Smart Agriculture Monitoring Automation System

BY:

Malak Walid Helmy (AI)

Omar Hassan Mansour (AI)

Sara Ahmed Awad (AI)

Omar AbdElAziz Mahmoud (Full Stack)

Seif ElDin Ahmed (Flutter)

Fady Atef Caesar (Embedded System)

Project Overview

This project aims to develop a comprehensive AI-powered system for precision agriculture, specifically focusing on tomato crops. The system will encompass crop disease detection, pest identification, irrigation scheduling, and automation with smart water valves. The system will leverage a variety of data sources and will integrate with both mobile and web interfaces for ease of use by farmers.

Data Collection

1. Image Data

- **Sources:**
  - **Ground Sensors:** Cameras mounted on tractors, within the embedded system or on the handheld devices.
- **Specifications:**
  - Capture high-resolution images of healthy and diseased tomato crops, pests at various life stages, and different field conditions.
  - Ensure images capture various lighting conditions, growth stages, and resolutions for robust model training.

2. Sensor Data

- **Types of Sensors:**
  - Soil moisture sensors
  - Temperature sensors
- **Data Collected:**
  - Real-time data on soil moisture, temperature and humidity.
  - Weather conditions like temperature and humidity.

Data Preprocessing

1. Image Data

- **Segmentation:** Implement image segmentation techniques to isolate regions of interest (plants, leaves).
- **Augmentation:** Apply data augmentation techniques (rotation, flipping) to increase training data diversity.
- **Formatting:** Convert images to a suitable format for the chosen AI model .

2. Sensor Data

- **Cleaning:** Clean sensor data to remove noise and outliers.
- **Normalization:** Normalize data to ensure consistency and compatibility with AI models.

AI Model Development

1. Crop Disease Detection and Pest Identification

- **Algorithm:** Use Convolutional Neural Networks (CNNs) for image recognition tasks.
- **Training:**
  - Train separate CNN models for disease and pest identification.
  - Utilize pre-trained models like VGG16 & CNN or VGG19 for faster training and leverage transfer learning.
  - Fine-tune these models with the specific tomato image dataset.

2. Irrigation Scheduling

- **Algorithm:** Employ Long Short-Term Memory (LSTM) networks for time-series forecasting.
- **Training:**
  - Train an LSTM model on historical sensor data and crop yield information.
  - Model will learn the relationship between environmental factors and water needs, predicting optimal irrigation schedules.

Smart Valve Integration

- **Hardware:**
  - Equip irrigation systems with smart valves controllable remotely.
  - Connect these valves to a central hub or directly to the cloud platform.
- **Integration:**
  - Integrate the irrigation scheduling AI model with the cloud platform or local control system.
  - Automatically adjust smart valve settings based on the AI-generated schedule.
- **Mobile App:**
  - Display the AI-generated irrigation schedule alongside real-time sensor data and valve status.

    - Allow farmers to manually override or adjust the automated schedule through the app.

Evaluation and Improvement

- **Monitoring:** Continuously monitor AI model performance in real-world scenarios.
- **Feedback:** Collect feedback from farmers and agronomists to identify areas for improvement.
- **Retraining:** Retrain models with new data to enhance accuracy over time.

Embedded Systems

- **Sensor Nodes:** Create sensor nodes for monitoring soil moisture, temperature and humidity in agricultural fields.

Mobile Application

- **Functionality:** Develop a mobile app that connects to the agricultural monitoring system.
- **Features:**
    - Remote monitoring of crops.
    - Real-time alerts.
    - Control of irrigation systems.

Web Interface

- **Dashboard:**
    - Real-time monitoring: Display live data from sensors in the fields.
    - Historical data analysis: View historical trends, analyze crop growth patterns, and identify correlations between environmental factors and crop performance.
    - Customizable alerts: Set up alerts for critical events via email or SMS.
    - Remote control and automation: Remotely control irrigation systems.
    - Weather forecasts: Integrate weather forecast data to assist in decision-making.

Integration Using Flask API

1. **Data Collection and Preprocessing:**

- Use Flask API endpoints to collect and preprocess image and sensor data.
- Upload images and sensor data to the server via the API.

