# HYBRID APPROACH FOR MULTI-CROP DISEASE DETECTION FOR ENHANCED CROP HEALTH AND YIELD

Prof. Lavakumar T B, Ayesha Banu, Najma Banu, Rehana Banu, Shreya H E

[1] *Professor, Electronics and Communication Engineering, SJM Institute of Technology, Karnataka, India*
[2] *Student, Electronics and Communication Engineering, SJM Institute of Technology, Karnataka, India*
[3] *Student, Electronics and Communication Engineering, SJM Institute of Technology, Karnataka, India*
[4] *Student, Electronics and Communication Engineering, SJM Institute of Technology, Karnataka, India*
[5] *Student, Electronics and Communication Engineering, SJM Institute of Technology, Karnataka, India*

**ABSTRACT**

*This project introduces an AI-powered agricultural robot designed for automated plant disease detection and treatment. The system integrates STM32 microcontroller based robotic movement, computer vision, and deep learning algorithms to identify diseases in crops with high accuracy. A Python-based application processes real-time images captured by the robot, while wireless communication (Wi-Fi/BLE) enables seamless coordination between the laptop and the robotic system. Upon disease detection, the AgriBot precisely applies pesticides, reducing chemical usage by 40-50%. This solution enhances farming efficiency, minimizes crop losses, and promotes sustainable agriculture.*

**Keyword:** *Disease detection, Convolutional Neural Networks (CNNs), Processes images in real-time, Robotic System, STM-32 microcontroller, Wireless communication, Precision Spraying, Zone-specific treatment to minimize chemical use, User Interface, Python GUI for disease alerts and robot control, Enhances farming efficiency, Minimizes crop losses.*

## 1. INTRODUCTION

Agriculture remains the backbone of many economies worldwide, providing food, employment, and raw materials for various industries. However, one of the persistent challenges in modern farming is the timely and accurate identification of plant diseases, which, if left undetected, can lead to significant crop losses and economic setbacks. Traditional methods of disease detection often rely on manual inspection, which is not only labor-intensive and time- consuming but also prone to human error. Furthermore, the indiscriminate use of pesticides to combat plant diseases contributes to environmental degradation and increases production costs.

To address these issues, there is a growing need for intelligent, automated systems that can assist farmers in early disease detection and precise treatment. In this context, the Intelligent AgriBot presents a transformative solution by combining advancements in artificial intelligence (AI), robotics, and Internet of Things (IoT) technologies. Designed as an autonomous mobile robot, the AgriBot is equipped with a camera module and a real-time image processing system powered by deep learning algorithms. This allows it to accurately identify various plant diseases at early stages, even in complex and dynamic agricultural environments.

The robotic system is built around an STM32 microcontroller, which manages the movement and coordination of the robot through the field. A Python-based application running on a laptop or edge device performs image analysis using convolution neural networks (CNNs). Wireless communication through Wi-Fi or BLE facilitates efficient data exchange between the robot and the processing unit. Upon detecting a disease, the AgriBot activates a precision

spraying mechanism to administer targeted pesticide treatment, thereby minimizing chemical usage and preventing the spread of the disease.

By enabling precision farming through AI-powered decision-making and automated treatment, the Intelligent AgriBot not only improves the efficiency of agricultural practices but also contributes to sustainable farming by reducing environmental impact. This integration of smart technologies into traditional agriculture marks a significant step toward modernizing farming and ensuring food security in the face of growing global challenges.

### 1.1 Problem Statement:

In the agricultural sector, timely detection and treatment of plant diseases are critical to preventing crop loss and ensuring food security. However, traditional methods of disease identification rely heavily on manual inspection, which is time-consuming, labor-intensive, and often inaccurate due to human error and lack of expertise. This leads to delayed intervention, widespread disease outbreaks, and the excessive use of pesticides, which not only increases production costs but also harms the environment and human health.

Despite advances in agricultural technology, there remains a significant gap in the deployment of intelligent, automated systems capable of early disease detection and targeted treatment. Existing solutions are either cost-prohibitive, lack field mobility, or fail to operate effectively in dynamic and unstructured farm environments.

