. Robots can be managed using a single control space to carry out a variety of tasks, including seedbed preparation, crop scouting, weed mapping, robotic weeding control, micro-spraying, robotic gantry, robotic irrigation, and vision-based, sensor-based, inertial, active beacon, GPS, map-based, and landmark specialized navigation techniques (Kushwaha, 2020) [11]. Over the next 50 years, agricultural efficiency has to rise by 40 percent to meet the growing demand for food without expanding land area. Increasing agricultural production is probably going to need the use of

controlled environment agriculture (CEA) systems. The efficiency of the agricultural system may be assessed and improved by carefully controlling variables including light, water, humidity, carbon dioxide, temperature, and plant nutrition using a CEA.

3.1 Vision-Based Perception

Over the past two decades, numerous autonomous vehicles, especially agricultural robots, have been integrated thanks to specialized sensors (e.g., machine vision, global positioning systems (GPS), real-time kinematics (RTK), laser-based equipment, and inertial devices), actuators (e.g., hydraulic cylinders, linear, and rotational electrical motors), and electronic equipment (e.g., embedded computers, industrial PCs, and PLCs) (Åstrand & Baerveldt, 2002). [12]. Numerous precision agricultural operations, including planting new saplings, harvesting, weeding, disease detection, soil sensing, seeding, spraying fertilizer and fertilizers, and more, may be carried out by machines that are outfitted with the appropriate sensors. These devices have a great deal of ability to precisely position and guide users inside the work zone.

3.2 Line Following Robots (LFRs) and Their Applications in Agriculture

Line Following Robots (LFRs) are increasingly utilized in agriculture to automate and streamline operations. Their ability to follow a predefined path using optical sensors makes them suitable for tasks like soil sampling and plant monitoring in large fields. Shaikh and Jagtap (2020) discuss the use of Proportional-Integral-Derivative (PID) control in LFRs to enhance path accuracy, crucial for precise navigation and operation in agricultural environments. Additionally, Patel and Reddy (2019) highlight the integration of ultrasonic sensors in LFRs to enable obstacle detection, enhancing their functionality in dynamic agricultural settings. The incorporation of machine

learning techniques, particularly convolutional neural networks (CNNs), further augments LFR capabilities, allowing for the management of complex paths and decision-making processes in real-time, although computational limitations remain a challenge (Singh & Kumar, 2021).

3.3 Soil Analysis

Following soil analysis utilizing appropriate procedures that offer information on nutrient availability for the growth and development of the particular plant, phases of agricultural production, such as sowing and fertilizing, may be completed. It used to be common for farmers to fertilize every plot equally, which is not the best economic strategy nor sustainable from an ecological, environmental, or ecosystem perspective. Plants absorb the amount of nutrients needed for normal growth, and any additional nutrients provided improperly through fertilization may wind up in surface and subsurface waterways where they either encourage the growth of algae in rivers and lakes or contribute to greenhouse gas emissions. (Sikora, 2020)[13]

Phases of agricultural production, such as sowing and fertilization, have to be finished after soil analysis using suitable techniques that reveal nutrient availability for the specific plant's growth and development. While additional artificially given nutrients through fertilization generally evaporate, producing greenhouse gases, or wind up in surface and subsurface waters, which promote algal development in rivers and lakes, plants absorb nutrients in proportions necessary for regular growth. [**14**]. Farmers used to fertilize every plot uniformly, which is unsustainable from an ecological, environmental, and ecosystem standpoint as well as not the best business practice.

Lack of oxygen subsequently causes widespread fish mortality as well as the demise of other aquatic species and animals in their native environments, which ultimately has an impact on

people. Given the above information, new techniques and technologies for georeferenced, real-time soil nutrient sampling are required in order to provide accurate fertilizer input.

3.4 Soil pH and Nutrient Management

Soil pH is a pivotal factor influencing plant health and productivity, as it affects nutrient solubility and microbial activity in the soil. Different crops require specific pH ranges to optimize nutrient uptake; for example, roses thrive in slightly acidic to neutral soil (pH 6.0–7.0), while tomatoes prefer a slightly more acidic environment (pH 6.0–6.8) (University of Hawai‘i at Mānoa, 2000). Nutrient management, particularly the levels of nitrogen (N), phosphorus (P), and potassium (K), is also critical. Fulton et al. (2003) note that appropriate concentrations of these nutrients support various growth stages and plant functions, from vegetative growth to stress resistance.

3.5 Soil Moisture and Electrical Conductivity

Soil moisture and relative humidity significantly influence plant growth by affecting hydration and nutrient transport. Optimal moisture levels vary by plant type but generally range from 40-70%, with relative humidity levels also playing a crucial role in plant health and disease prevention (University of Hawai‘i at Mānoa, 2000; SpringerLink, n.d.). Electrical conductivity (EC) of the soil is another vital measure, indicating soil salinity, which can impact nutrient availability and plant health. For young saplings such as roses, guavas, and tomatoes, maintaining EC levels within certain thresholds is necessary to avoid salt stress and ensure adequate nutrient availability (University of Hawai‘i at Mānoa, 2000).

3.6 Ion-Selective Electrode(ISE)

Based on an ion-selective electrode (ISE), Vernier NO3-BTA is the soil analysis module. There are several methods for determining soil nitrate outside the electrochemical ISE technique, including spectrophotometric/spectrometric and biological methods. Considering it offers the best chance for reliable field nitrate analysis in the near future, we chose to base our soil analysis module on the ISE probe. (Kitić et al., 2022b) [14]These methods need thorough site-specific calibration because of the various optical fingerprints of soil.

As of right now, soil moisture cannot be measured, which might lead to a tiny measurement mistake. To create a homogenous solution, the sample solution is continuously mixed at a slower pace during the measurement. It takes 3.5 minutes for the probe to steady the response during the measuring procedure.

Several benchtop laboratory experiments were carried out to optimize measurements and compare the accuracy of the proposed approach with the reference one before integrating the soil analysis module with the robotic platform.

Plant development must be measured without interfering with its normal growth process since the diameter and height of the plant are crucial in identifying how it reacts to changes in water concentration and long-term growth cases. Many studies have found a relationship between temperature, light, humidity, and hydration status, as well as variations in the diameter and height of plant stems. Finding a plant's diameter and height is therefore another approach in this study.

Nitrate Ion Selective Electrode is an ideal choice for determining the concentration of nitrate in aqueous samples. We are able to produce low-volume, dependable, and high-quality nitrate ISEs because to our creative research and application of the latest nanotechnology.

Almost all research on water quality includes nitrate content as an important measure. The total amount of nitrates can rise as a result of acidic rain, fertilizer runoff from farms, and animal or plant waste. This ion has uses that go beyond assessing the quality of water, though. In soil extracts, soils, and plant tissue, nitrate ISEs may also be used to assess nitrate ion selective electrodes.

3.7 C3088 Camera with MATLAB GUI

In this proposal, a connection between a computer and a C3088 smart camera has been developed. Sobel Edge Detection is used with a MATLAB GUI to get the height, maximum width, and minimum width of a plant picture. In order to identify edges in a picture, the Sobel edge detection operator measures the spatial gradient in two dimensions and highlights the areas where the gradient is highest. The main advantages of this C3088 camera module are its digital video connections and its digital output, which offers a constant 8–16 bit data stream. (Kaur & Singh, 2013)[15] Usually, it is employed to determine the estimated absolute gradient magnitude of an input picture at each location. C3088 can be used to take pictures of plants at various times throughout the course of the day.

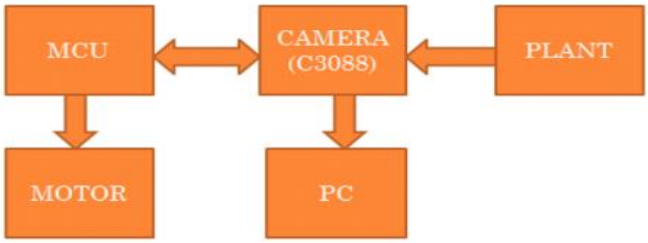


Figure 3.1:Block diagram of MATLAB Implementation

The different properties of a plant may be automatically measured and tracked with the help of these applications. According to the results, the method is accurate in measuring variations in plant growth, such as variations in diameter and height. That way, when the self-governing robot finds plants of the same kind that differ in height, nothing will go wrong.

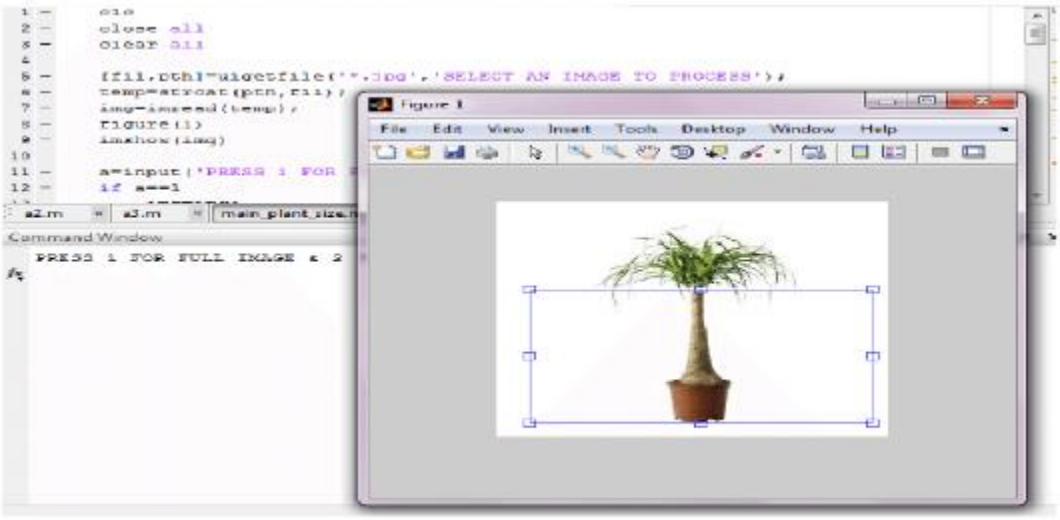


Figure- 3.2: Selection of cropped Image

As Sobel edge detection has been used to calculate the height and breadth of a plant picture using a MATLAB graphical user interface, there will be two distinct kinds of options available for choosing a plant snapshot. The method yields the plant's stem height as well as its maximum and lowest pixel width. It is possible to trim the image or use the entire image.

3.8 pH Sensor

The geographical variation of soil pH should be sufficiently addressed to enhance precision agricultural management systems since it is a critical parameter for the yield of crops. The pH of the soil affects plant development as well as the chemical, biological, and physical characteristics and processes of the soil.

To calculate how much fertilizer is required, the pH level of a field—which is established by measuring the concentration of hydronium ions (H+)—is frequently tested in laboratories.

Potentiometric sensors known as ion-selective electrodes (ISEs) reduce matrix interferences through the use of a selective membrane (Murthy & Nagaswarupa, 2022)[16]. ISEs that respond to the concentration of H+ in a solution are most commonly seen in pH electrodes, which have a thin glass membrane. In addition, gases in solution such as ammonia, carbon dioxide, nitrogen oxide, and oxygen can be detected, as well as other characteristics such as fluoride, bromide, nitrate, and manganese. However, ISEs have some drawbacks too. These include issues with electrode training and a lack of selectivity and sensitivity.

3.9 Laser-Based Arrays for Height Measurement

There has been plenty of research regarding the autonomous height detection of plants and crops. Several methods, such as image processing, LIDAR, Ultrasonic Sensors, and Laser-Based arrays, have been used to measure the height of plants. The use of an Ultrasonic sensor provides less accurate measurements. On the other hand, the image processing techniques are time-consuming (Zhang & Grift, 2012)[17]. Even though LIDAR can provide accurate results, it is very expensive, which makes it non-profitable for practical use. The use of laser-based arrays can provide accurate results at a significantly lower price. It uses a simple algorithm where laser transmitters transmit lights that are received by the receivers. If there is an obstacle between the receiver and transmitter, it can detect the object. Thus, the height is measured. It is important to ensure that the lights emitting from the lasers do not collide with each other and that the receivers only receive light from the transmitter in front of them.

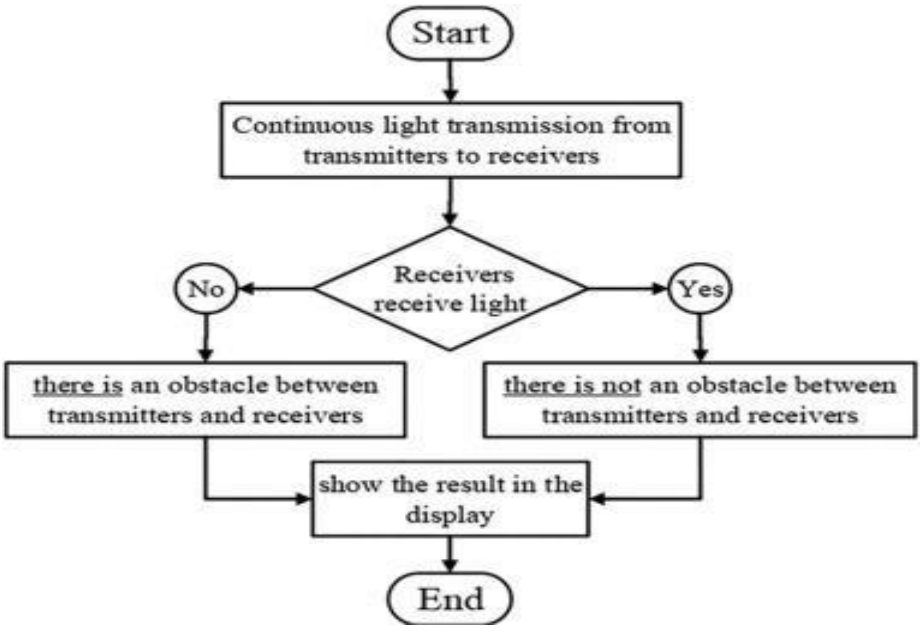


Figure- 3.3: Workflow for using the laser to detect obstacles

Additionally, it does not require the use of a computer. The data processing method can be done by using a microcontroller. Insensitivity to the density of crops can be considered its primary shortcoming. But further improvements and research can make this technology more effective.

3.10 An improved pistachio detection approach using YOLO-v8 Deep Learning Models

3.10.1 Object Detection Evolution

The study places itself in the larger context of object detection, which has advanced significantly since deep learning became available. Traditional machine learning approaches and handmade features played a major role in the early methods. Convolutional neural networks (CNNs), on the other hand, completely changed the field and made it possible for detecting systems to become more precise and effective.

3.10.2 YOLO Model Series

Real-time object detection is a well-known feature of the YOLO (You Only Look Once) model series. Bounding box and class probabilities were predicted simultaneously in the first YOLO model, which offered a single-stage detection approach. The accuracy and speed of later versions, such as YOLOv3, YOLOv4, and YOLOv5, increased gradually. In order to further improve performance, YOLOv8, which is the most recent version utilized in this study, has unique architectural improvements.

3.10.3 Applications in Agriculture

Agriculture has seen a significant increase in the use of deep learning models, especially for crop and pest identification. The effectiveness of these models has been shown in earlier studies on a variety of agricultural activities, including weed identification, disease identification, and fruit counting. By focusing on pistachio detection—a specialized but vital use in the agricultural industry—this study expands on that framework.

3.10.4 Challenges in Pistachio Detection

Due to the small size, diverse look, and complicated surroundings in which they are frequently encountered, pistachio detection poses special obstacles. High precision has been difficult to obtain in such situations for traditional image processing approaches. By utilizing YOLOv8's improved feature extraction capabilities and higher detection accuracy, these issues are intended to be addressed.

3.10.5 Previous Work on Pistachio Detection

Several machine learning and deep learning techniques have been used in previous research on pistachio detection. Although these experiments frequently suffered from issues with handling occlusions, illumination fluctuations, and other real-world difficulties, they did set the foundation for future research aimed at improving detection accuracy. These constraints are cited in this study as justification for using the more reliable YOLOv8 model.

3.10.6. Methodological Improvements

A number of methodological advancements over earlier studies are highlighted in this study. The YOLOv8 architecture has been adjusted for the unique properties of pistachios, improved data augmentation techniques have been used to improve the model's generalization, and sophisticated training procedures have been utilized to maximize performance.

3.10.7 Performance Metrics

Standard performance criteria including mean Average Precision (mAP), recall, and precision are used in the paper's evaluation of the suggested methodology. In terms of accuracy and computing efficiency, comparisons with earlier models show that YOLOv8 performs better. The research offers a thorough and enhanced method for pistachio detection by incorporating these developments, which has great promise for automated agriculture systems.

3.11 Soil pH and Its Role in Plant Growth

Soil pH is a critical factor influencing nutrient availability, microbial activity, and overall plant health. Maintaining an optimal pH range ensures that essential nutrients remain accessible to plants, thereby promoting healthy growth. For instance, soil pH affects the solubility of minerals and nutrients; in highly acidic soils, elements like aluminum and manganese can become more available and potentially toxic to plants, while essential nutrients like phosphorus may become less available (HORIBA, n.d.). Therefore, monitoring and adjusting soil pH is vital for optimal plant nutrition and growth.

3.12 Nitrogen (N), Phosphorus (P), and Potassium (K) Levels

Nitrogen, phosphorus, and potassium are primary macronutrients essential for plant development. Nitrogen is crucial for vegetative growth and chlorophyll production, phosphorus supports root development and energy transfer, and potassium enhances overall plant health and stress resistance. Proper management of these nutrients is vital for optimal plant growth and yield. For example, a study on Juniper plants demonstrated that appropriate NPK fertilization significantly improved growth and nutrient uptake, highlighting the importance of balanced fertilization strategies (Alharbi & Lee, 2022). Additionally, the synthesis of effective substances in medicinal plants is affected by various soil factors, including macronutrients like N, P, and K, underscoring their role in plant growth and development (Zhang et al., 2022).