2. **Model Training and Prediction:**
   - Train AI models using the preprocessed data.
   - Implement endpoints for model predictions .
3. **Smart Valve Control:**
   - Create endpoints to control smart valves
   - Integrate with the irrigation scheduling model to automate valve control.
4. **Mobile App and Web Interface:**
   - Develop API endpoints for the mobile app to fetch real-time data, alerts, and irrigation schedules.
   - Create a web interface using Flask to display sensor data, historical analysis, and provide remote control capabilities.

<u>The Embedded Systems Chapter:</u>

Smart Agricultural System

The smart agricultural system utilizes various hardware components to monitor and manage agricultural activities effectively. This system aims to automate and optimize irrigation, monitor soil moisture levels, and ensure efficient water usage, all while providing real-time feedback to the user.

Hardware Components

1. **Arduino Mega2560 R3**
   - **Description**: The Arduino Mega2560 R3 is the central microcontroller used in this system. It is based on the ATmega2560 microcontroller and offers more input/output pins compared to other Arduino boards. This allows it to interface with multiple sensors and actuators simultaneously.
   - **Function**: It reads data from various sensors, processes this data, and controls other components such as LEDs, buzzer, and the water valve system.
2. **Buzzer (Optional)**
   - **Description**: A small piezoelectric buzzer.
   - **Function**: It provides audio alerts for various system statuses or warnings, such as low water levels, low battery, or hardware errors.

3. **Precision Digital Temperature & Humidity Sensor Module (DHT22)**

- **Description**: The DHT22 is a highly accurate digital sensor module that measures temperature and humidity.
- **Function**: It provides precise and real-time temperature and humidity readings for environmental monitoring and control systems.

4. **4 LEDs**
    - **Description**: Four LEDs used to indicate different system statuses.
    - **Functions**:
        - **WiFi**: Indicates the status of the WiFi connection.
        - **Water Level**: Indicates the water level status in the storage tank.
        - **Battery Level**: Indicates the battery charge level.
        - **Hardware Error**: Alerts to any hardware malfunction.
5. **Water Valve System Components**

    - **ESP32-CAM Development Board (with camera) OV2640(Optional)**
        - **Description**: Is a low-cost microcontroller module with an integrated OV2640 camera, offering Wi-Fi and Bluetooth capabilities.
        - **Function**: It captures high-resolution images and streams video, enabling wireless image processing and transmission for various IoT applications.

    - **ESP32**
        - **Description**: A powerful WiFi-enabled microcontroller module with dual-core processing.
        - **Function**: Connects the system to a WiFi network, enabling remote monitoring and control.
    - **Soil Moisture Sensor Module**
        - **Description**: Detects the moisture content in the soil.
        - **Function**: Measures soil moisture levels to decide when to activate the water valve.
    - **Single Channel Relay**
        - **Description**: An electromechanical switch.
        - **Function**: Controls the water valve (DC motor or solenoid valve) by switching it on or off based on signals from the Arduino.

- **Battery 12V**
  - **Description**: A 12-volt battery.
  - **Function**: Powers the water valve system, ensuring it operates independently of the main power supply.
- **DC Motor / Solenoid Valve**
  - **Description**: The actuator that controls water flow.
  - **Function**: Opens or closes the water supply to irrigate the plants based on the soil moisture levels.
- **LED**
  - **Description**: An indicator LED.
  - **Function**: Provides visual feedback for various states, such as valve operation or system status.
- **LDR (Light Dependent Resistor)**
  - **Description**: A sensor that detects light levels.
  - **Function**: Could be used to detect day/night cycles for time-based irrigation control.
- **Potentiometer 10k**
  - **Description**: A variable resistor.
  - **Function**: Used to adjust sensor thresholds or fine-tune the system's sensitivity to inputs.
- **Jumper Wires**
  - **Description**: Wires used for connecting different components in the circuit.
  - **Function**: Facilitates the connections between the sensors, Arduino, relay, and other components.
- **Ultrasonic Sensor**
  - **Description**: A sensor that measures distance using ultrasonic waves.
  - **Function**: Measures the water level in the tank, ensuring there is sufficient water for irrigation.
- **Required resistors**
  - 330 ohm for LEDs
  - 10k ohm for other components