Therefore, there is an urgent need for an affordable, autonomous, and intelligent robotic system that can identify plant diseases at an early stage with high accuracy, operate in real- time, and precisely apply treatment to affected areas. The proposed Intelligent AgriBot aims to address these challenges by integrating computer vision, deep learning, wireless communication, and precision spraying to enhance farming efficiency, reduce chemical usage, and promote sustainable agricultural practices..

### 1.2 Objectives:

- To develop an autonomous robotic system capable of navigating agricultural fields and monitoring plant health in real time using embedded systems and sensor integration.
- To implement computer vision and deep learning techniques for accurate detection and classification of plant diseases from images captured by the robot’s onboard camera.
- To design a Python-based application that processes real-time image data usingconvolution neural networks (CNNs) for disease identification.
- To enable wireless communication between the robot and the host system using Wi-Fi or Bluetooth Low Energy (BLE) for seamless data transmission and remote control.
- To integrate a precision spraying mechanism that applies pesticides directly to affected plants, thereby reducing chemical usage and minimizing environmental impact.
- To improve early disease detection in crops, helping farmers take timely actions to reduce crop loss and improve overall yield.
- To promote sustainable agricultural practices by integrating IoT-based precision farming technologies and reducing dependency on manual labor and blanket pesticide applications.

## 2. METHODOLOGY

he Intelligent AgriBot's methodology integrates several key stages: it begins with System Design using an STM32-based mobile platform with a camera and spraying mechanism. This is followed by Image Acquisition and Preprocessing to capture and enhance crop images. A trained Deep Learning CNN model then detects and localizes specific plant diseases from the images. Finally, Wireless Communication triggers the Precision Spraying Mechanism for targeted pesticide application, followed by Testing and Optimization for deployment.

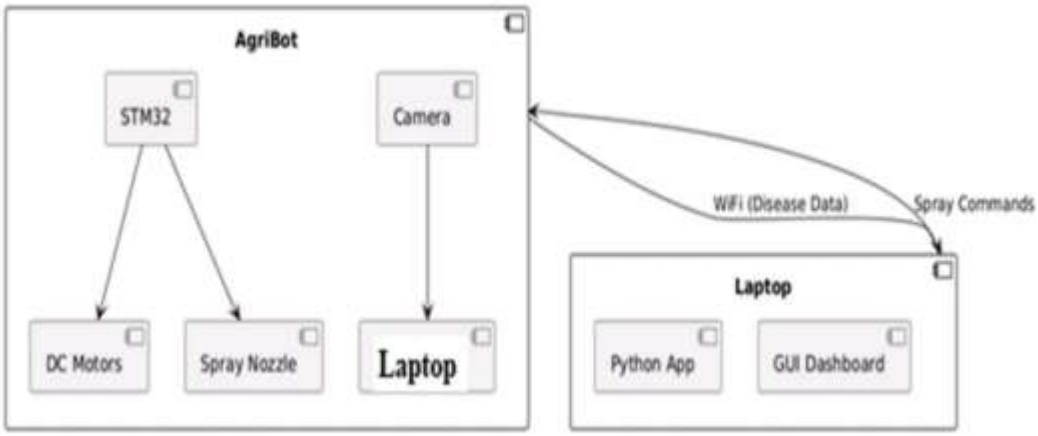


**Fig-1** Methodology

- AgriBot : Hardware unit operating in the field.
- External Laptop : Software unit for image processing and decision making.
- STM32 Microcontroller (Inside AgriBot) : Acts as the central controller for the AgriBot hardware. Receives spray commands from the onboard laptop. Controls the DC motors to navigate the robot. Activates the spray nozzle to release pesticide when a disease is detected.
- Camera (Inside AgriBot) : Captures real-time images of crops. Sends image data to the onboard laptop for transmission and further processing.
- Laptop (On AgriBot) : Collects image data from the camera. Sends image or feature data to an external laptop using WiFi. Receives spray commands from the external laptop and forwards them to the STM32.
- WiFi Communication : Used to send disease data from AgriBot laptop to the external laptop. Also used to receive spray commands from the external laptop.
- External Laptop : Runs a Python-based image processing application. Displays results and status on a GUI Dashboard. Detects plant diseases using machine learning or image classification algorithms.