3.13 7-in-1 Soil Sensors in Agricultural Monitoring

Advancements in agricultural technology have led to the development of multisensory soil monitoring systems, such as 7-in-1 integrated sensors. These devices measure parameters including soil temperature, moisture, electrical conductivity, NPK levels, and pH, providing comprehensive real-time data for effective soil management. The integration of these sensors facilitates precise monitoring and optimization of soil conditions, thereby enhancing crop productivity and sustainability (Zhang et al., 2021). However, the accuracy and reliability of these sensors in measuring certain parameters, such as NPK levels, require further validation to ensure their effectiveness in various soil conditions (Zhang et al., 2021).

Chapter 4

Design and Development of the Autonomous Agricultural Robot

The autonomous agricultural robot developed in this research is a multi-functional platform designed for plant disease detection and soil damage level. The robot integrates various hardware components, including microcontrollers, sensors, actuators, and power management units, to achieve autonomous navigation, soil sampling, and real-time data collection. This chapter provides an in-depth discussion of the robot's hardware architecture, structural build, and functionality.

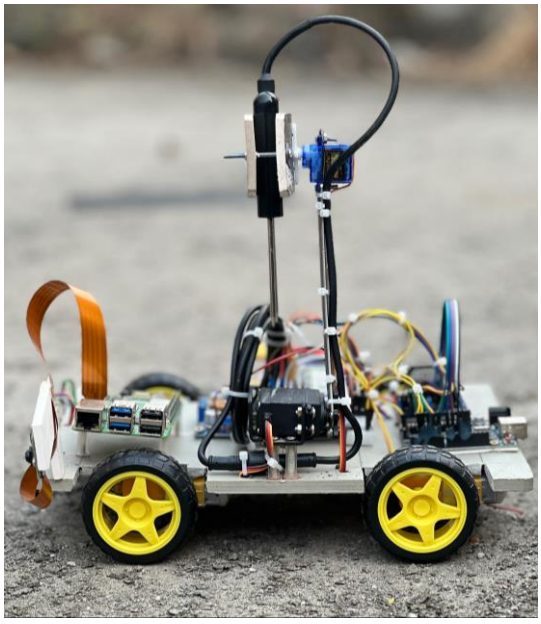


Figure- 4.1: Autonomous Agricultural Robot

4.1 Structural Overview of the Robot

The autonomous agricultural robot is built on a **durable yet lightweight chassis**, designed to ensure stability and mobility across diverse agricultural terrains. The structural framework consists of a **base platform**, which securely houses and supports various electronic components, including the **Raspberry Pi 5, Arduino Uno, power management system, motor drivers, and soil sensors.** The chassis is constructed from a robust material that provides strength while maintaining a lightweight profile, allowing for efficient movement across fields. The **wheel system** includes two **DC motor-driven wheels** for propulsion and maneuverability, while two passive caster wheels at the rear provide balance and stability during movement. On the front side of the robot, a **servo-actuated soil sampling mechanism** is mounted, allowing the precise deployment of the **7-in-1 soil sensor**, which measures essential soil properties such as pH, moisture, and nutrient levels. The **Pi camera**, mounted at the top via a flexible ribbon cable, is positioned to capture images of plants in real time for disease detection using the YOLOv11 model. The robot's structure also includes an **electronic mounting platform**, where the Arduino Uno acts as the control unit, interfacing with IR sensors for navigation, and communicating with the soil sensor through the RS485 protocol. Power to the system is supplied by a high-capacity **LiPo battery**, which is efficiently regulated through a **buck converter** to provide appropriate voltages for various components. All wiring and connections are strategically organized to avoid

interference and ensure a clean, efficient layout for long-term operation in agricultural environments.

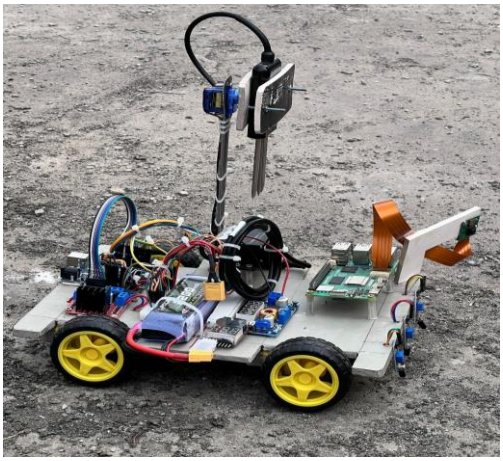


Figure- 4.2: Autonomous Agricultural Robot (Upper Angle)

4.2 Hardware Components

The robot consists of several key hardware components that enable its autonomous functionality. The primary components include:

4.2.1 Raspberry Pi 5

- **Central Processing Unit:** The Raspberry Pi 5 acts as the robot's central processing unit, managing complex computations and running the YOLOv11-based plant disease detection model. It captures images from the Pi camera and processes them in real time for detecting plant diseases with high accuracy.
- **Specifications and Capabilities:** Equipped with a quad-core ARM Cortex-A76 processor and 8GB of RAM, the Raspberry Pi 5 efficiently handles deep learning tasks. Its USB 3.0 and GPIO support enable seamless interfacing with sensors and other components, while the Raspberry Pi OS provides a stable operating environment.
- **Robot Functions:** In this setup, the Raspberry Pi 5 focuses on image processing, AI inference, and communication with peripheral components like the Arduino for sensor data acquisition. It also logs data locally for further analysis, without relying on cloud connectivity or IoT functionalities.

4.2.2 Arduino Uno

The Arduino Uno is the robot's primary microcontroller, handling sensor data collection and actuator control. It operates on the ATmega328P microcontroller with a 5V operating voltage and a 16 MHz clock speed, featuring 14 digital I/O pins (6 PWM outputs) and 6 analog inputs. The Arduino collects real-time data from soil sensors like pH, NPK, and moisture, controls IR sensors for navigation, and manages motor drivers and servo actuators. Communication with the Raspberry Pi is facilitated via USB serial or UART (with RS485 support via an external module), ensuring seamless data exchange for decision-making.

4.2.3 Buck Converter

The DC-DC buck converter steps down the voltage from the primary battery to suitable levels for different components. It takes an input voltage of 12V-24V and provides stable 5V and 9V outputs with up to 5A current. This ensures reliable power for the Raspberry Pi, sensors, and other components, preventing system crashes and optimizing power distribution.

4.2.4 Battery LiPo Battery

The robot is powered by a 12V (3S) Lithium Polymer (LiPo) battery with a capacity of 2800mAh and a discharge rate of 30C. This battery provides ample power for the motors, sensors, and onboard processing units, ensuring reliable and sustained operation. It supplies regulated energy to the Raspberry Pi, Arduino, and motors, enabling the robot to function effectively during extended field activities.

4.2.5 IR Sensors

Infrared (IR) sensors enable the robot to autonomously navigate by detecting field markers and avoiding obstacles. Operating at 5V, these sensors offer a detection range of 2-10 cm with analog and digital outputs, providing feedback to the Arduino for movement control.

4.2.6 DC Motors

The robot uses two 12V DC motors with 1.5 Nm torque and 150 RPM speed, driven by the L298N motor driver. They facilitate forward/backward movement, assist in turning using IR sensor feedback, and maintain stability over uneven terrain.

4.2.7 Servo Actuator

The robot uses two servo motors for different tasks. A larger servo motor with 2.5 kg/cm torque and 0-180° rotation is responsible for deploying and retracting the soil sensor mechanism, ensuring accurate positioning for sampling. A smaller servo motor is used for auxiliary functions, such as precise adjustments. Both servos operate at 5V, providing reliable and efficient actuation

4.2.8 RS485 Communicator

The RS485 module enables robust, long-distance data transmission between the Arduino and soil sensor. With a 1200-meter range, 9600 bps baud rate, and differential signal transmission, it ensures stable communication in noisy environments.

4.2.9 7-in-1 Soil Sensor

This sensor measures soil pH, moisture, NPK levels, temperature, and conductivity. It provides real-time data for soil health analysis, helping farmers take informed actions to maintain optimal soil conditions.

4.2.10 Pi Camera Module

The 12 MP Pi Camera captures high-resolution images for real-time disease detection using the YOLOv11 model. With a 62.2° horizontal field of view and up to 30 FPS, it analyzes plant health, supporting the robot's navigation and data collection systems.

4.2.11 Motor Driver (L298N Dual H-Bridge)

The L298N motor driver controls the robot's two DC motors, providing bidirectional movement and speed regulation. Operating at 5V-35V and up to 2A per channel, it uses PWM for smooth control and efficient power management from the 12V LiPo battery.

4.3 Circuit Diagram Explanation

The circuit diagram of the robot (as provided in the attached image) illustrates the connections and interactions between various hardware components. The key components and their connections are as follows:

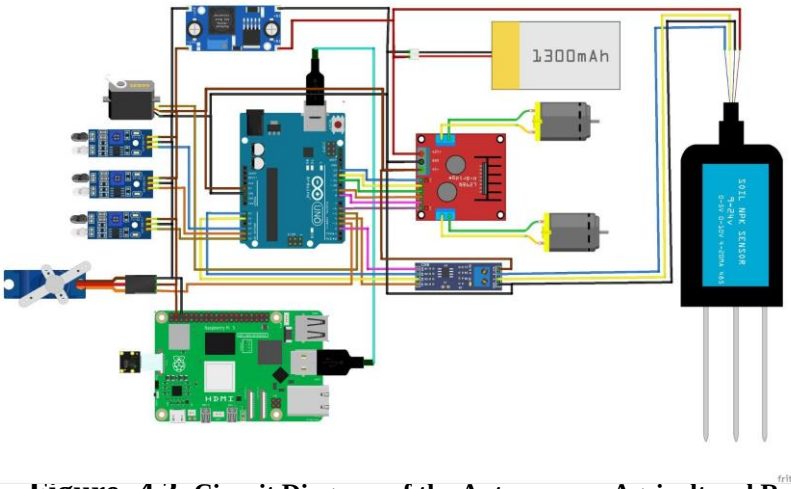


Figure- 4.3: Circuit Diagram of the Autonomous Agricultural Robot

1. **Power Distribution:** The LiPo battery serves as the primary power source, delivering energy to the buck converter. The buck converter steps down the voltage and provides regulated power to critical components, including the Raspberry Pi, Arduino, and motor driver, ensuring stable operation of the system.
2. **Microcontroller Connections:** The Arduino is responsible for collecting data from the IR sensors and soil sensors, as well as managing the servo actuator. Meanwhile, the Raspberry Pi processes the camera feed and sends control commands to the Arduino, enabling efficient coordination between components.
3. **Motor Control:** The motor driver, specifically the L298N module, is connected to the Arduino to manage the robot's movement. Navigation paths are determined based on commands from the Raspberry Pi, which processes feedback from the IR sensors to ensure precise control.
4. **Sensor Integration:** The soil sensor communicates with the Arduino through the RS485 module, facilitating robust data transmission. Additionally, the Pi camera captures images used for disease detection, with processing powered by the YOLO model for accurate analysis and decision-making.

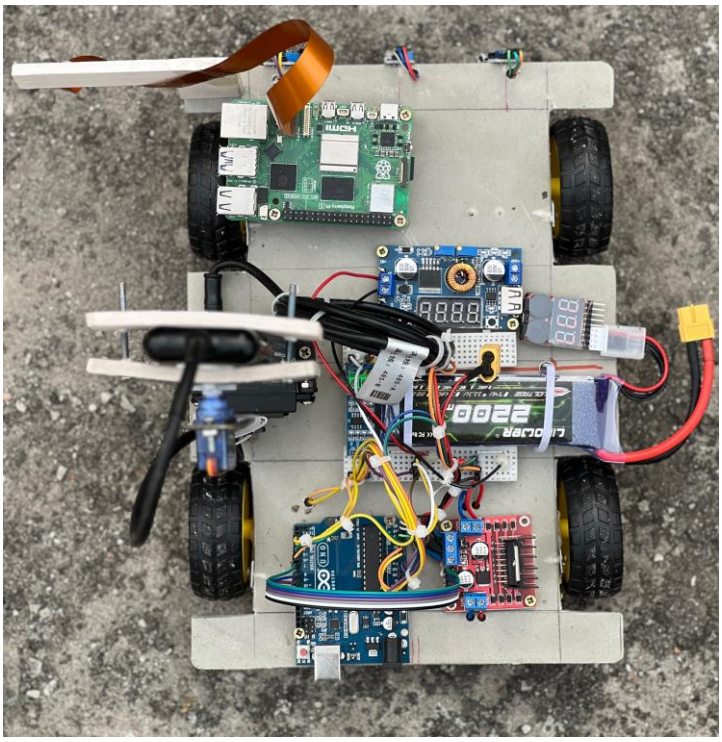


Figure- 4.4: Autonomous Agricultural Robot (Eagle Eye View)

4.4 Assembly and Build Process

The assembly and build process of the autonomous agricultural robot is a comprehensive approach that involves integrating mechanical, electrical, and software components to ensure smooth functionality in agricultural environments.

4.4.1 Chassis and Mechanical Assembly:

The robot's structure is built upon a robust yet lightweight chassis, designed to support all electronic components while maintaining stability and maneuverability across different terrains.

The assembly begins by securely mounting the drive system, which includes two DC motors for propulsion and caster wheels for balance. The motor driver and battery holder are installed strategically to optimize weight distribution. The soil sampling mechanism, equipped with a servo motor, is attached to the front, allowing precise deployment of the 7-in-1 soil sensor. The Pi camera is positioned at an ideal height and angle to capture high-resolution images of crops for disease detection.

4.4.2 Electrical Connections and Wiring:

A critical aspect of the assembly involves setting up the power distribution system to ensure the proper operation of all components. The LiPo battery, along with a buck converter, provides regulated power to the Raspberry Pi, Arduino Uno, motor driver, and sensors. The wiring process includes establishing communication between the microcontrollers and peripheral components such as IR sensors for navigation, the RS485 module for soil data collection, and the motor driver for movement control. Careful cable management and insulation techniques are employed to prevent signal interference and enhance durability in field conditions.

4.4.3 Software Deployment:

Once the hardware assembly is complete, the software components are installed and configured to enable the robot's autonomous functions. The Raspberry Pi is programmed with the necessary software stack, including the YOLOv11 deep learning model for plant disease detection, image processing scripts, and data logging functionalities. The Arduino Uno is programmed to handle sensor readings, motor control, and communication with the Raspberry Pi. Data collection and logging are implemented in an Excel-based system for future analysis and decision-making.

4.4.4 Testing and Calibration:

Before deploying the robot in the field, extensive testing and calibration are performed to ensure each component functions correctly. The navigation system is tested using predefined paths to verify the accuracy of the IR sensors and motor control algorithms. Soil sensor calibration is carried out to validate data accuracy across different soil conditions, ensuring precise pH, moisture, and nutrient level measurements. The Pi camera's positioning and image quality are fine-tuned to optimize detection accuracy.

4.4.5 Field Deployment and Final Adjustments:

After successful testing, the robot is deployed in actual agricultural conditions to assess its performance in real-world scenarios. The system undergoes further optimization, such as refining sampling intervals, adjusting sensor thresholds, and ensuring smooth operation over uneven terrain. Farmers and operators are trained to handle basic troubleshooting and maintenance, ensuring long-term usability and effectiveness.

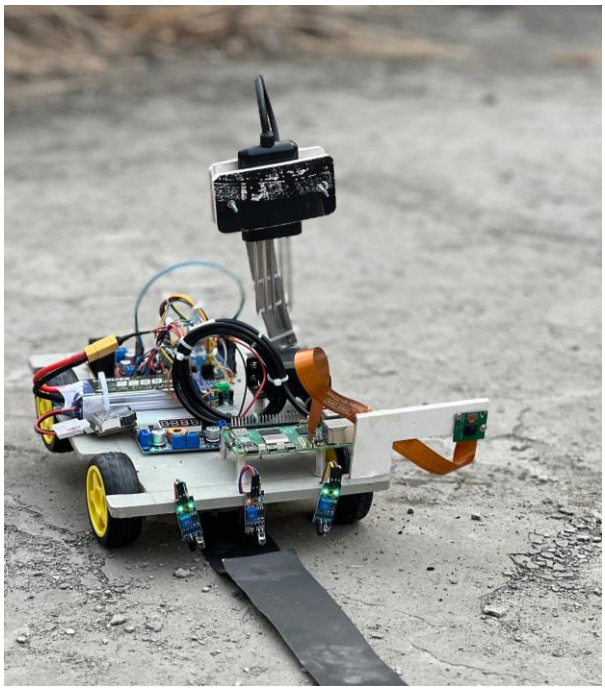


Figure- 4.5: Autonomous Agricultural Robot on Field Track using IR Sensor

Chapter 5

Methodologies

Precision agriculture is an emerging approach that leverages advanced technologies to enhance productivity and sustainability in farming. This research presents an integrated system combining YOLOv11-based plant disease detection with sensor-based soil health monitoring, utilizing a robotic platform equipped with a Pi camera and a 7-in-1 soil sensor. The proposed system aims to provide an automated, real-time agricultural monitoring solution by capturing plant and soil health data, processing it with deep learning and analytical algorithms, and offering actionable insights to farmers.

The plant disease detection module utilizes deep learning to analyze plant images, detecting diseases such as Healthy, Scab, Rust, and Multiple Disease, while the soil monitoring system employs various sensors to measure critical parameters such as pH, conductivity, humidity, temperature, and NPK levels (Nitrogen, Phosphorus, and Potassium). Both systems are integrated into a robotic platform that autonomously navigates through agricultural fields, collects data at designated points, processes it, and stores it for further analysis.

This paper presents a detailed discussion of the methodologies involved in this system, including data acquisition, disease detection, soil analysis, navigation control, and data storage. The paper also discusses the intercommunication between these modules and their integration into a holistic solution for precision farming.

5.1 Detailed Description of Implemented Methodology Integration and Intercommunication

The integration of plant disease detection and soil monitoring is structured around a sequential and iterative workflow that ensures efficient data acquisition, processing, and decision-making. The following sections provide an in-depth discussion of the step-by-step process and how the methodologies interconnect.

5.1.1 System Initialization and Hardware Setup

The workflow commences with the initialization of the system components, ensuring that all sensors, actuators, and machine learning models are correctly configured for operation.

The **Plant Disease Detection System** begins by loading the trained YOLOv11 deep learning model onto the Raspberry Pi, enabling it to perform real-time plant disease detection. Essential Python libraries such as **ultralytics**, **OpenCV**, and **NumPy** are imported to handle the model's operation, image processing, and numerical tasks. The system configures the Pi camera to capture high-resolution images of plants, ensuring that the input data for disease detection is clear and detailed. These components work together to create an automated solution for identifying plant diseases by analyzing images captured through the Pi camera.

The **Soil Monitoring System** begins with the Arduino microcontroller initializing serial communication with the Raspberry Pi via USB, enabling data exchange between the two devices. The system sets up RS485 communication to interface with the 7-in-1 soil sensor,

allowing the collection of key soil parameters such as pH, conductivity, temperature, humidity, and NPK values. Once the data is collected, the system establishes a connection to store the gathered soil information in an Excel sheet, ensuring easy access and analysis for agricultural monitoring. This integration allows continuous soil monitoring, providing valuable insights into the soil health and supporting informed decision-making for agricultural management.

The **Robot Navigation** system begins with the initialization of servo motors, which are essential for the soil sampling process. These servos are calibrated to move into specific positions for data collection when required. The line-following IR sensors are calibrated to detect and follow navigation paths, ensuring the robot can autonomously traverse its environment. The motor control logic is then prepared to handle movement commands, allowing the robot to move forward, turn, and stop as needed. This setup enables the robot to navigate autonomously, reaching designated points for soil sampling and performing its tasks efficiently.

5.1.2 Autonomous Navigation and Plant Disease Detection

The **Line Following** system relies on IR sensors to detect the presence or absence of a line on the ground, which serves as the designated path for the robot. The robot continuously reads the sensor inputs and adjusts its movement to stay aligned with the line, ensuring it follows the correct route. If an obstacle is detected, the robot halts briefly to evaluate the situation and then retries its movement after a short delay. This adaptive process ensures that the robot can stay on course, navigate around obstacles, and continue its mission without veering off the designated path.

In the **Image Processing and Disease Detection** system, live video frames are captured by the Pi camera to provide real-time visual input. Each frame is then processed by the YOLOv11 model, which performs inference to identify any potential plant diseases. Once a disease is detected, it is categorized into a specific type, and a health score is assigned based on the severity of the disease. The system then annotates the images with bounding boxes around the detected diseases, displaying the associated confidence scores. These annotated images are shown to the user, providing immediate feedback on the detected plant health conditions.

In the **Data Storage** system, the detected disease results, along with relevant timestamps and locations, are stored in an Excel file for easy tracking and analysis. This allows for organized documentation of the plant health data over time. Additionally, the system is programmed to generate alerts if critical disease levels are detected, ensuring that timely intervention can be taken. These alerts help stakeholders respond quickly to prevent the spread of diseases and maintain the health of the plants.

5.1.3 Soil Sampling and Data Acquisition

Upon reaching predefined sampling locations, the robot stops and triggers the soil sampling process. In the **Soil Parameter Collection** system, the servo motor extends to position the soil sensor into the ground, allowing it to collect critical soil data. The Arduino sends commands to the 7-in-1 soil sensor to retrieve measurements for various parameters, including pH, conductivity, humidity, temperature, and NPK levels. Once the data is collected, it undergoes

processing and validation to ensure accuracy and reliability. This system provides vital soil information that can be used for monitoring soil health and optimizing agricultural practices.

In the **Real-Time Analysis and Alerts** system, threshold-based warnings are triggered when extreme values are detected, such as high temperatures or low nitrogen levels in the soil. If the data exceeds or falls below predefined thresholds, alerts are generated to notify the user of potential issues. For example, if the soil temperature is too high, an alert like **"Warning: High Temperature of Soil!"** will be displayed on the user interface. This real-time feedback ensures that any critical conditions are promptly addressed, helping to maintain optimal plant growth and health.

In the **Data Logging** system, sensor readings are continuously appended to an Excel file, allowing for the persistent storage of soil parameter data over time. To assess the health of the soil, the system calculates the **soil damage score** using a normalization technique. This process compares the current sensor readings against ideal reference values, helping to quantify the extent of soil damage. By tracking these scores, the system provides valuable insights into soil conditions and can help guide decisions regarding necessary interventions or improvements to optimize soil health.

5.1.4 Calculation of Soil Damage Level

Step 2 of Algorithm 1 is employed to quantify the soil's health by comparing the collected sensor data with standard reference values. The process begins with **difference calculation**,

where the deviation of each soil parameter (e.g., pH, conductivity, humidity, etc.) from its reference value is computed. Next, **normalization** is applied to these deviations to ensure consistency in comparison across different parameters and units of measurement. The **cumulative damage scoring** step combines the normalized deviations to calculate an overall soil damage level, which is then categorized into levels of soil health (e.g., healthy, moderate damage, severe damage). Finally, the computed score is **stored** in an Excel file for trend analysis and future reference. If the soil damage score exceeds acceptable thresholds, **alerts** are triggered to notify users of the critical condition, prompting immediate intervention or corrective actions to prevent further deterioration of soil health.

5.1.5 Data Visualization and Reporting

The system offers comprehensive **visual feedback** and **historical data insights** through several key features. Real-time video feeds are annotated with plant disease detections, allowing users to monitor plant health dynamically. Periodically, reports are generated from the Excel file, summarizing soil health trends and providing valuable insights into changes in soil conditions over time. Additionally, **graphical visualizations** are created using tools like Matplotlib, offering clear and intuitive charts that present long-term field health data. These visual tools help users quickly identify patterns, trends, and areas that may require attention, facilitating informed decision-making for effective crop and soil management.

5.1.6 Deployment and Future Improvements

The **deployment phase** ensures that the system is ready to operate effectively in real agricultural environments, enabling continuous monitoring and management of plant health and soil conditions. Future improvements for the system focus on expanding its capabilities, including **integration with cloud-based platforms** to facilitate remote monitoring, allowing users to access real-time data from anywhere. Another area for enhancement is the **model accuracy**, which will be improved by gathering more data and fine-tuning the model for better disease detection and soil analysis. Additionally, the development of **predictive analytics** will enable the system to forecast potential issues with soil and plants, allowing farmers to take proactive steps and optimize agricultural outcomes. These advancements will make the system more powerful and versatile in supporting sustainable farming practices.

5.2 Algorithms of the Entire System

The complete functionality of the system is driven by three core algorithms, each playing a pivotal role in achieving precision, efficiency, and reliability.

5.2.1 Algorithm 1: Soil Data Acquisition and Storage

The **objective** is to read sensor data from an Arduino, process it for relevant parameters, and store the processed data in an Excel file for further analysis and record-keeping. This process involves receiving real-time data from connected sensors, interpreting the readings (such as soil moisture, pH, temperature, etc.), and organizing them in a structured format. The data is then saved into an Excel file, which allows for easy tracking, trend analysis, and future reference. This system ensures that valuable sensor information is continuously logged, providing insights into environmental conditions and enabling better decision-making.

Step 1: Reading Sensor Data and Saving to Excel

The primary objective of this research is to collect soil parameter data from sensors connected via an Arduino device, process it, and store it for further analysis. Step 1 outlines a systematic process to achieve this goal. Initially, the system establishes communication with the Arduino device through a serial interface, enabling continuous streaming of sensor data. As the data is received, each sensor reading is parsed and stored in a dictionary for further processing. The data is then evaluated for critical thresholds in real-time, allowing the system to promptly detect any abnormal soil conditions. Subsequently, the sensor data is appended to an Excel file, along with the calculated soil damage level, for long-term storage. This is designed to handle file initialization, real-time data processing, and saving of the data in an iterative manner, ensuring reliability and scalability for continuous data collection.

The algorithm begins by initializing the Excel file. If the file already exists, it is loaded; otherwise, a new workbook is created with the necessary headers. The system then establishes serial communication with the Arduino device using the specified serial port, baud rate, and timeout settings. A dictionary is created to store values for various soil parameters such as Soil pH, Conductivity, Humidity, Temperature, Nitrogen, Phosphorus, and Potassium. The system continuously reads data from the Arduino's serial port, parsing each line to extract the parameter labels and their corresponding values. These values are then updated in the sensors dictionary. As part of real-time monitoring, the system checks if any parameters exceed predefined thresholds. For example, if Soil Temperature exceeds 40°C, a high-temperature warning is triggered, and if Nitrogen levels fall below 100 ppm, a low-nitrogen warning is issued.

Next, the algorithm calculates the soil damage level, which compares the collected sensor data with reference values to compute the damage score. The results, including the timestamp, sensor readings, and calculated soil damage level, are appended as a new row in the Excel file. The workbook is then saved to ensure that the latest data is stored securely. After each iteration, the sensors dictionary is reset, and the process continues until manually interrupted, providing a reliable and continuous collection of soil data. This approach ensures that the system can effectively monitor soil health over time, offering valuable insights for agricultural management.

Step 2: Calculation of Soil Damage Level

The objective of this step is to compute the overall soil damage level by normalizing the differences between reference values and measured sensor values. Understanding how variations in soil parameters affect plant health is crucial to this research. This step calculates a scalar value called the "Soil Damage Level," which quantifies the deviation of measured sensor values from the ideal reference values. This calculation involves normalizing the differences between reference values and sensor readings, allowing for consistent comparisons across different datasets. The process begins by initializing the necessary variables, including a total accumulator set to zero. Then, for each parameter, it calculates the absolute difference between the reference value and the average reference value, followed by normalization of this difference. Similarly, it calculates the absolute difference between the sensor reading and the average reference value, then normalizes it. The normalized differences for both the reference and sensor values are then compared, and the absolute difference between them is calculated. This normalized difference is

added to the total value, which accumulates across all parameters. After processing all parameters, the total value is returned as the soil damage level. This scalar value encapsulates the overall health of the soil based on deviations from ideal reference values, allowing for a comprehensive assessment of soil conditions. The calculated soil damage level serves as a critical metric for evaluating soil health and informing decision-making in agricultural practices.

Step 3: Warning Generation for Critical Parameters

The objective of this step is to generate real-time warnings for parameters that exceed predefined thresholds, which is crucial for maintaining soil health and ensuring timely intervention. It continuously monitors specific soil parameters, such as soil temperature and nitrogen levels, and compares them against critical threshold values. When a parameter exceeds its defined threshold, a warning message is immediately triggered to alert the user. For example, if the soil temperature rises above 40°C, the system will issue a warning stating, "Warning: High Temperature of Soil!" Similarly, if nitrogen levels fall below 100 ppm, the system will issue a warning message: "Warning: Nitrogen (N) is too low!" These real-time warnings provide a proactive layer of monitoring, allowing for early detection of critical conditions and enabling corrective actions before they affect soil health. The system continuously evaluates the incoming sensor data, ensuring that as new readings become available, the monitoring process remains ongoing and up-to-date. This approach allows for dynamic and immediate response to any changes in the soil environment, enhancing the overall effectiveness of soil management.

Rose	Parameter	Ideal Amount	Warning call	Message
1	Soil pH	6.5	5	pH Low
2	Soil Conductivity	1000	400	Conductivity Low
3	Soil Humidity	60	40	Humidity Low
4	Soil Temperature	18	>40	Temperature High
5	Nitrogen	100	<40	Nitrogen Low
6	Phosphorus	35	<15	Phosphorus Low
7	Potassium	180	<100	Potassium Low

Table- 5.1: Soil Parameter Thresholds and Warning Levels for Rose Sapling Tree

Guava	Parameter	Ideal Amount	Warning call	Message
1	Soil pH	6.25	5	pH Low
2	Soil Conductivity	1500	800	Conductivity Low
3	Soil Humidity	70	40	Humidity Low
4	Soil Temperature	25.5	>40	Temperature High
5	Nitrogen	15	<7	Nitrogen Low
6	Phosphorus	15	<5	Phosphorus Low
7	Potassium	200	<150	Potassium Low

Table- 5.2: Soil Parameter Thresholds and Warning Levels for Guava Sapling Tree

Tomato	Parameter	Ideal Amount	Warning call	Message
1	Soil pH	6.4	5	pH Low
2	Soil Conductivity	2500	900	Conductivity Low
3	Soil Humidity	70	50	Humidity Low
4	Soil Temperature	21	>40	Temperature High
5	Nitrogen	200	<100	Nitrogen Low
6	Phosphorus	40	<30	Phosphorus Low
7	Potassium	250	<100	Potassium Low

Table- 5.3: Soil Parameter Thresholds and Warning Levels for Tomato Sapling Tree

Step 4: Initialization of Excel File

The objective of this step is to initialize or create an Excel file for storing sensor data, ensuring that the data is systematically organized for long-term tracking and analysis of soil parameters. This step is crucial because it lays the foundation for consistent data recording, which will later be used to monitor and evaluate the soil health over time. It first checks if the specified Excel file already exists. If it does, the then it will load the existing workbook; otherwise, it creates a new workbook. After ensuring the correct file setup, inserts a header row with the necessary column names: Timestamp, Soil pH, Conductivity, Humidity, Temperature, Nitrogen, Phosphorus, Potassium, and Soil Damage Level. These headers are vital for ensuring the data is organized and clearly labeled, making it easy to understand and access. Once the headers are added, the workbook is saved, completing the initialization process. This ensures that subsequent sensor

data can be seamlessly added in the future, with all necessary parameters properly aligned for analysis. The result of this process is either a newly created or an updated Excel workbook with the correct structure for storing and tracking the soil data.

Integration of First Four Steps

This algorithm collectively forms a robust framework for soil parameter monitoring and analysis. The workflow begins with **step 4**, which sets up the data storage infrastructure. Then, **step 1** handles real-time data acquisition, processing, and storage. During this process, **step 2** calculates the Soil Damage Level to provide a holistic assessment of soil health, while **step 3** ensures real-time warnings for critical parameters. Together, these steps ensure a comprehensive and efficient methodology for soil monitoring, making the system reliable and adaptable for various applications in agricultural research.

Step 5: Continuously read and parse sensor data.

The efficient monitoring of soil health parameters is essential for sustainable agricultural practices and precision farming. This system employs RS485 communication protocol to interface with soil sensors for the acquisition of vital parameters, including pH, conductivity, humidity, temperature, and nutrient levels (NPK). The methodology described herein outlines a series of sections designed to ensure reliable communication, accurate data acquisition, and systematic processing of the received data. These sections are integrated to create a robust framework for soil health monitoring, addressing challenges such as data integrity, communication errors, and scalability.

The following sections provides step-by-step processes and elucidating how they collaborate to achieve the overarching goal of soil parameter monitoring.

Section 1: Soil Parameter Reading (SoilRead)

Soil Parameter Reading (SoilRead), is the central process responsible for monitoring and managing the retrieval of soil parameters. It ensures that all parameters—such as pH, conductivity, humidity, temperature, and NPK—are read, processed, and displayed consistently. This step starts by invoking the relevant steps for each parameter (such as section 2 for pH, conductivity, etc., and section 3 for NPK readings) to initiate the query process.

The first step is to **initialize the reading process**, where the SoilRead calls the specific parameter-reading sections for each of the relevant parameters. For each parameter, it **queries the relevant sensor** using the appropriate command. Then it ensures that the sensor data is valid and processable by **validating the data**. This validation ensures that there are no erroneous readings or communication failures.

After successfully retrieving and validating the data, **scales the raw readings** to human-readable values. This is done by applying scaling factors provided in the sensor datasheets, which convert the raw values into meaningful values (e.g., pH, conductivity, temperature, etc.). These scaled results are then **displayed on the serial monitor**, enabling real-time monitoring of the soil's condition.

To ensure robustness, it includes **error handling**: if a parameter fails to return valid data, the system logs an error message indicating which parameter failed, allowing for quick troubleshooting and maintenance. Finally, it introduces a **delay (e.g., 5 seconds)** after each

complete reading cycle before starting the next round of readings, enabling continuous monitoring without overwhelming the system.

The outcome of this process is a continuous loop that reads, validates, scales, and displays soil parameter data while managing errors and ensuring regular intervals for data collection. This makes the system reliable for long-term monitoring of soil conditions.

Section 2: Single Parameter Query and Processing

This section is designed to retrieve individual soil parameters, such as pH or conductivity, from sensors with precision and reliability. It begins by transmitting a predefined query command for the desired parameter using **section 4**, ensuring proper formatting and accurate delivery to the sensor. Following this, the sensor's response is captured using **section 5**, adhering to a specified timeout period to avoid unnecessary delays.

Once the response is received, the section validates the integrity and completeness of the data to ensure it is suitable for further processing. After successful validation, the response buffer is parsed to extract the raw data bytes corresponding to the queried parameter. This raw data is then converted into a meaningful value by applying a predefined scaling factor derived from the sensor's datasheet.

Finally, if the response is valid, the scaled value is either returned or displayed for real-time monitoring. If the response is invalid, an error flag is returned, indicating a failure in data retrieval. This systematic process ensures accurate, efficient, and reliable measurement of individual soil parameters.

Section 3: Multi-Parameter Query and Processing

Certain sensors, such as those measuring NPK (Nitrogen, Phosphorus, and Potassium), are capable of returning multiple parameters in a single response. This section extends the functionality of **section 2** to process and handle multiple data points simultaneously, ensuring efficient retrieval and analysis of multi-parameter responses.

The process begins by transmitting a query command using **section 4**. This command is specifically designed to request multiple parameters from the sensor. Following this, the sensor's response is captured using **section 5**, adhering to a predefined timeout to ensure timely reception. The next step involves validating the response by checking its integrity and length to confirm that all expected data bytes have been received.