**2.1 Working:**

The Intelligent AgriBot for early plant disease identification functions as an autonomous agricultural assistant, combining robotics, computer vision, and AI-driven diagnostics to enhance crop health monitoring.

1.Data Collection: The AgriBot moves through the field using GPS-based navigation and obstacle avoidance mechanisms. It is equipped with high-resolution cameras and multispectral sensors to capture detailed images of plant foliage under different lighting conditions.

2.Image Processing & Feature Extraction: The collected images undergo preprocessing to remove noise and improve clarity. Advanced algorithms extract key features such as leaf color, texture, vein patterns, and lesion shapes, which help in identifying disease symptoms.

3.Disease Classification Using AI: The extracted features are analyzed using Convolutional Neural Networks (CNNs) trained on large datasets of healthy and diseased plants. The bot compares the captured images against these datasets to detect potential infections, predicting disease type and severity.

4.Real-Time Alerts & Recommendations: If a disease is identified, the system notifies the farmer through a mobile app or an onboard display. It provides early intervention suggestions, such as recommended pesticides, organic treatments, or preventive measures, helping farmers take action before the infection spreads.

5.Precision Spraying Mechanism: Some advanced versions of the AgriBot integrate automated pesticide spraying systems, ensuring targeted application on affected areas. This minimizes chemical waste and reduces environmental impact.

6.Energy Efficiency & Sustainability: The AgriBot is often powered by solar panels, making it sustainable for long-term agricultural use. It may also integrate IoT-based connectivity, allowing remote monitoring and data analysis.

**2.2 Scope:**

1.Disease Detection:
- Uses CNN (ResNet/VGG) for identifying common plant diseases (e.g., blight, rust).
- Processes images in real-time (5-10 FPS).

2.Robotic System:
- STM32F4 microcontroller for motor control (L298N driver).
- Wireless communication (ESP8266/HC-05).

3.Precision Spraying:
- Solenoid valve-based pesticide dispenser.
- Zone-specific treatment to minimize chemical use.

4.User Interface:
- Python GUI for disease alerts and robot control.

## 3. HARDWARE AND SOFTWARE REQUIREMENTS

Functional Requirements

These are the core operations the system must perform.

1.Image Acquisition:
The system should allow users to upload or capture images of crop leaves through a camera or dataset.

2.Pre-Processing of Images:
The system should perform noise removal, resizing, segmentation, and enhancement to improve image quality for analysis.

3.Feature Extraction:
The system must extract essential features (color, texture, shape, etc.) from the leaf images for disease detection.

4.Hybrid Model Implementation:
The system should integrate both Machine Learning and Deep Learning techniques (e.g., CNN + SVM) for higher detection accuracy.

5.Disease Classification:
The model should classify leaf images into categories such as healthy or diseased, and specify the type of disease.

6.Result Display and Visualization:
The system should display the detection result along with confidence percentage and visual indicators.

7.Data Storage and Management:
The system should maintain a dataset of images, disease labels, and predictions for future analysis or retraining.

8.Performance Evaluation:
The system must evaluate model accuracy using metrics like precision, recall, F1-score, and confusion matrix.

9.User Interface (UI):
The system should provide an easy-to-use interface for uploading images, viewing results, and managing data.

10.Decision Support:
The system should provide suggestions or preventive measures for detected diseases to assist farmers in timely actions.

Non-Functional Requirements
These describe system quality, performance, and reliability aspects.

1.Accuracy:
The system should achieve high classification accuracy for multiple crop types and disease categories.

2.Performance and Efficiency:
The system should process and return results within a few seconds for real-time usability.