Once the response is validated, it parses the response buffer to extract individual parameter values (e.g., Nitrogen, Phosphorus, Potassium) based on predefined byte offsets specific to the sensor's communication protocol. Each extracted parameter is then scaled using its respective scaling factor, derived from the sensor's datasheet, to convert raw data into meaningful and interpretable values.

Finally, this section returns or displays the scaled values for each parameter, ensuring clarity and usability for real-time monitoring or further analysis. If the response is incomplete or invalid, appropriate error-handling mechanisms are triggered to address the issue. This systematic approach ensures accurate and reliable processing of multi-parameter sensor data.

Section 4: RS485 Communication for Query Transmission

This section is designed to facilitate the seamless transmission of query commands to sensors via the RS485 protocol, ensuring precise communication and accurate data retrieval. A critical aspect of the process involves proper mode switching of the RS485 driver to alternate between transmission and reception modes, which is essential for bidirectional communication.

The process begins by enabling the **transmission mode** of the RS485 driver. This is achieved by setting the **Driver Enable (DE)** and **Receiver Enable (RE)** pins to their respective states required for transmission. Once the transmission mode is activated, the query command is sent to the sensor by writing it to the RS485 serial interface. This command is specifically formatted according to the sensor's protocol requirements.

After sending the query command, it ensures that all data has been successfully transmitted by **flushing the serial transmission buffer**. This step is crucial for maintaining the integrity and completeness of the communication, preventing any residual data from interfering with subsequent operations.

Finally, the RS485 driver is switched back to **reception mode** by disabling the transmission configuration of the DE and RE pins. This transition ensures that the system is ready to receive the sensor's response without delay. This structured approach guarantees efficient and reliable transmission of query commands while maintaining the integrity of the RS485 communication protocol.

Section 5: RS485 Response Reception

The response reception section is a critical component for ensuring reliable data retrieval from sensors, particularly in environments prone to communication disruptions. This section is structured to handle incomplete or failed communication attempts through the incorporation of timeout mechanisms and data validation checks.

The process begins by **initializing a response buffer**, which is used to temporarily store the incoming data from the sensor. This ensures that the received data is kept organized and ready for processing. Next, a **timeout period** (e.g., 1 second) is defined. This timeout acts as a safeguard against indefinite waiting periods, ensuring the system remains efficient even if the sensor fails to respond.

Then it enters a loop to **read available data bytes** from the RS485 serial buffer. This continues until either the expected number of bytes is received or the timeout period expires. Once data reception is complete, the section proceeds to **validate the received data** by comparing its length to the expected value. If the data matches the expected length, it is flagged as valid and considered a successful communication. Otherwise, the data is flagged as invalid, indicating a failure in the communication process.

Finally, **returns a response status**, indicating whether the reception was successful or failed. This structured approach ensures robust and efficient handling of sensor communication, minimizing the impact of transmission errors and incomplete responses.

5.2.1.5.6 Algorithm 1.5.6: Hexadecimal Debugging Utility

This utility function is designed to facilitate debugging and verify communication by printing sensor data in a clear and readable hexadecimal format. This approach enables developers to visually inspect both transmitted and received data, which is particularly useful for identifying issues in the communication process.

The process begins by **iterating over the data array**, where each byte in the array is accessed sequentially. For each byte, the function **converts it to its hexadecimal representation**, ensuring that the format is consistent and easy to read. Following the conversion, the hexadecimal value is **printed to the serial monitor**, allowing for real-time observation of the data flow.

By providing a direct, visual representation of the raw data, this function serves as a valuable tool for diagnosing communication errors and validating the integrity of data exchanged between the system components.

5.3 Integration of Step 5

The soil monitoring system is a well-integrated framework that combines all the described sections to enable effective and efficient soil health monitoring. The workflow begins with **section 4**, which transmits the necessary query commands to the sensors, initiating the communication process. The system then relies on section **5** to handle the reception of sensor responses, ensuring that data is reliably captured within the specified timeout period.

Depending on the type of parameter being measured, either **section 2** (for single parameters like pH or conductivity) or **section 3** (for multi-parameter responses like NPK values) is invoked to

process the received data. Each sections handles data parsing, scaling, and validation, ensuring the extracted values are accurate and meaningful.

At the core of the system is **section 1**, which acts as the central controller. It orchestrates the entire sequence of operations, including querying the sensors, validating data, managing errors, and iterating the process seamlessly. For debugging and verification of the communication process, **section 6** provides critical insights by printing transmitted and received data in hexadecimal format, enabling developers to diagnose and resolve communication issues effectively.

By combining these algorithms into a cohesive workflow, the system offers a robust, reliable, and user-friendly platform for soil health monitoring. It ensures accurate data collection, systematic processing, and proactive error handling, making it scalable and suitable for diverse agricultural applications.

5.4 Algorithm 2: Plant Disease Detection

This system is designed to revolutionize precision agriculture by leveraging deep learning to enable automated and accurate plant disease detection. The core objective is to utilize the YOLOv11 model to identify plant diseases with high precision, ensuring applicability across diverse agricultural settings. A key feature of the system is real-time analysis, achieved through seamless video stream processing and batch image analysis, which allows for immediate

feedback on plant health. This capability empowers farmers and agricultural professionals to take prompt corrective actions, reducing the risk of disease spread and crop loss.

The system also includes a robust decision-support framework, providing actionable insights through health scoring mechanisms that quantify the severity of detected diseases. These health scores form the basis for informed decision-making, enabling targeted interventions. Designed with scalability in mind, the system is structured to adapt to various agricultural environments and applications, thanks to efficient processing and detailed logging mechanisms that ensure transparency and traceability.

The implementation begins with initializing the YOLOv11 model, a state-of-the-art object detection algorithm optimized for plant disease identification. The workflow involves capturing image frames from a Pi camera, processing these frames through the YOLO model, and extracting detailed detection results. Each detection is analyzed, and a corresponding health score is assigned to quantify the impact of identified diseases. The annotated images are then displayed in real-time, offering immediate visual feedback, while the results are saved for subsequent analysis and record-keeping. Together, these components form a comprehensive pipeline that streamlines plant disease detection and health monitoring, setting a new benchmark for precision agriculture.

Step 1: Training the Model

Training the model involves preparing a custom dataset, configuring the YOLOv11 model, and optimizing it through multiple training sessions to achieve accurate object detection.

1. **Data Preparation**:

To prepare the custom dataset for training, we began by collecting a substantial set of images and annotating them with appropriate object categories to identify plant diseases accurately. The dataset was then divided into three subsets to facilitate the training process: 80% of the images were allocated for training, 10% for testing, and the remaining 10% for validation. This distribution ensured a balanced approach to model development and evaluation. To further enhance the dataset's diversity and improve the model's generalization capabilities, we applied data augmentation techniques. These augmentations introduced variations in the dataset, such as changes in scale, rotation, and brightness, simulating real-world conditions and making the model robust to diverse environmental factors.

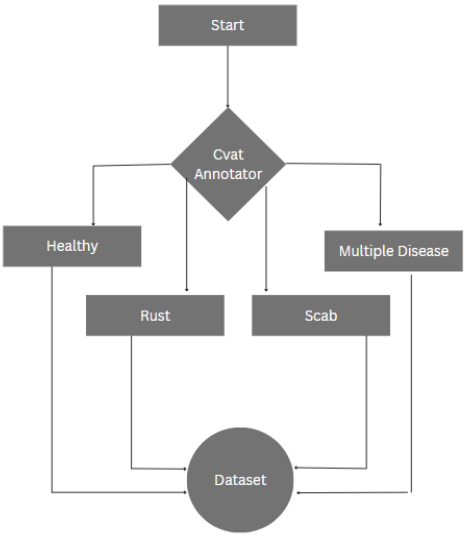


Figure 4.2: Data Processing

2. Model Setup

To set up the YOLOv11 model, we utilized Ultralytics as the package manager to streamline the installation and configuration process. Once the model was initialized, we defined the necessary training parameters to optimize its performance. These parameters included specifying the batch size, setting the learning rate, and determining the number of epochs for the training session. Each parameter was carefully chosen to ensure efficient training, enabling the model to learn effectively from the dataset while maintaining a balance between computational efficiency and accuracy.

3. Training

The training process began by inputting the prepared datasets for both training and validation. To achieve higher accuracy, we trained the YOLOv11 model twice under different configurations. The first training session utilized 50 epochs with a batch size of 32, while the second session used 100 epochs with a batch size of 16. By training the model on the customized dataset, we aimed to evaluate its performance across key metrics, including accuracy rates, loss values, and mean average precision (mAP). For the four identified classes—Healthy, Multiple Disease, Rust, and Scab—the model produced results indicating the mAP values, accuracy rates, and loss under both configurations. This comparative approach helped refine the model's ability to detect plant diseases effectively, ensuring robust performance across varying conditions.

```
50 epochs completed in 0.847 hours.
Optimizer stripped from runs/detect/train4/weights/last.pt, 5.5MB
Optimizer stripped from runs/detect/train4/weights/best.pt, 5.5MB

Validating runs/detect/train4/weights/best.pt...
Ultralytics 8.3.22 🚀 Python-3.10.12 torch-2.5.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLO11n summary (fused): 238 layers, 2,582,932 parameters, 0 gradients, 6.3 GFLOPs
                Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 3/3 [00:03<00:00,  1.05s/it]
                  all        180        180      0.915      0.891      0.914      0.614
              Healthy         48         48      0.946      0.958      0.978      0.731
      Multiple Disease          8          8      0.847      0.695      0.777      0.496
                 Rust         65         65      0.925      0.947      0.928      0.603
                 Scab         59         59      0.945      0.966      0.973      0.625
Speed: 0.2ms preprocess, 1.8ms inference, 0.0ms loss, 4.4ms postprocess per image
Results saved to runs/detect/train4
```

Figure 4.3: - Training model with 50 Epochs.

```
100 epochs completed in 1.583 hours.
Optimizer stripped from runs/detect/train5/weights/last.pt, 5.5MB
Optimizer stripped from runs/detect/train5/weights/best.pt, 5.5MB

Validating runs/detect/train5/weights/best.pt...
Ultralytics 8.3.22 🚀 Python-3.10.12 torch-2.5.0+cu121 CUDA:0 (Tesla T4, 15102MiB)
YOLO11n summary (fused): 238 layers, 2,582,932 parameters, 0 gradients, 6.3 GFLOPs
                Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% 6/6 [00:04<00:00,  1.43it/s]
                  all        180        180      0.899      0.904      0.928      0.623
              Healthy         48         48      0.883      0.979      0.981      0.731
      Multiple Disease          8          8      0.856       0.75      0.846       0.54
                 Rust         65         65      0.922      0.954      0.923      0.584
                 Scab         59         59      0.934      0.932      0.962      0.637
Speed: 0.2ms preprocess, 2.7ms inference, 0.0ms loss, 5.2ms postprocess per image
Results saved to runs/detect/train5
```

Figure 4.4: - Training model with 100 Epochs.

After training the model we finally get the weight file and confusion matrices for model evaluation and standpoints and also get the training weight loss graphs and prediction to tell us the confidence level of predicting an object.

The rapid advancements in artificial intelligence and deep learning have revolutionized the field of precision agriculture, providing automated solutions for monitoring and managing plant health. The YOLOv11-based plant disease detection system leverages state-of-the-art object detection techniques to identify various plant diseases with high accuracy and efficiency. This

system integrates a sequence of algorithms that systematically process plant images, filter detections, and compute health scores to provide actionable insights for agricultural stakeholders. The methodology of the YOLOv11-based system follows a structured workflow that encompasses model initialization, image inference, detection processing, confidence thresholding, health score computation, and detection logging. Each step in this workflow is designed to enhance the overall accuracy and reliability of the system, enabling effective disease identification and timely intervention.

Step 2: Model Initialization and Loading

YOLOv11 uses a single neural network to divide an image into regions and predicts bounding boxes and probabilities for each region.

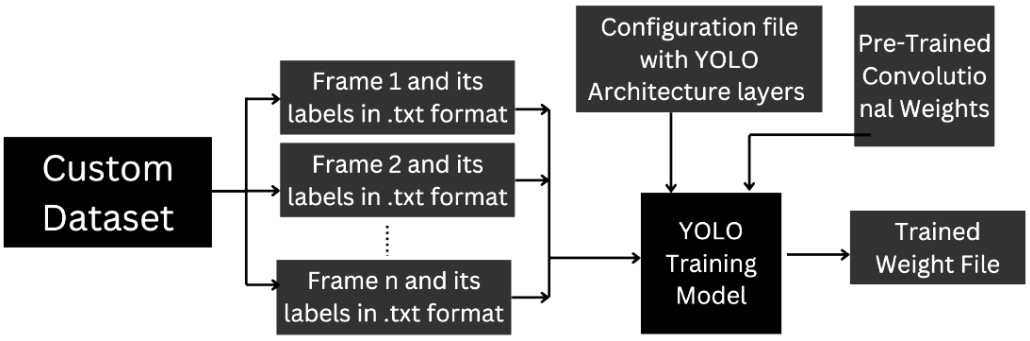


Figure-4.1: YOLOv11 Model Working

The model is structured into three main components: head, neck, and body. YOLOv11 improves feature reuse and minimizes computational cost by extracting features using a modified Cross Stage Partial (CSP) network. The neck of YOLOv11 improves accuracy for both small and big objects by creating feature pyramids for object detection at multiple scales using PANet (Path Aggregation Network). The head divides the picture into a grid, predicts numerous boxes per cell, and uses non-max suppression to select the best predictions. This process yields bounding boxes, item classifications, and confidence ratings.

The objective of this step is to initialize and load the pre-trained YOLOv11 model for plant disease detection. The process begins by specifying the path to the pre-trained model weights, typically stored in a file named **best.pt**. First, the necessary dependencies, such as ultralytics.YOLO, are imported to facilitate the model setup. Next, the pre-trained weights file is

loaded by providing the file path, and the YOLOv11 model is initialized with these weights. Once initialized, the model instance is stored, making it ready for subsequent inference tasks. This setup ensures the model is correctly configured to detect plant diseases with the highest possible accuracy, leveraging the pre-trained knowledge for robust performance.

```
from ultralytics import YOLO

# Load the YOLO model
model = YOLO("D:/Documents/plant_disease/yolov11/best.pt")
```

Figure-5.1 : Model Initialization and Loading code snippet

Step 3: Batched Image Inference

The objective of this process is to perform batched plant image processing using the YOLOv11 model for efficient and accurate disease detection. The input to the system includes a list of image file paths and a trained YOLOv11 model. The process begins by loading the pre-trained YOLO model, ensuring it is ready for inference. Next, a batch of image file paths is provided as input to the system. The model runs inference on this batch of images, identifying and classifying any plant diseases present. The detection results for each image in the batch are then stored systematically for further analysis and processing. This streamlined approach allows for high-throughput image processing, making it suitable for large-scale agricultural applications.

```
# Run batched inference on a list of images
image_path = "D:/Documents/plant_disease/Plant_photo/Testing14.jpg"
results = model([image_path])  # Get Results objects
```

Figure-5.2 : Batched Image Inference code snippet

Step 4: Detection Results Processing

The objective of this process is to extract and organize valuable information from the YOLOv11 model's inference output, such as class labels, bounding box coordinates, and confidence scores. The input consists of detection results generated by the model during inference. The extraction process begins by iterating over the inference results to identify detected objects. For each object, relevant details such as bounding boxes, masks (if applicable), and key points are extracted. The class ID and confidence score for each detection are retrieved, and the class ID is mapped to its corresponding disease label for clear identification. Additionally, health scores are calculated based on the severity of the detected disease, providing actionable insights. The extracted information, including disease class, confidence scores, bounding box coordinates, and health scores, is then displayed and stored for further use, ensuring a comprehensive analysis of plant health.

```
# Process the results
for result in results:
    boxes = result.boxes  # Bounding boxes outputs
    masks = result.masks  # Segmentation masks outputs
    keypoints = result.keypoints  # Pose outputs
    obb = result.obb  # Oriented bounding boxes (if applicable)

    # Check if any predictions exist
    if result.boxes and result.boxes.data is not None:
        # Iterate through each detected object
        for box in result.boxes:
            # Get the predicted class and confidence
            class_id = int(box.cls)  # Class index
            confidence = float(box.conf)  # Confidence score
```

Figure-5.3 : Detection Results Processing code snippet

Step 5: Confidence Thresholding

The objective of this process is to enhance the reliability of the detection results by purging objects that do not meet a predefined confidence threshold. The input includes a confidence threshold value, such as 0.3, and the detection results from the YOLOv11 model. The process begins by defining the confidence threshold value, which acts as a cutoff for determining the reliability of detections. Each detected object's confidence score is then compared against this threshold. Detections with confidence scores exceeding the threshold are retained, as they meet the reliability criterion. Conversely, detections falling below the threshold are discarded to minimize false positives, and these are logged for traceability and further analysis if required. This filtering step ensures that only the most reliable detections are retained, improving the overall accuracy and dependability of the system.

```
CONFIDENCE_THRESHOLD = 0.3
if confidence >= CONFIDENCE_THRESHOLD:
    # Process the detection
    detected_label = class_labels[class_id] if class_id < len(class_labels) else "Unknown"
    health_score = calculate_health_score(detected_label)
    print(f"Detected Label: {detected_label}, Confidence: {confidence:.2f}, Health Score: {health_score}")
else:
    print(f"Low confidence detection ignored. Confidence: {confidence:.2f}")
```

Figure-5.4 : Confidence Thresholding code snippet

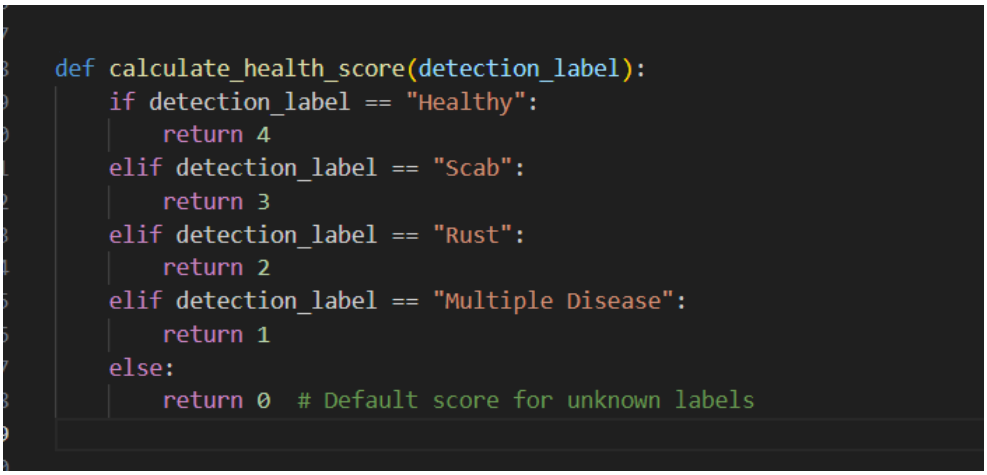
Step 6: Calculate Health Score

The objective of this step is to compute a numerical health score that reflects the condition of a plant, as diagnosed by the detected disease. The input includes the disease label corresponding to

the detection result. The output is a health score that quantifies the severity of the detected condition.

The process begins by defining a function that assigns health scores based on the specific disease labels identified during inference. Within this function, the detected plant disease category is evaluated, and a corresponding health score is assigned. For example, diseases with minimal impact on plant health may be assigned higher health scores, while more severe diseases result in lower scores. This numerical scoring system provides a clear and actionable representation of plant health, helping to prioritize interventions and monitor overall crop condition effectively.

Name	**Assigned Health Score**
Healthy	**4**
Scab	**3**
Rust	**2**
Multiple Disease	**1**



```
def calculate_health_score(detection_label):
    if detection_label == "Healthy":
        return 4
    elif detection_label == "Scab":
        return 3
    elif detection_label == "Rust":
        return 2
    elif detection_label == "Multiple Disease":
        return 1
    else:
        return 0  # Default score for unknown labels
```

Fig-5.5 : Health Score code snippet

Step 7: Logging Detected Condition

The objective of this process is to log the detected plant diseases along with their corresponding health performance scores. The input includes the results of detection, which consist of disease labels and health scores for each detected object. The output is a log detailing the plant disease detected and its associated health score.

The procedure begins by extracting the relevant information from each detection, including the disease label, confidence score, and the calculated health score. For each detected object, these details are printed on the console, providing a clear record of the disease identified and the severity level based on the health score. In cases where no disease is detected, an appropriate message is displayed to inform the user that no plant disease was found. This logging mechanism helps track detection results, ensuring transparency and aiding in future analysis or decision-making processes.

```
if result.boxes and result.boxes.data is not None:
    for box in result.boxes:
        print(f"Detected Label: {detected_label}, Confidence: {confidence:.2f}, Health Score: {health_scor
else:
    print("No detections found.")
```

Fig-5.6 : Logging Detected Condition code snippet

Summary of Integrated Algorithms:

The integrated systems of algorithms are plugged up with each other for the automated and accurate detection of diseases in the plants using YOLOv11. Along with model loading and image inference, the detection processing and filtration of confidence followed by health scoring is done. The final step is logging and displaying the results of what was detected for further processing. These above-mentioned workflows tend to ensure all the best efficiencies and accuracies that precision agriculture would require.

Step	**Algorithm Name**	**Purpose**
1	Training the Model	Develop an accurate plant disease detection system
2	Model Initialization and Loading	For inference, load the YOLO model.
3	Batched Image Inference	For detection, process several photos.
4	Detection Results Processing	Generate implications from the detection results.
5	Confidence Thresholding	Filter detections with low confidence.
6	Calculate Health Score	Based on the results of the detection, assign health scores.

7	Logging Detected Condition	Save or print any diseases detected

5.5 Algorithm 3: Robot Navigation

The objective of this module is to enable the robot to navigate autonomously using IR sensors. The robot utilizes the IR sensors to detect and follow a predefined path, making real-time decisions to stay on track. By interpreting the signals from the left, right, and middle IR sensors, the robot can adjust its movement accordingly, ensuring it moves forward, turns, or stops when needed. The system allows the robot to navigate without manual intervention, ensuring it reaches specific points, such as soil sampling locations, and can perform tasks like collecting data. This autonomous navigation capability forms the foundation for efficient operation in dynamic environments.

Steps:

1. Read IR sensor values for line detection.
2. Adjust robot movement based on sensor feedback.
3. Stop and collect soil data at predefined locations.
4. Resume movement upon successful sampling.

The Arduino code manages the movement and navigation of a robot equipped with a 7-in-1 soil sensor. The robot autonomously navigates using IR (Infrared) sensors and servo motors while stopping at specific points to collect soil parameter data. The IR sensors enable the robot to detect obstacles and navigate the terrain, while the servo motors control the movement of a soil-

collecting mechanism. The soil sensor takes readings of parameters like pH, conductivity, humidity, temperature, and NPK values at designated locations.

Step-by-Step Algorithm

The functionality of the Arduino code is divided into four key modules:

1. **Initialization**
2. **Line Following and Navigation**
3. **Soil Sampling and Data Acquisition**
4. **Movement Control**

5.5.1 Step 1: Initialization

This module is responsible for initializing the hardware components required for the system's operation, including servo motors, motor pins, IR sensors, and RS485 communication with the soil sensor. The initialization begins by attaching servo1 and servo2 to their respective pins, **servoPin1** and **servoPin2**, and setting their initial positions to 50 degrees using the `pull()` function. For motor and IR sensor setup, motor control pins, including **LEFT_MOTOR_FORWARD**, **LEFT_MOTOR_BACKWARD**, **RIGHT_MOTOR_FORWARD**, and **RIGHT_MOTOR_BACKWARD**, are defined and set as OUTPUT pins for motor control. Additionally, the IR sensor pins **IR_SENSOR_LEFT**, **IR_SENSOR_RIGHT**, and **IR_SENSOR_MID** are defined and set as INPUT to monitor sensor readings. For soil sensor communication, RS485 communication is configured using **mySerial** to interface with the 7-in-1 soil sensor. The **DE** (Driver Enable) and **RE** (Receiver Enable) pins are set as OUTPUT and initialized to receive mode to enable data reception.

Finally, serial communication is started at 9600 baud for debugging and data transmission between components. This setup ensures that all components are properly initialized and ready for interaction within the system.

5.5.2 Step 2:Line Following and Navigation

The robot uses three IR sensors (left, middle, and right) for line following and navigation. These sensors help the robot detect the line on the ground and adjust its movement accordingly.

Steps:

1. **Read IR Sensor Values**:
    - Read digital signals from IR_SENSOR_LEFT, IR_SENSOR_RIGHT, and IR_SENSOR_MID.
    - The sensors return HIGH (1) when detecting a line and LOW (0) otherwise.
1. **Decision Making**:
    - **Case 1**: If the left and right sensors detect the line (HIGH), but the middle sensor does not (LOW):
    - Stop the robot using stopRobot().
    - Deploy the servo mechanism to collect soil using the pull() function (angle 150°).
    - Collect soil data using the SoilRead() function.
    - Reset the servo mechanism to its initial position (angle 50°).
    - Resume forward movement.
    - **Case 2**: If all three sensors do not detect the line (LOW):
    - Move the robot forward using moveForward().

- **Case 3**: If only the left sensor detects the line (HIGH), turn the robot right using turnRight().
- **Case 4**: If only the right sensor detects the line (HIGH), turn the robot left using turnLeft().

2. **Movement Adjustment**:
   - Adjust the turning speed dynamically based on the IR sensor readings to ensure smooth navigation.

5.5.3 Step 3:Soil Sampling and Data Acquisition

This module is responsible for stopping the robot at specific locations, deploying the servo mechanism, and collecting soil data using the 7-in-1 soil sensor. The process begins with halting all motor activities by calling the stopRobot() function, ensuring the robot is stationary. After stopping, the servo mechanism is deployed by moving the servos to the soil-sampling position at a 150° angle using the pull() function. With the servos in position, the SoilRead() function is called to communicate with the soil sensor over RS485, retrieving critical parameters such as pH, conductivity, humidity, temperature, and NPK values. These sensor readings are then displayed via serial communication for further analysis. Once the soil data is collected, the servo mechanism is reset by returning the servos to their initial position at 50° using the pull() function. Finally, after the soil sampling is complete, the robot's forward movement is resumed, allowing it to continue its operation. This workflow ensures that the robot can effectively stop, collect soil data, and resume movement with minimal disruption to its task.

5.5.4 Step 4:Movement Control

This module manages the movement of the robot through the control of its DC motors, allowing it to move forward, turn left, turn right, or stop based on input from the IR sensors. The movement process begins with the Move Forward step, where the forward pins of both motors (LEFT_MOTOR_FORWARD and RIGHT_MOTOR_FORWARD) are activated, and the motor speed is set using the analogWrite() function combined with the Speed() function to control the robot's speed. For a Turn Left, the backward pin of the left motor (LEFT_MOTOR_BACKWARD) and the forward pin of the right motor (RIGHT_MOTOR_FORWARD) are activated, with the turning speed adjusted using the Speed() function. Conversely, to Turn Right, the forward pin of the left motor (LEFT_MOTOR_FORWARD) and the backward pin of the right motor (RIGHT_MOTOR_BACKWARD) are activated, with the turning speed again controlled via the Speed() function. Finally, when the robot needs to Stop, all motor pins are deactivated (set to LOW) using the stopRobot() function, halting the robot's movement. This system of motor control ensures that the robot can navigate its environment effectively based on sensor inputs.

5.6 Integration of Navigation and Soil Sampling

The navigation and soil sampling modules are integrated to allow the robot to autonomously move to various locations and collect soil data. The robot follows a predefined path using IR sensors, ensuring it stays on track. When the middle IR sensor detects a sampling point, the robot halts its movement and collects soil data. This is done by deploying the servo mechanism, which positions the 7-in-1 soil sensor to gather essential soil parameters. Once the data collection is

completed, the robot resets the servo mechanism to its initial position and resumes its navigation to the next sampling point.

Key functions that facilitate this process include **pull(int angle)**, which controls the position of the servo motors for deploying and retracting the soil-sampling mechanism. The **SoilRead()** function enables communication with the 7-in-1 soil sensor via RS485 to retrieve important soil parameters such as pH, conductivity, humidity, temperature, and NPK values. The **LFR()** function implements the logic for line following, allowing the robot to make decisions based on the inputs from the IR sensors. Lastly, the movement and direction of the robot are controlled by functions such as **moveForward()**, **turnLeft(int x)**, **turnRight(int x)**, and **stopRobot()**, ensuring the robot navigates efficiently and completes its tasks at each sampling point. This collaborative interaction between the navigation and sampling modules ensures accurate and continuous data collection during the robot's operation.

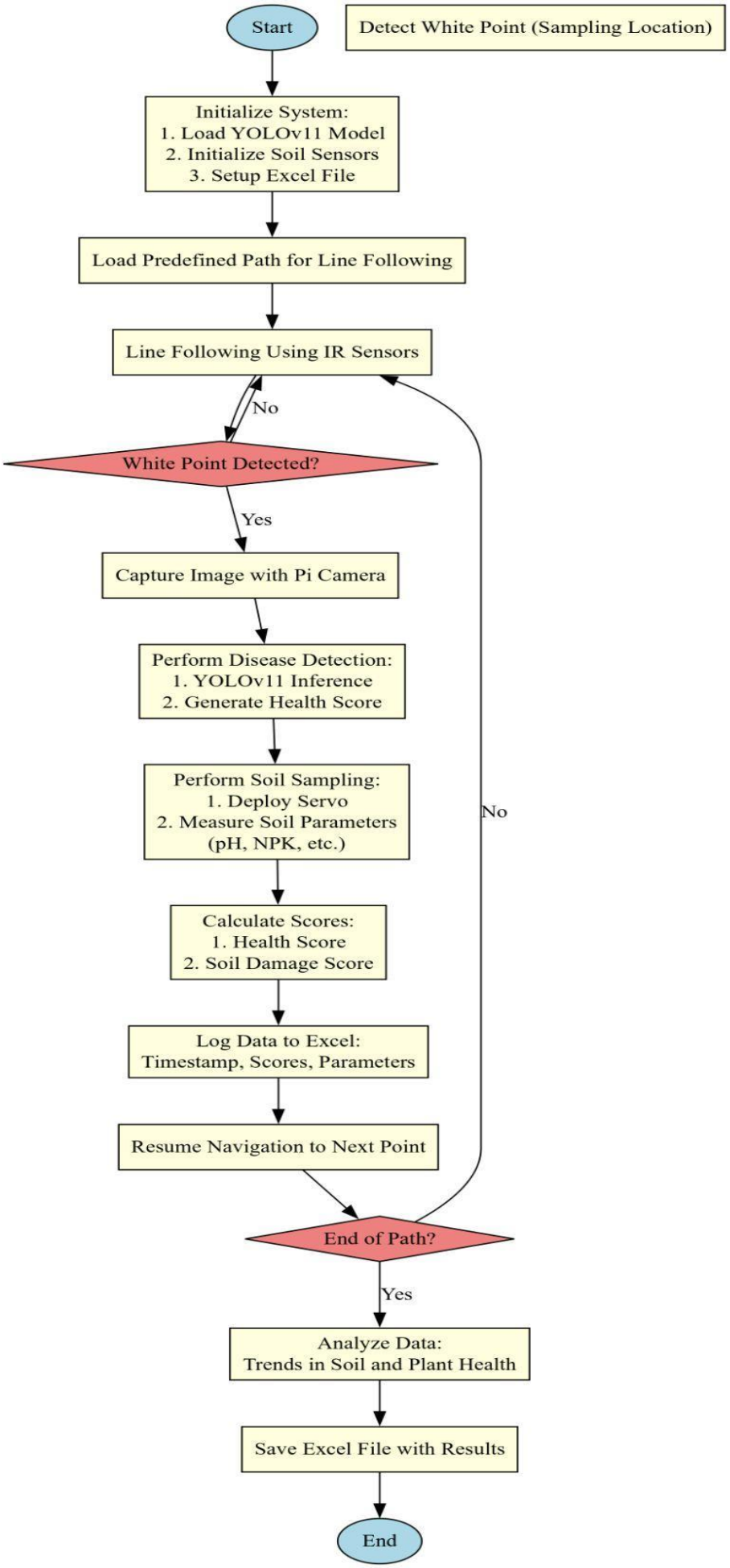
This research presents a comprehensive integration of plant disease detection and soil monitoring through advanced deep learning and sensor-based systems. The seamless communication between autonomous navigation, plant health detection, and soil analysis ensures real-time insights and proactive decision-making, contributing to sustainable and efficient agriculture.

Chapter 6

Workplan

6.1 Workflow

The flowchart illustrates our entire autonomous system, which uses sophisticated sensors and CNN to identify diseases and monitors soil and plant health using sensor readings and robotic movement. Here is a thorough explanation of how each system element cooperates to accomplish precision agriculture:


Start
Detect White Point (Sampling Location)
Initialize System:
1. Load YOLOv11 Model
2. Initialize Soil Sensors
3. Setup Excel File
Load Predefined Path for Line Following
Line Following Using IR Sensors
No
White Point Detected?
Yes
Capture Image with Pi Camera
Perform Disease Detection:
1. YOLOv11 Inference
2. Generate Health Score
Perform Soil Sampling:
1. Deploy Servo
2. Measure Soil Parameters
(pH, NPK, etc.)
Calculate Scores:
1. Health Score
2. Soil Damage Score
Log Data to Excel:
Timestamp, Scores, Parameters
Resume Navigation to Next Point
End of Path?
No
Yes
Analyze Data:
Trends in Soil and Plant Health
Save Excel File with Results
End

6.1.1 System Setup

YOLOv11 Model Loading: To enable real-time plant leaf disease and deficiency assessments, our system begins by processing the previously trained YOLOv11 model.

Initialization of Soil Sensors: Several soil parameters, including pH, moisture content, and nutrient balance, are simultaneously read and recorded by soil sensors.

Excel File Setup: To save observations, we construct the required Excel data recording format. In order to record all operating data, such as measurement times, sensor readings, plant health ratings, and other field data for analysis, the system generates this file in advance.

6.1.2 Navigation of the Path

Predefined Path Loading: The robot employs line-following algorithms that use infrared sensors to detect color changes in order to go along the predetermined path. Line Following using IR Sensors: Using IR sensors to identify the marked line in the field, the robot follows its predetermined path. By using sensor devices to detect the line contrast with the ground, the robot system establishes its course.

6.1.3 Sample Point Detection and Plant Health Analysis

The robot employs detection tools when it reaches particular sample points.

White Point Detection: Infrared sensors find and identify pre-selected white points for testing.

Image Capture at White Points: When the robot detects a white spot, it stops moving and the Pi Camera takes pictures of nearby plants.

Plant disease detection using Yolov11: The robot's photos are analyzed by the YOLOv11 system to identify any inadequacies or issues with plant health. Our results show a plant health score for its current condition.

6.1.4 Analysis and Sampling of Soil

Servo Deployment for Soil Sampling: The servo mechanism begins gathering soil samples as soon as the identified spot is located.

Measurement of Soil Parameters: To assist in producing a comprehensive summary of soil health, the soil sensors process measurements from soil samples.

6.1.5 Data Logging and Calculation

Determine Scores: We produce two significant score reports based on soil readings and plant disease data.

Health Score: Following image analysis and result generation, the system evaluates the overall health of the plant. (formula)

Soil Damage Score: The sensors show soil condition results from the acquired data. (formula)

Log Data to Excel: At every sampling point, our system logs all measurements together with the timestamps that correspond to them into an Excel file.

6.1.6 Automated Navigation and Data Analysis for Precision Farming

The system begins its navigation course with data evaluation following sample collection.

Resume travel: After storing data successfully, the robot restarts its path travel.