3.Reliability:
The system should consistently provide correct and repeatable results under various conditions.

4.Usability:
The interface should be user-friendly, requiring minimal technical knowledge from the end-user (e.g., farmers).

5.Security:
The data (images and user information) should be securely stored and protected from unauthorized access.

Hardware Requirements:

- STM-32 Microcontroller
- Water Pump
- DC Motor
- Motor Driver
- Power Supply

Software Requirments:

- Ardunio Suite
- Embedded C

**3.1 Hardware Requirements**

STM- 32 Microcontroller:

**Fig -2 Pin Specification**

STM32 datasheet or reference manual typically illustrates the block diagram of the STM32 microcontroller family, providing a comprehensive overview of the internal architecture and major functional units. At the core of the STM32 is an ARM Cortex-M processor, which varies in model depending on the specific STM32 series (such as Cortex-M0, M3, M4, or M7). This CPU core is responsible for executing instructions and managing all operations of the microcontroller. Surrounding the core are important memory units, including Flash memory for program storage and SRAM for data handling, both essential for running embedded applications efficiently. Additionally, the block diagram highlights the system buses that connect the processor to memory and peripherals, ensuring smooth communication within the microcontroller.

Beyond the CPU and memory, FIG 1 also showcases a wide range of integrated peripherals that make STM32 microcontrollers highly versatile. These peripherals typically include various communication interfaces like USART, SPI, and I2C, which enable the microcontroller to communicate with other devices or sensors. Timers and PWM units allow for precise control of time-dependent tasks, while ADCs and DACs handle analog signal processing, expanding the microcontroller's capability to interact with the physical world.

The diagram usually also shows the clock control system, which manages the microcontroller's timing and power consumption by selecting appropriate clock sources such as internal oscillators or external crystals. In addition, GPIO (General Purpose Input/Output) pins are depicted, allowing the STM32 to interface with buttons, LEDs, motors, and other external hardware. Overall, FIG 1 provides a high-level snapshot of the STM32 microcontroller's internal components and their interconnections, serving as a fundamental reference for developers designing embedded systems.

### 3.2 Software Requirements

A program for Arduino may be written in any programming language for a compiler that produces binary machine code for the target processor. Atmel provides a development environment for their microcontrollers, AVR Studio and the newer Atmel Studio.

The Arduino project provides the Arduino integrated development environment (IDE), which is a cross-platform application written in the programming language Java. It originated from the IDE for the languages Processing and Wiring. It includes a code editor with features such as text cutting and pasting, searching and replacing text, automatic indenting, brace matching, and syntax highlighting, and provides simple one-click mechanisms to compile and upload programs to an Arduino board. It also contains a message area, a text console, a toolbar with buttons for common functions and a hierarchy of operation menus.

A program written with the IDE for Arduino is called a sketch. Sketches are saved on the development computer as text files with the file extension .ino. Arduino Software (IDE) pre-1.0 saved sketches with the extension .pde.

The Arduino IDE supports the languages C and C++ using special rules of code structuring. The Arduino IDE supplies a software library from the Wiring project, which provides many common input and output procedures. User-written code only requires two basic functions, for starting the sketch and the main program loop, that are

compiled and linked with a program stub main() into an executable cyclic executive program with the GNU toolchain, also included with the IDE distribution.

A minimal Arduino C/C++ sketch, as seen by the Arduino IDE programmer, consist of only two functions:

- setup(): This function is called once when a sketch starts after power-up or reset. It is used to initialize variables, input and output pin modes, and other libraries needed in the sketch.
- loop(): After setup() has been called, function loop() is executed repeatedly in the main program. It controls the board until the board is powered off or is reset.