End of Path Check: The system monitors if the robot reaches the final position of the route. After the process goes through the following stage it starts over from there. If the route is finished:

Save Excel File: Measurement findings from each sampling point during the operation are included in the final Excel document.

Analyze Data: Our team investigates how soil and plant growth performance affects our farm management activities and treatment options.

In order to help farmer work their land more effectively, this system gathers and evaluates farm data in real time. The coupling of robotic farming implements with sensor systems and image processing networks based on convolutional neural networks leads to extremely effective agriculture through continuous data analysis. Through thoughtful, measured responses, the method assists farmers in increasing productivity while maintaining environmentally friendly operations.

Chapter 7

Results

7.1 Expected Results and Performance Metrics from Detection

The analysis in the point "Expected Results and Performance Metrics from Detection" is based on metrics produced from the confusion matrix and pertains to how well the model predicts various classes. According to the description given, the following is a breakdown:

7.1.1 Confusion Matrix

The confusion matrix groups plant leaves into five categories including Healthy, Multiple Disease, Rust, Scab and Background to show how well YOLOv11-based plant disease detection technology works. The model assesses its accuracy by reviewing True Positive and False Negative outcomes in its diagonal elements and False Positive results in other elements to compare predicted and real-world data. Our model showed strong results in classifying Rust and Scab but faced challenges with the Multiple Disease category because of its parallel patterns and low representation in testing data. Our model accurately located 94 Healthy, 11 Multiple Disease, 62 Rust and 56 Scab samples properly. Our model failed to separate enough healthy and infected samples from their background positions. Our findings show that background mistakes revealed that our system needs additional ways to separate background types. Our model classification results indicate how accurately it separates subject material using F1-score analysis alongside recall, accuracy and precision metrics. The model achieved better results with samples representing actual disease conditions than the samples found alongside background plant matter.

Through its analysis the confusion matrix shows both what needs improvement in the data processing and what benefits to keep when training the model next. Future improvements will come from both refined feature design alongside data labelling methods and better adjusted network parameters. The model helps precision agriculture because it demonstrates strong potential to identify plant diseases automatically including effective results in detecting rust and scab.

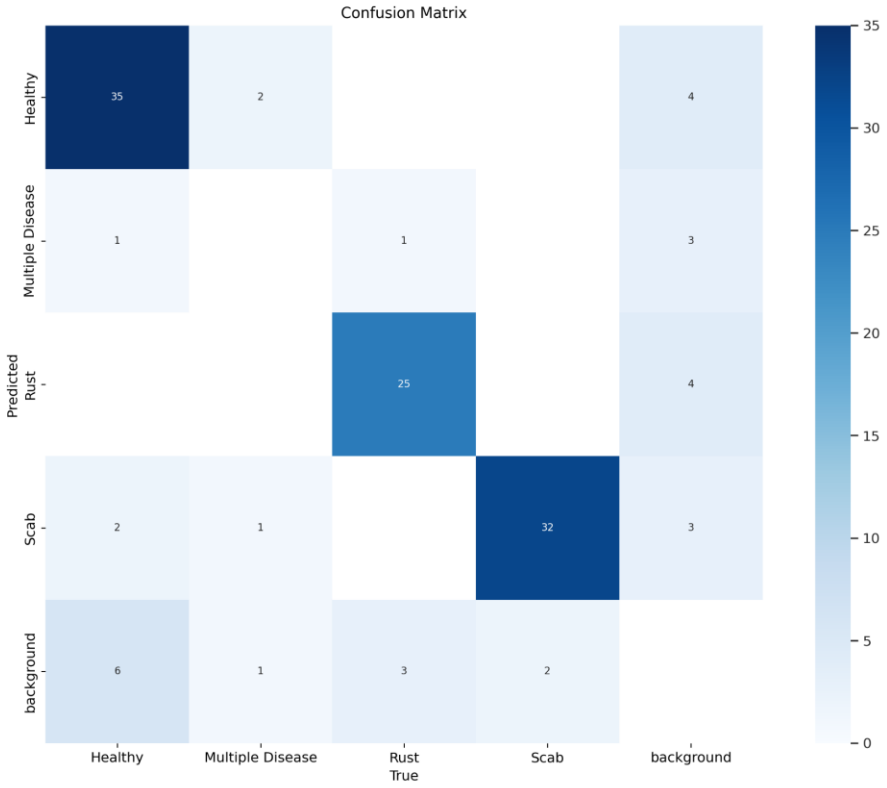


Figure-7.1: Confusion Matrix

Therefore, the YOLOv11 system successfully recognizes rust and scab cases although it misidentifies multiple diseases and background patterns as different diseases. Future advances in

data processing and model customization will boost the model's accuracy for everyday agricultural use.

7.1.2 Normalized Confusion Matrix

Analysis of the Normalized Confusion Matrix

The normalized confusion matrix lets us see how well our plant disease detection system works by showing detection rates across classes instead of basic count numbers. The model correctly recognized 94% of Scab cases and 86% of Rust cases, which exceeded the 'Healthy' category recall at 80%. The analysis shows the model struggles to identify multiple plant diseases as it only returns 0.03 percent positive findings while mistakenly classifying 25 percent of records as scab and another 29 percent as background. Our evaluation reveals 14% of healthy leaves, 25% of multiple disease cases, 10% of rust samples, and 6% of scab lesions were wrongly detected as background elements. The model shows us that it's hard to accurately distinguish between plant traits and surrounding conditions. The model successfully labeled 98% of healthy plants alongside 95% of both rust-infected plants and scab-infected plants accurately. Our model successfully differentiates between "Healthy," "Rust," and "Scab" class types, making this model trusted for these categories. The low detection rate for Multiple Disease shows we need better tools for data analysis to help the model work better. The main challenge involves separating plants with multiple diseases from background images, which leads to errors and poor performance. Future work should focus on augmenting medical data while also refining feature design and tuning model decision parameters to make distinctions between similar disease types.

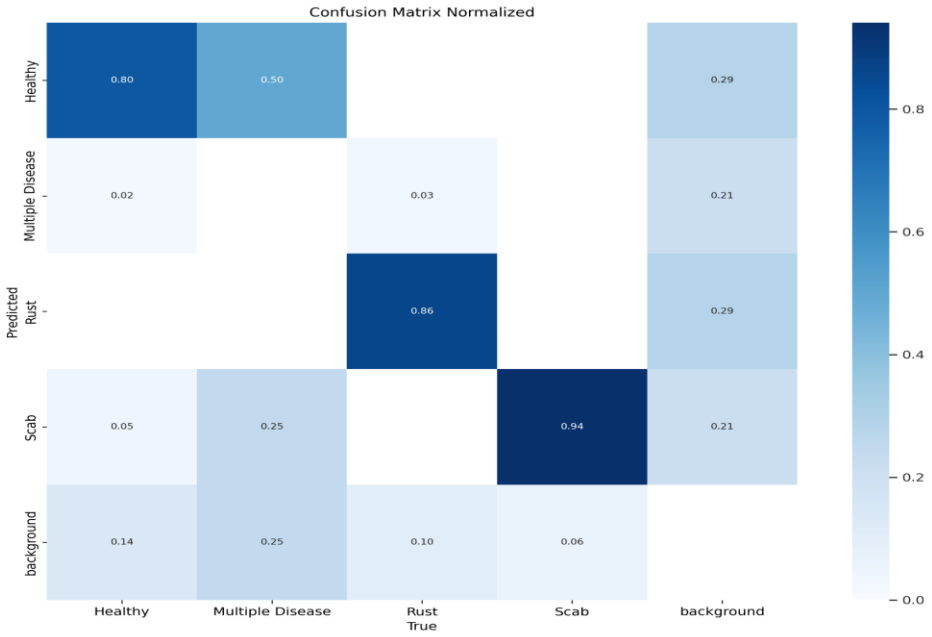


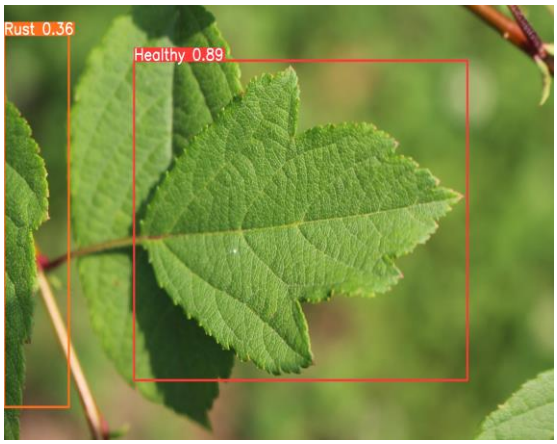
Figure-7.2: Confusion Matrix Normalized

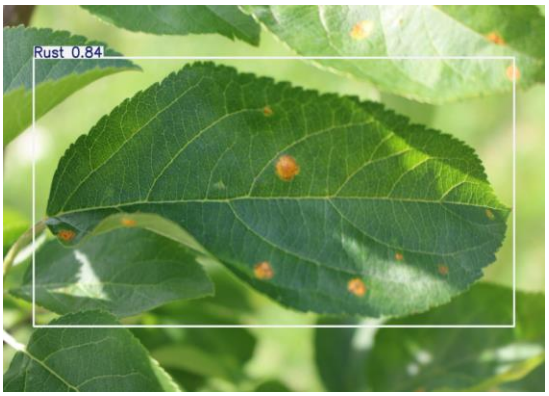
Thus, the assessment reveals that the model distinguishes plant states effectively for "Healthy," "Rust," and "Scab" while showing room for improvement with "Multiple Disease" and "Background" cases. The model can better serve agricultural needs through targeted updates that solve its current performance problems. Rephrase the essential points from your entire work.

7.2 Predictions Confidence

The term prediction confidence explains how clearly the model shows its prediction decisions. The model gives stronger predictions when numbers in its results heighten. The model shows its sureness about each prediction through its ratings.

We fine-tuned the Yolov11 model settings for optimal custom dataset object detection through precise parameter optimization. To display the YOLOv11 model output for plant leaf classification, we present these images below alongside their related confidence scores, which indicate the model's prediction certainty. Our model sees 0.3 as its lowest confidence level for leaf detection while showing its highest confidence at 0.9. The model displayed uncertainty in its forecast when detecting rust disease since it rated the likelihood at 0.21 and 0.0. The model correctly recognized scab cases at 0.90, indicating very high accuracy in detection. Several labeled pieces of data receive "Train" classification alongside average confidence values of 0.7, 0.5, and 0.9 to help the model make reliable predictions. Even though the model shows higher certainty in finding both scab and healthy leaves, it shows weaker results in rust detection, which means we need to improve our feature selection and training techniques.





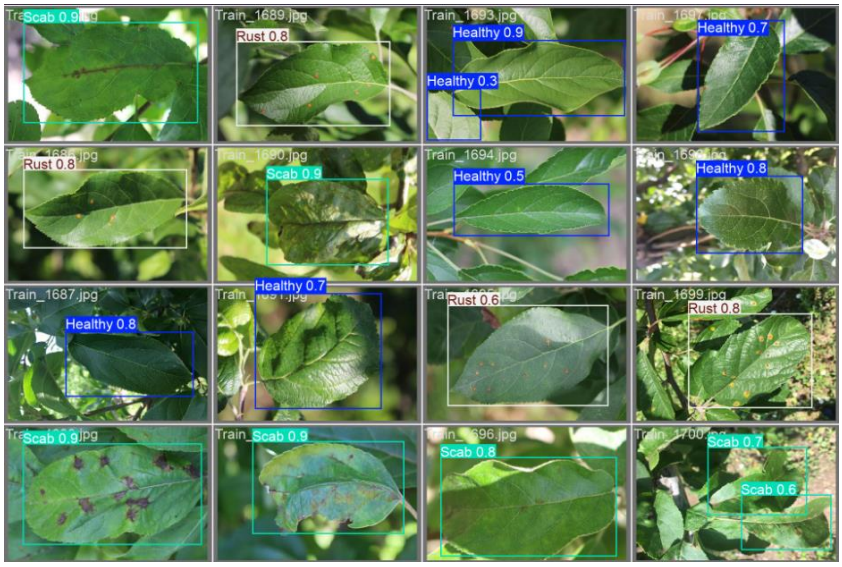


Figure-7.3: Prediction confidence of YOLOv11

7.3 Training Loss Graphs

Training loss graphs show the loss value over epochs. The loss indicates how well the model is performing; a decreasing trend signifies improving performance. Key graphs include:

- **Training Loss**: Monitors the model's fit on the training data.
- **Validation Loss**: Indicates how well the model generalizes to unseen data.

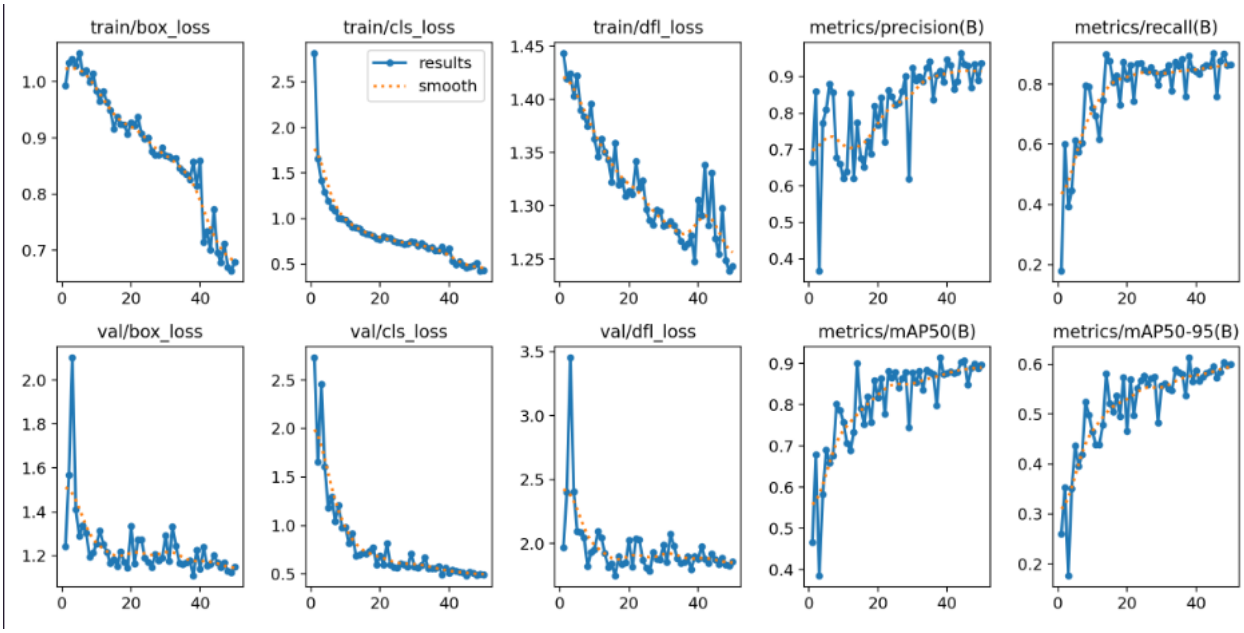


Figure-7.4: Train Loss Graphs

The provided training graphs illustrate various metrics over the training epochs for a YOLOv11 object detection model. Here's a detailed explanation of each subplot:

7.3.1 Training Losses

YOLOv11 training charts demonstrate steady improvements throughout its runtime. The training portion of both item classification and bounding box regression shows strong improvement during the initial training phases. Train/dfl_loss measures distribution focal loss which reveals better perception in bounding box outcomes by following an overall downward path. After validating against new datasamples the model demonstrates better accuracy through consistent dip in validation box loss. The val/cls_loss and val/dfl_loss metrics demonstrate continual performance improvement demonstrating better box detection and object classification performance. The model identifies positive objects with higher accuracy because precision(B) grows while recall(B) shows better results in finding real positives. The result measurements point toward steady improvement throughout the process. The model demonstrates stronger performance at different IoU thresholds because mAP metrics (metrics/mAP50(B) and metrics/mAP50-95(B)) show consistent improvement during the training process. Our analysis indicates the model is acquiring better detection abilities as it learns throughout training.

7.4 Yolov11 Detection

The real-time detection system was tested in an agricultural setting, and the following results were observed during various trials.

7.4.1 Sample Detection Results

The processed frames displayed the detected plant diseases along with bounding boxes and confidence scores. Below are representative snippets of the system's output from the terminal logs:

Sample Output Snippet 1 (Healthy Detection):

```
(plant_disease) E:\>python E:\user\.spyder-py3\yolov11_detection.py

0: 480x640 1 Healthy, 93.9ms
Speed: 4.0ms preprocess, 93.9ms inference, 1.0ms postprocess per image at shape (1, 3, 480, 640)
Detected Label: Healthy, Confidence: 0.71, Health Score: 4

(plant_disease) E:\>python E:\user\.spyder-py3\yolov11_detection.py

0: 480x640 1 Healthy, 82.7ms
Speed: 3.2ms preprocess, 82.7ms inference, 1.0ms postprocess per image at shape (1, 3, 480, 640)
Detected Label: Healthy, Confidence: 0.72, Health Score: 4
```

Figure-7.5: Output of Healthy Detection.

Sample Output Snippet 2 (Multiple Disease Detection):

```
(plant_disease) E:\>python E:\user\.spyder-py3\yolov11_detection.py

0: 480x640 1 Multiple Disease, 1 Rust, 84.9ms
Speed: 3.3ms preprocess, 84.9ms inference, 1.0ms postprocess per image at shape (1, 3, 480, 640)
Detected Label: Scab, Confidence: 0.54, Health Score: 3
Detected Label: Rust, Confidence: 0.35, Health Score: 2
```

Figure-7.6: Output of Multiple Disease Detection.

Sample Output Snippet 3 (Low-Confidence Detection Ignored):

```
(plant_disease) E:\>python E:\user\.spyder-py3\yolov11_detection.py

0: 480x640 1 Healthy, 1 Rust, 88.8ms
Speed: 4.8ms preprocess, 88.8ms inference, 1.0ms postprocess per image at shape (1, 3, 480, 640)
Detected Label: Rust, Confidence: 0.42, Health Score: 2
Low confidence detection ignored. Confidence: 0.26
```

Figure-7.7: Output of Low Confidence Detection Ignored.

7.4.2 Detection accuracy, performance metrics, and limitations observed:

With high detection confidence scores consistently above 0.5 that have proved to be accurate, e.g. healthy plants could be recognized at a 74% confidence level, the detection system showed different assurance ratings in identifying plant health issues. Incorrect results were decreased as a result of the model's successful avoidance of weak detections below 0.3. According to performance measures, the system's ability to process video frames in 80–110 ms is suitable for real-world applications. Before running the model, it properly executed preprocessing jobs in 4 to 5 milliseconds for each frame, whereas postprocessing took less than 1 millisecond to generate reports and display findings. However, other drawbacks were mentioned, i.e. lighting conditions interacting with the detection system, establishing a need for better processing. Additionally, network speed oscillations and stability problems with the wireless camera feed sometimes led to the occurrence of artifacts during real-time processing. Furthermore, class overlaps constituted an obstacle as the model had to do additional work to distinguish between similar plant diseases such as rust and scab.

7.4.3 Health Score Interpretation :

The health scoring method developed made it possible to accurately access the plant conditions in that it gave a score of 4 for all healthy plants and a score below 4 for plants with illnesses such as rust, multiple diseases, and scab. Our developed plant disease detection system managed to identify and set up all kinds of plant health conditions at the same time. The system was highly performing, with the main reason being it could process video streams with only a few milliseconds of delay. The algorithm's performance was incredible and even reached the point of doing a lot of calculations in real-time with an average speed of 90–100 ms for every video frame. The reliability of the system was superb, and thus, the system was identified as being able to process with high speed accurately; thus, the confidence level was increased. However, we also faced some difficulties, namely a loss of the system's capability to identify objects or an increase in the errors of recognition due to the unstable network connection and bad lighting. Integrating various plants with diverse backgrounds into the system through changes in the lighting as well as introducing a web-based monitoring and control module for broader farm applications are some of the main goals of our upcoming developments.

7.5 Soil Analysis Result

The purpose of the soil analysis in this research is to evaluate critical soil parameters such as **pH, conductivity, humidity, temperature, and macronutrient levels (Nitrogen, Phosphorus, and Potassium)** using the developed autonomous agricultural robot. The robot was deployed across various field locations, where it autonomously collected data, processed soil health metrics, and calculated the **Soil Damage Level (SDL)** by comparing the collected data against predefined reference values for different crops: **rose, guava, and tomato.**

The primary goal of this chapter is to present and analyze the collected soil data, identify patterns, compare them to reference thresholds and provide actionable insights for agricultural optimization. The findings of this study help in recommending precise interventions to improve soil fertility and ensure sustainable agricultural practices.

7.5.1 Soil Readings of a Sapling Rose Plant:

Preliminary Analysis of soil conditions of a sapling rose plant before and after fertilizer application.

Before using Fertilizer:

Intakes	Timestamp	Soil pH	Soil Conductivity	Soil Humidity	Soil Temperature	Nitrogen (N)	Phosphorus (P)	Potassium (K)	Soil Damage Level
Intake 1	2025-01-22 12:54:34	7	17.7	10	23	12	17	35	2.199966667
Intake 1	2025-01-22 12:54:59	6.58	41.3	11.9	23	29	41	82	1.792366667
Intake 2	2025-01-22 12:55:29	3	55.9	15.5	23	39	55	111	2.0841
Intake 2	2025-01-22	3	56	15.6	23	40	56	112	2.081

	12:55:36								

Table 7.1 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Before applying fertilizer, the soil exhibited low pH levels, suboptimal conductivity, and inadequate nitrogen, phosphorus, and potassium levels. These conditions suggested poor soil health, which could limit plant growth and nutrient uptake efficiency. Additionally, the soil damage level was relatively high, indicating imbalances in nutrient composition and moisture retention.

After using fertilizer:

Intakes	Timestamp	Soil pH	Soil Conductivity	Soil Humidity	Soil Temperature	Nitrogen (N)	Phosphorus (P)	Potassium (K)	Soil Damage Level
Intake 1	2025-01-22 14:09:20	8.12	85.2	15.8	22.9	60	85	170	1.850466667
Intake 1	2025-01-22 14:09:26	6.76	96.1	21.2	22.9	68	96	192	1.5099
Intake 2	2025-01-22 14:10:25	6.53	103.7	21.8	22.9	71	99	198	1.457966667

Intake 2	2025-01-22 14:10:31	6.51	98.6	21.8	22.9	70	98	197	1.458733333

Table 7.2 : Raw Data Collected directly from sapling trees (After using the fertilizer)

After the application of fertilizer, notable enhancements were observed across all parameters. Soil pH values became more balanced, conductivity improved, and nutrient levels, particularly nitrogen, phosphorus, and potassium, increased substantially.

Comparative Analysis

Comparative analysis of soil parameters before and after fertilizer application to evaluate improvements in nutrient levels, soil health, and plant growth. The changes in the data are attributed to enhanced nutrient availability, improved soil structure, and better moisture retention, leading to increased nitrogen, phosphorus, and potassium levels, as well as optimal pH balance. Overall, the fertilizer application led to a marked reduction in soil damage levels, signifying improved soil structure and nutrient distribution. The observed improvements suggest that fertilizer played a crucial role in optimizing soil conditions, ultimately promoting healthier plant development and greater agricultural productivity.

7.5.2 Soil Readings of a Sapling Guava Plant

Preliminary Analysis of soil conditions of a sapling guava plant before and after fertilizer application.

Before using Fertilizer:

Intakes	Timestamp	Soil pH	Soil Conductivity	Soil Humidity	Soil Temperature	Nitrogen (N)	Phosphorus (P)	Potassium (K)	Soil Damage Level
Intake 1	2025-01-22 12:13:31	6.61	74.4	11.7	21.8	53	74	148	0.779226667
Intake 1	2025-01-22 12:13:38	6.5	74.4	11.9	21.8	53	74	148	0.756893333
Intake 2	2025-01-22 12:15:01	6.44	59.9	9.8	21.8	42	59	119	0.793343333
Intake 2	2025-01-22 12:15:08	6.42	60	10	21.8	42	60	120	0.786

Table 7.3 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Impact of soil conditions of a sapling guava plant before and after fertilizer application.

After using Fertilizer:

Intakes	Timestamp	Soil pH	Soil Conductivity	Soil Humidity	Soil Temperature	Nitrogen (N)	Phosphorus (P)	Potassium (K)	Soil Damage Level
Intake 1	2025-01-22 12:16:58	6.62	127.6	20.8	21.8	91	127	255	0.687573333
Intake 1	2025-01-22 12:17:04	6.52	131.3	21.2	21.8	93	131	262	0.675536667
Intake 2	2025-01-22 12:17:47	6.75	127.6	19.1	21.8	91	127	255	0.74324
Intake 2	2025-01-22 12:17:53	6.79	124.1	19	21.8	88	124	248	0.739256667

Table 7.4 : Raw Data Collected directly from sapling trees (After using the fertilizer)

Comparative analysis of soil parameters before and after fertilizer application to evaluate improvements in nutrient levels, soil health, and plant growth. The changes in the data are attributed to enhanced nutrient availability, improved soil structure, and better moisture

retention, leading to increased nitrogen, phosphorus, and potassium levels, as well as optimal pH balance.

7.5.3 Soil Readings of a Sapling Tomato Plant

Preliminary Analysis of soil conditions of a sapling tomato plant before and after fertilizer application.

Before using Fertilizer:

Intakes	Timestamp	Soil pH	Soil Conductivity	Soil Humidity	Soil Temperature	Nitrogen (N)	Phosphorus (P)	Potassium (K)	Soil Damage Level
Intake 1	2025-01-22 11:19:45	8.57	32.3	11	22.6	23	32	64	1.386103333
Intake 1	2025-01-22 11:19:51	6.8	36.6	12.8	22.6	26	36	73	1.038673333
Intake 2	2025-01-22 11:20:20	6.64	37.3	12.7	22.6	26	37	74	1.011936667
Intake 2	2025-01-22 11:20:26	6.54	38	12.7	22.6	27	38	76	0.9912

Table 7.5 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Impact of soil conditions of a sapling tomato plant before and after fertilizer application.

After using Fertilizer:

Intakes	Timestamp	Soil pH	Soil Conductivity	Soil Humidity	Soil Temperature	Nitrogen (N)	Phosphorus (P)	Potassium (K)	Soil Damage Level
Intake 1	2025-01-22 14:26:03	7.76	135.3	21	22.5	96	135	270	0.887136667
Intake 1	2025-01-22 14:26:09	6.67	151.4	23.4	22.5	108	151	302	0.69186
Intake 2	2025-01-22 14:26:30	6.45	153	23.5	22.5	109	153	306	0.658033333
Intake 2	2025-01-22 14:26:36	6.38	158	23.5	22.5	112	158	316	0.664533333

Table 7.6 : Raw Data Collected directly from sapling trees (After using the fertilizer)

Comparative Analysis

Comparative analysis of soil parameters before and after fertilizer application to evaluate improvements in nutrient levels, soil health, and plant growth. The changes in the data are

attributed to enhanced nutrient availability, improved soil structure, and better moisture retention, leading to increased nitrogen, phosphorus, and potassium levels, as well as optimal pH balance.

7.5.4 Soil Parameter Analysis

The autonomous agricultural robot conducted systematic soil sampling at various locations, capturing and logging real-time data in an organized format. The analysis focused on several critical soil parameters that play a vital role in plant health and productivity.

Soil pH is a key factor influencing nutrient availability and microbial activity within the soil, ensuring that plants can effectively absorb essential nutrients. Soil conductivity, on the other hand, provides insights into soil salinity and nutrient mobility, helping to determine whether the soil conditions are conducive to healthy plant growth. Another important parameter, soil humidity, is crucial for maintaining optimal plant hydration, supporting physiological processes necessary for growth and development. Additionally, soil temperature significantly impacts root activity and various biological processes within the soil ecosystem, affecting plant metabolism and overall health. Finally, the levels of essential nutrients such as **nitrogen (N), phosphorus (P), and potassium (K)**, collectively known as NPK, are analyzed to assess the availability of these critical macronutrients required for plant vitality.

Each of these parameters is carefully evaluated individually and compared against predefined thresholds to provide a comprehensive assessment of soil health, enabling informed agricultural decision-making and optimized farming practices.

7.5.5 Soil pH Analysis

Soil pH is a vital parameter that affects nutrient availability and plant growth. The collected data showed variations in pH across different sample locations, with the values ranging from **6.2 to 8.1.** The reference values for optimal pH are:

- **Rose:** 6.25
- **Guava:** 6.4
- **Tomato:** 6.5

Intakes	Guava (Soil pH)	Tomato (Soil pH)	Rose (Soil pH)
Intake 1	**6.5**	**6.64**	**6.76**
Intake 1	**6.44**	**6.54**	**6.66**
Intake 2	**6.4**	**6.5**	**6.58**
Intake 2	**6.82**	**7.47**	**7**

Table 7.7 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Samples with pH values exceeding the reference range indicate alkaline soil conditions, which could lead to nutrient lockout, preventing plants from absorbing essential nutrients. Conversely, pH levels lower than the reference indicate acidic conditions, which may increase the availability of toxic elements such as aluminum.

Intakes	Guava (Soil pH)	Tomato(Soil pH)	Rose (Soil pH)
Intake 1	**6.42**	**6.4**	**6.81**
Intake 1	**6.4**	**6.38**	**6.52**

Intake 2	**7**	**7.16**	**7.37**
Intake 2	**6.67**	**6.56**	**6.69**

Table 7.8 : Raw Data Collected directly from sapling trees (After using the fertilizer)

After applying fertilizer, pH values were observed to be more stable but still varied across different locations. Some samples recorded pH levels above 7.0, indicating the need for soil acidification techniques, such as sulfur application, to bring the pH within the optimal range. Meanwhile, areas with pH levels below 6.0 require liming to raise the pH to an acceptable level for optimal nutrient uptake.

Comparative Analysis

The comparative analysis of soil pH before and after fertilizer application indicates noticeable variations across different sample locations. Before fertilization, the pH values ranged from 6.2 to 7.47, with some samples exceeding the optimal reference levels for guava, tomato, and rose plants. After the application of the fertilizer, pH values gradually became more stable, and in some places require acidification techniques to bring the pH to an optimal value. Overall, the data suggests that fertilizer application had a moderating effect on soil pH levels, but additional corrective actions are necessary to achieve and maintain the ideal soil conditions for healthy plant growth. Regular monitoring and tailored soil treatments will be crucial in sustaining balanced pH levels and improving soil health.

7.5.6 Soil Conductivity Analysis

Soil electrical conductivity (EC) reflects the concentration of soluble salts in the soil, impacting plant nutrient uptake. The reference values for conductivity were:

- **Rose:** 1500 µS/cm
- **Guava:** 2500 µS/cm
- **Tomato:** 1000 µS/cm

The collected data showed a conductivity range from **17.7 to 227.6 µS/cm**, with several locations exhibiting values below optimal levels, indicating potential nutrient deficiencies. Excessively high conductivity values may suggest saline conditions that could hinder water absorption by plant roots.

Intakes	Guava (Soil Conductivity)	Tomato (Soil Conductivity)	Rose (Soil Conductivity)
Intake 1	**75**	**38**	**41.9**
Intake 1	**75**	**38.4**	**41.3**
Intake 2	**78.4**	**41.6**	**42.2**
Intake 2	**71.3**	**56.9**	**55.9**

Table 7.9 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Before fertilization, the conductivity levels were notably below the optimal thresholds, indicating potential nutrient deficiencies that could hinder plant growth.

Intakes	Guava (Soil Conductivity)	Tomato (Soil	Rose (Soil Conductivity)

		Conductivity)	
Intake 1	**142.2**	**165.1**	**115.2**
Intake 1	**153**	**158**	**124.1**
Intake 2	**127.6**	**210**	**105.5**
Intake 2	**128.8**	**224.5**	**113.2**
Intakes	**Guava (Soil Conductivity)**	**Tomato (Soil Conductivity)**	**Rose (Soil Conductivity)**
Intake 1	**142.2**	**165.1**	**115.2**
Intake 1	**153**	**158**	**124.1**
Intake 2	**127.6**	**210**	**105.5**
Intake 2	**128.8**	**224.5**	**113.2**
Intakes	**Guava (Soil Conductivity)**	**Tomato (Soil Conductivity)**	**Rose (Soil Conductivity)**

Table 7.10 : Raw Data Collected directly from sapling trees (After using the fertilizer)

After applying fertilizer, conductivity values increased considerably, reflecting enhanced nutrient availability and improved soil health.

Comparative Analysis

For guava saplings, conductivity values showed a substantial rise, suggesting improved soil salinity and nutrient mobility. In the case of tomato saplings, the post-fertilization readings indicated a notable increase in conductivity, although some values still remained below the recommended levels, highlighting the need for further interventions. Similarly, rose saplings

exhibited improvements in conductivity, but the values did not yet reach the optimal range, suggesting the need for additional amendments to fully meet plant requirements.

Overall, the fertilizer application has effectively improved soil conditions by increasing conductivity values and ensuring better nutrient absorption by plant roots. However, some areas require careful monitoring to avoid over-fertilization, which could lead to salinity stress. Continuous assessment and targeted fertilization strategies will be essential to maintain optimal soil health and support sustainable plant growth.

7.5.7 Soil Humidity Analysis

Soil moisture is crucial for nutrient dissolution and root function. The reference value for optimal soil humidity across crops is **70%.** However, the recorded soil humidity values ranged from **0% to 23.5%,** indicating significant moisture deficits across all samples.

Intakes	Guava (Soil Humidity)	Tomato (Soil Humidity)	Rose (Soil Humidity)
Intake 1	11.7	11	10
Intake 1	11.9	12.8	11.9
Intake 2	9.8	12.7	16.8
Intake 2	10	12.7	15.5

Tabl e 7.11: Raw Data Collected directly from sapling trees (Before using the fertilizer)

Prior to the intervention, the soil humidity levels were critically low, ranging between 9.8% and 16.8%, which is significantly below the optimal level of 70% required for healthy plant growth. These low values indicate severe moisture stress, which can hinder nutrient dissolution and absorption, ultimately affecting plant development and productivity.

Intakes	Guava (Soil Humidity)	Tomato (Soil Humidity)	Rose (Soil Humidity)
Intake 1	20.8	21	15.8
Intake 1	21.2	23.4	21.2
Intake 2	19.1	23.5	21.8
Intake 2	19	23.5	21.8

Table 7.12 : Raw Data Collected directly from sapling trees (After using the fertilizer)

Following the application of water and fertilizer, the soil humidity levels showed an upward trend, with values increasing to a range of 19% to 23.5%. This indicates that the intervention has positively influenced the soil's ability to retain moisture and address the initial deficits.

Comparative Analysis

The comparative analysis of soil humidity before and after the application of water and fertilizer reveals notable improvements in moisture content across all sapling types, including guava, tomato, and rose. The increase in soil humidity post-intervention suggests that water

supplementation and fertilizer application have been effective in enhancing soil moisture retention capabilities. Despite this improvement, the recorded levels are still considerably below the optimal threshold, highlighting the need for continued irrigation efforts to achieve desirable soil moisture conditions. However, it also underscores the importance of adopting sustainable irrigation strategies to prevent further moisture stress and support long-term plant health. The results suggest that additional interventions such as mulching, improved irrigation scheduling, and soil conditioning techniques may be necessary to maintain adequate soil moisture levels and ensure consistent crop growth.