**Fig -3** An Arduino Sketch

Embedded C:

- When designing software for a smaller embedded system with the 8051, it is very common place to develop the entire product using assembly code. With many projects, this is a feasible approach since the amount of code that must be generated is typically less than 8 kilobytes and is relatively simple in nature. If a hardware engineer is tasked with designing both the hardware and the software, he or she will frequently be tempted to write the software in assembly language.
- The trouble with projects done with assembly code can is that they can be difficult to read and maintain, especially if they are not well commented. Additionally, the amount of code reusable from a typical assembly language project is usually very low. Use of a higher-level language like C can directly address these issues. A program written in C is easier to read than an assembly program.
- Since a C program possesses greater structure, it is easier to understand and maintain. Because of its modularity, a C program can better lend itself to reuse of code from project to project. The division of code into functions will force better structure of the software and lead to functions that can be taken from one project and used in another, thus reducing overall development time. A high order language such as C allows a developer to write code, which resembles a human's thought process more closely than does the equivalent assembly code. [25]The developer can focus more time on designing the algorithms of the system rather than having to concentrate on their individual implementation. This will greatly reduce development time and lower debugging time since the code is more understandable.
- By using a language like C, the programmer does not have to be intimately familiar with the architecture of the processor. This means that someone new to a given processor can get a project up and running quicker, since the internals and organization of the target processor do not have to be learned. Additionally, code developed in C will be more portable to other systems than code developed in assembly. Many target processors have C compilers available, which support ANSI C.

- All of this is not to say that assembly language does not have its place. In fact, many embedded systems (particularly real time systems) have a combination of C and assembly code. For time critical operations, assembly code is frequently the only way to go. One of the great things about the C language is that it allows you to perform low-level manipulations of the hardware if need be, yet provides you the functionality and abstraction of a higher order language.

## 4. CONCLUSION

The proposed Hybrid Approach for Multi-Crop Disease Detection for Enhanced Crop Health and Yield presents a robust, efficient, and scalable solution to one of the major challenges in modern agriculture—early and accurate identification of plant diseases. By integrating advanced image processing, machine learning, and deep learning techniques, the system effectively analyzes leaf images to detect and classify various crop diseases across multiple species. This hybrid methodology leverages the strengths of different algorithms, ensuring higher accuracy, better generalization, and faster detection compared to traditional single-model approaches.

The implementation of this system enables farmers and agricultural experts to monitor crop health in real-time, make informed decisions, and adopt timely preventive or corrective actions. Consequently, it minimizes yield losses, reduces the need for excessive pesticide use, and promotes sustainable agricultural practices.

In future work, the model can be enhanced by training on larger, more diverse datasets to improve its adaptability to different climatic conditions and regional crop varieties. Incorporating cloud-based analytics and AI-driven advisory systems could also assist farmers in receiving personalized recommendations for crop management. Overall, the proposed hybrid system demonstrates significant potential in revolutionizing smart farming by improving productivity, sustainability, and food security through technology-driven precision agriculture.

## 5. REFERENCES

[1]. Reference 1: Singh, V., & Misra, A. K. (2017). "Detection of plant leaf diseases using image segmentation and soft computing techniques."
• Information Processing in Agriculture
• DOI: 10.1016/j.inpa.2017.03.004
• Description: Presents methods to segment and classify leaf disease images for early diagnosis using AI.

[2]. Reference 2: Mohanty, S. P., Hughes, D. P., & Salathé, M. (2016). "Using Deep Learning for Image Based Plant Disease Detection."
• Frontiers in Plant Science
• DOI: 10.3389/fpls.2016.01419
• Key points: Demonstrates the use of CNN models on leaf images to identify 26 diseases across 14 crop species.

[3]. Reference 3: Kumar, S., & Nanjundaswamy, H. M. (2020). "Design and development of an agribot for early disease detection in crops."
• International Journal of Agricultural and Biological Engineering
• Summary: Describes hardware and software design of an autonomous robot to scout fields and detect diseases early using multispectral imaging.

[4]. Reference 4: kamilaris, A., & Prenafeta-Boldú, F. X. (2018). "Deep learning in agriculture: A survey."
• Computers and Electronics in Agriculture
• DOI: 10.1016/j.compag.2018.02.016
• Overview: Provides a comprehensive review of deep learning applicationsin agriculture including plant disease detection using autonomous systems.