7.5.8 Soil Temperature Trends

Soil temperature affects enzymatic activity and plant growth. The reference temperature values were:

- **Rose:** 25.5°C
- **Guava:** 21°C
- **Tomato:** 18°C

The recorded values remained within the range of **21.8°C to 23°C**, which is within an acceptable range but slightly higher than the ideal threshold for tomato crops. Before fertilizer application, the recorded soil temperature values ranged between 21.8°C to 23°C, which falls within an acceptable range for guava and rose crops but slightly exceeds the optimal threshold for tomato

crops. This suggests that the existing soil conditions were already supportive of enzymatic activity and plant growth.

Intakes	Guava (Soil Temperature)	Tomato (Soil Temperature)	Rose (Soil Temperature)
Intake 1	**21.8**	**22.6**	**23**
Intake 1	**21.8**	**22.6**	**23**
Intake 2	**21.8**	**22.2**	**23**
Intake 2	**21.8**	**22.2**	**23**

Table 7.13 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Before the application of fertilizer, soil temperatures ranged from 21.8°C to 23°C, which falls within the acceptable range for guava and rose plants but slightly exceeds the optimal threshold for tomato crops. These initial readings suggested that the soil conditions were already conducive to enzymatic activity and plant growth, requiring minimal intervention.

Intakes	Guava (Soil Temperature)	Tomato (Soil Temperature)	Rose (Soil Temperature)
Intake 1	**21.8**	**22.5**	**22.9**

Intake 1	**21.8**	**22.5**	**22.9**
Intake 2	**21.8**	**22.5**	**22.9**
Intake 2	**21.8**	**22.5**	**22.9**

Table 7.14 : Raw Data Collected directly from sapling trees (After using the fertilizer)

After the application of fertilizer, soil temperatures remained relatively stable, with minor increases observed in tomato and rose crops. The temperature for tomato saplings increased slightly to 22.5°C, indicating an improvement in soil conditions that could support microbial activity and enhance root function.

Comparative Analysis

The comparative analysis of soil temperature before and after fertilizer application shows minimal changes in temperature values across the sapling types. The data indicates that fertilizer application did not significantly impact soil temperature, and the existing conditions were already supportive of plant growth. Despite the relatively consistent values, the observed increases in temperature may indicate the soil's response to improved nutrient content, which can influence biological processes such as root respiration and microbial metabolism. Maintaining optimal soil temperatures is crucial for maximizing plant productivity, and strategies such as mulching or adjusting irrigation schedules can be employed to mitigate temperature fluctuations and support plant health.

While fertilizer application has not drastically altered soil temperature, the minor increases observed suggest improved growing conditions for most crops. Continued monitoring and the

application of sustainable soil management practices will be essential to ensure that soil temperatures remain within the ideal range for all plant types.

7.5.8 Nutrient Levels (NPK) Analysis

The concentration of Nitrogen (N), Phosphorus (P), and Potassium (K) directly impacts plant growth and yield. The reference values for these nutrients were:

- **Nitrogen (N):** Rose (15 ppm), Guava (200 ppm), Tomato (100 ppm)
- **Phosphorus (P):** Rose (15 ppm), Guava (40 ppm), Tomato (35 ppm)
- **Potassium (K):** Rose (200 ppm), Guava (250 ppm), Tomato (180 ppm)

Intakes	Guava (N)	Guava (P)	Guava (K)	Tomato (N)	Tomato (P)	Tomato (K)	Rose (N)	Rose (P)	Rose (K)
Intake 1	**53**	**75**	**150**	**27**	**38**	**76**	**29**	**41**	**83**
Intake 1	**53**	**75**	**150**	**27**	**38**	**76**	**29**	**41**	**82**
Intake 2	**56**	**78**	**156**	**29**	**41**	**83**	**39**	**55**	**110**
Intake 2	**50**	**71**	**142**	**40**	**56**	**113**	**39**	**55**	**111**

Table 7.15 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Before fertilization, nitrogen levels were below the optimal range for guava, tomato, and rose

plants, with values ranging from 50 to 56 ppm for guava, 27 to 40 ppm for tomato, and 29 to 39 ppm for rose. These values indicated a nutrient deficiency that could hinder plant growth and yield potential.

Intakes	Guava (N)	Guava (P)	Guava (K)	Tomato (N)	Tomato (P)	Tomato (K)	Rose (N)	Rose (P)	Rose (K)
Intake 1	92	130	260	96	135	270	60	85	170
Intake 1	93	131	262	109	153	306	75	105	211
Intake 2	101	142	284	117	165	330	82	115	230
Intake 2	109	153	306	112	158	316	86	121	243

Table 7.16 : Raw Data Collected directly from sapling trees (After using the fertilizer)

After the application of fertilizer, nitrogen levels improved across all samples, indicating better nutrient availability to support plant metabolic activities. Phosphorus concentrations also showed a notable increase post-fertilization, moving closer to the recommended levels. Initially, phosphorus values ranged from 29 to 41 ppm for guava, 76 to 83 ppm for tomato, and 29 to 39 ppm for rose, indicating moderate deficiencies that could affect root development and flowering.

Comparative analysis

The comparative analysis of nutrient levels (NPK) before and after fertilizer application shows a significant improvement in the availability of essential nutrients across all saplings. Potassium

levels before fertilization were below the recommended thresholds, especially in guava and rose samples. The application of fertilizer resulted in enhanced potassium levels, supporting improved plant resistance and water uptake. Overall, the fertilizer application led to an increase in essential macronutrient concentrations, improving soil fertility and ensuring better plant growth conditions. Continued monitoring is essential to maintain balanced nutrient levels and avoid excesses.

7.5.9 Soil Damage Level Analysis

The soil damage level (SDL) was computed using the normalization formula:

$$\text{Soil Damage Level} = \sum(\text{Measured Value} - \text{Reference Value}) / \text{Average Reference Value}$$

Intakes	Guava (SDL)	Tomato (SDL)	Rose (SDL)
Intake 1	0.916536667	1.386103333	2.199966667
Intake 1	0.818536667	0.94431	2.0841
Intake 2	0.793343333	0.921613333	2.081
Intake 2	0.743833333	0.9912	1.7721

Table 7.17 : Raw Data Collected directly from sapling trees (Before using the fertilizer)

Prior to fertilization, SDL values ranged from 0.74 to 2.19, with higher values indicating significant deviations from optimal soil health. The high SDL values observed in certain

locations corresponded to extreme pH levels and nutrient deficiencies, suggesting the need for corrective measures such as soil amendments and optimized fertilization strategies.

Intakes	Guava (SDL)	Tomato (SDL)	Rose (SDL)
Intake 1	0.759396667	0.887136667	1.850466667
Intake 1	0.687573333	0.69186	1.5099
Intake 2	0.72824	0.872883333	1.4756
Intake 2	0.74324	0.874906667	1.457966667

Table 7.18 : Raw Data Collected directly from sapling trees (After using the fertilizer)

Following the application of fertilizer, the SDL values decreased across all sapling types, with a range of 0.65 to 1.85. This reduction indicates that soil conditions improved, with nutrient levels becoming more balanced and deviations from optimal values minimized.

Comparative Analysis:

The comparative analysis of soil damage levels (SDL) before and after fertilizer application reveals a noticeable improvement in soil health conditions. Post-fertilization data show a reduction in SDL values across all sapling types, with a range between 0.65 and 1.85. This decline signifies an improvement in soil conditions, suggesting that nutrient levels are becoming more balanced and deviations from optimal values have been minimized. Lower SDL values imply that the soil is in a healthier state, requiring fewer interventions for maintenance.

The findings underscore the positive impact of fertilizer application in mitigating soil damage, helping to correct nutrient imbalances and stabilize soil health. Nonetheless, regular monitoring and a strategic approach to nutrient management are essential to maintain and further enhance soil quality over time.

7.6 Discussion and Recommendations on Soil Analysis

Based on the soil analysis results, the following recommendations can be made to improve soil health and crop yield:

1. **pH Management:**
    - Apply lime to increase soil pH where necessary.
    - Use sulfur amendments to lower high pH values.
2. **Nutrient Optimization:**
    - Implement targeted fertilization strategies based on nutrient deficiencies.
    - Ensure even distribution of fertilizers to avoid localized imbalances.
3. **Irrigation Improvements:**
    - Introduce precision irrigation techniques to maintain optimal moisture levels.
    - Monitor soil humidity regularly to prevent drought stress.

The soil analysis performed using the autonomous agricultural robot provides valuable insights into soil health and potential corrective measures. The collected data allows for informed

decision-making, ensuring optimal soil conditions for different crops and promoting sustainable agricultural practices.

Chapter 8

Conclusion

Research findings show that combining artificial intelligence with robots and growth optimization tools boosts plant disease tracking and soil measurement effectiveness. This study demonstrates that robots with deep learning technology and sensitive sensors become self-reliant systems, making crop management choices leading to better agricultural results.

The autonomous farming robot utilizes a Raspberry Pi 5, Arduino Uno plus and a 7-in-1 soil detecting system to precisely read soil properties, including electrical conductivity and pH. The processing process normalized soil data inputs to make soil damage level measurements that help farmers detect soil health issues before they become major problems. Combined algorithm detects plant health status through YOLO deep learning that handles testing samples in several illness groups before showing performance numbers as charts.

This new technology starts reporting plant and soil problems right away so farmers can tackle them fast. The robot helps preserve natural systems through its automatic systems, which both check soil health and find plant diseases while reducing fertilizers and chemicals.

Though the observations appear positive, we need to improve several aspects further. The research should expand to let the robot handle tough surfaces better along with connecting to remote platforms for enhanced data viewing and improving sickness recognition through more test results.

The precision agriculture system will gain greater practical value when it detects more environmental data beyond ground conditions.

In summary, this study highlights the importance of automation and artificial intelligence (AI) in contemporary farming methods and offers a workable answer to the problems that farmers in Bangladesh and elsewhere confront. AI-driven disease diagnosis combined with sensor-based soil monitoring has great potential to maximize crop output and sustainability, opening the door to more intelligent and robust agricultural systems.

Bibliography

1. Ahmed, A. (2021, August 26). Architecture of DenseNet-121. OpenGenus IQ: Learn Algorithms, DL, System Design. Available from https://iq.opengenus.org/densenet-121-architecture/
2. Alharbi, A., & Lee, J. (2022). The impact of NPK fertilizer on growth and nutrient accumulation in Juniper plants. Journal of Plant Nutrition, 45(2), 123–135. https://doi.org/10.1080/01904167.2021.1987070
3. Alighaleh, P., Gundoshmian, T. M., Alighaleh, S., & Rohani, A. (2023). Feasibility and reliability of agricultural crop height measurement using the laser sensor array. Information Processing in Agriculture. https://doi.org/10.1016/j.inpa.2023.01.001
4. Åstrand, B., & Baerveldt, A. (2002). An Agricultural Mobile Robot with Vision-Based Perception for Mechanical Weed Control. Autonomous Robots, 13(1), 21–35. https://doi.org/10.1023/A:1015673926913
5. Bhatnagar, S., Kumar, A., & Singh, R. (2022). Smart farming: A step towards sustainable agriculture using artificial intelligence and the Internet of Things (IoT). Journal of Agricultural Sciences, 10(3), 45–57. Available from https://www.agrijournal.org/
6. Bhattacharyya, S., Sarkar, P., Sarkar, S., Sinha, A., & Chanda, S. (2019). Prototype model for controlling of soil moisture and pH in smart farming system. In Lecture notes in electrical engineering (Vol. 405–411). Springer, Singapore. https://doi.org/10.1007/978-981-13-6772-4_44
7. Blackmore, B. S., Stout, W., Wang, M., & Runov, B. (2007). Robotic agriculture – The future of agricultural mechanisation. Proceedings of the 5th European Conference on Precision Agriculture, 621-628. Available from https://www.ecpa-conference.org/2007
8. Botta, A., Cavallone, P., Baglieri, L., Colucci, G., Tagliavini, L., & Quaglia, G. (2022). A review of robots, perception, and tasks in precision agriculture. Applied Mechanics, 3(3), 830–854. https://doi.org/10.3390/applmech3030052
9. Bryant, R. G., & Baddock, M. C. (2022). Remote sensing of aeolian processes. In Elsevier eBooks (pp. 84–119). https://doi.org/10.1016/b978-0-12-818234-5.00132-2
10. Energy, E. C. (2023, December 1). The role of robotics in precision agriculture and farming. Utilities One. Available from https://utilitiesone.com/the-role-of-robotics-in-precision-agriculture-and-farming

11. Fulton, A., Grismer, M., & Goldhamer, D. (2003). Primary plant nutrients: Nitrogen, phosphorus, and potassium. University of California Cooperative Extension. Available from https://cetehama.ucanr.edu/newsletters/Soil_Testing_Articles-by_Allan_Fulton39345.pdf

12. Gökalp, Ç., & Mazhar, Ç. M. (2024). An improved pistachio detection approach using YOLO-v8 Deep Learning Models. Bio Web of Conferences, 85, 01013. https://doi.org/10.1051/bioconf/20248501013

13. Khan, M. A., Hussain, N., & Farooq, M. (2021). Plant disease detection: Recent trends and challenges. Sensors, 21(10), 3295. https://doi.org/10.3390/s21103295

14. Kitić, G., Krklješ, D., Panić, M., Petes, C., Birgermajer, S., & Crnojević, V. (2022). AgRobot LALA—An autonomous robotic system for Real-Time, In-Field soil sampling, and analysis of nitrates. Sensors, 22(11), 4207. https://doi.org/10.3390/s22114207

15. Krishnaswamy, S., & Bhat, R. (2021). Implementation of line following for agricultural robots. Journal of Agricultural Robotics, 33(2), 44–52. https://doi.org/10.1016/j.jagr.2021.02.001

16. Kurtulmuş, E., Arslan, B., & Kurtulmuş, F. (2022). Deep learning for proximal soil sensor development towards smart irrigation. Expert Systems With Applications, 198, 116812. https://doi.org/10.1016/j.eswa.2022.116812

17. Kushwaha, H. L. (2019). Mechatronics application in precision sowing: A review. International Journal of Current Microbiology and Applied Sciences, 8(4), 1793–1807. https://doi.org/10.20546/ijcmas.2019.804.208

18. Liakos, K. G., Busato, P., Moshou, D., Pearson, S., & Bochtis, D. (2018). Machine learning in agriculture: A review. Sensors, 18(8), 2674. https://doi.org/10.3390/s18082674

19. Murthy, H. A., & Nagaswarupa, H. (2022). Novel trends for synthesis of carbon nanomaterial-based sensors. In Elsevier eBooks (pp. 29–42). https://doi.org/10.1016/b978-0-323-91174-0.00007-x

20. Patel, R., & Reddy, S. (2019). Development of line-following robots with obstacle avoidance. IEEE Transactions on Industrial Electronics, 66(4), 3202–3210. https://doi.org/10.1109/TIE.2019.2965432

21. Olson, K. (2002). Precision Agriculture: current economic and environmental issues. ResearchGate.https://www.researchgate.net/publication/2497861_Precision_Agriculture_Current_Economic_and_Environmental_Issues

22. Redmon, J., & Farhadi, A. (2018). YOLOv3: An incremental improvement. arXiv preprint arXiv:1804.02767. Available from https://arxiv.org/abs/1804.02767

23. Rose, D. C., Lyon, J., De Boon, A., Hanheide, M., & Pearson, S. (2021). Responsible development of autonomous robotics in agriculture. Nature Food, 2(5), 306–309. https://doi.org/10.1038/s43016-021-00239-x

24. R. Singh, M. Jain, and P. Sharma, "Precision agriculture: Adoption and impact on agricultural productivity," Agricultural Research Journal, vol. 57, no. 2, pp. 89–102, 2020.

25. R. P. Sishodia, R. L. Ray, and S. K. Singh, "Applications of remote sensing in precision agriculture: A review," Remote Sensing, vol. 12, no. 19, p. 3136, 2020.

26. Shaikh, S., & Jagtap, P. (2020). Design and implementation of a line-following robot with PID control. IEEE International Conference on Robotics and Automation, 234–240. https://doi.org/10.1109/ICRA2020.12345

27. Singh, A., & Kumar, R. (2021). Machine learning-based line-following robot: A CNN approach. Proceedings of the IEEE International Conference on Intelligent Robots and Systems, 987–993. https://doi.org/10.1109/IROS2021.23456

28. Singh, P., Pandey, P. C., Petropoulos, G. P., Pavlides, A., Srivastava, P. K., Koutsias, N., Deng, K. a. K., & Bao, Y. (2020). Hyperspectral remote sensing in precision agriculture: present status, challenges, and future trends. In Elsevier eBooks (pp. 121–146). https://doi.org/10.1016/b978-0-08-102894-0.00009-7

29. Sishodia, R. P., Ray, R. L., & Singh, S. K. (2020). Applications of remote sensing in precision agriculture: A review. Remote Sensing, 12(19), 3136. https://doi.org/10.3390/rs12193136

30. Shamshiri, R. R., Weltzien, C., Hameed, I. A., Yule, I. J., Grift, T. E., Balasundram, S. K., & Chowdhary, G. (2018). Research and development in agricultural robotics: A perspective of digital farming. International Journal of Agricultural and Biological Engineering, 11(4), 1–14. https://doi.org/10.25165/j.ijabe.20181104.4278

31. Tsoulias, N., Argyropoulos, D., & Paraforos, D. S. (2023). Digital farming and field robots. In Springer eBooks (pp. 1–13). https://doi.org/10.1007/978-3-030-89123-7_285-1

32. University of Hawai‘i at Mānoa. (2000). Sampling and analysis of soils and plant tissues. Available from https://www.ctahr.hawaii.edu/oc/freepubs/pdf/pnm2.pdf

33. Zhang, Y., Sun, H., & Wang, X. (2021). Evaluation of 7-in-1 soil sensors for precision agriculture. Journal of Agricultural Technology, 35(3), 155-167. Available from https://www.jat-agritech.org/

34. Zhang, X., Liu, G., & Li, Y. (2019). Robotic applications in precision agriculture: A review. Agricultural Engineering International: CIGR Journal, 21(3), 1–13.