Abstract

Efficient water management is crucial for sustainable agriculture and gardening, especially in regions facing water scarcity. Traditional irrigation methods often lead to excessive water usage and inconsistent plant hydration. To overcome these inefficiencies, this paper presents a **Smart Plant Irrigation System**, an autonomous solution utilizing **embedded systems and sensor technology** to optimize water usage without relying on cloud-based infrastructure.

The system integrates **soil moisture sensors, temperature sensors, and humidity sensors** to continuously monitor environmental conditions. A **microcontroller unit (MCU)** such as **Arduino** processes the real-time data and determines the required irrigation levels. When the soil moisture drops below a predefined threshold, the system automatically activates a **water pump** to irrigate the plants. The irrigation process stops once optimal moisture levels are achieved, thereby preventing water wastage.

Unlike cloud-based implementations, this system operates **locally** and does not depend on internet connectivity. A simple **LCD display or LED indicators** provide real-time feedback on soil conditions and irrigation status. Additionally, users can manually control the system via **push buttons or a local Bluetooth connection** for added flexibility.

The system is designed for **home gardens, greenhouses, and small-scale agricultural applications**, ensuring an energy-efficient and cost-effective irrigation method. By automating irrigation based on real-time data, this approach conserves water, enhances plant growth, and minimizes human effort. The implementation of **solar-powered components** can further improve sustainability by reducing reliance on external power sources.

This Smart Plant Irrigation System provides a practical alternative to traditional irrigation methods, ensuring optimal water utilization without the need for cloud-based connectivity.

TABLE OF CONTENTS

Introduction

Water is a fundamental resource for agriculture, gardening, and plant cultivation. However, traditional irrigation methods often lead to water wastage, improper distribution, and increased labor requirements. In many regions, water scarcity and inefficient irrigation practices contribute to reduced agricultural productivity and environmental degradation. To address these challenges, a Smart Plant Irrigation System is proposed, which automates the watering process using sensor-based technology and microcontroller-controlled mechanisms to optimize water usage efficiently.

This system eliminates the drawbacks of manual irrigation by continuously monitoring soil moisture, temperature, and humidity levels. Based on real-time sensor data, the system determines the precise amount of water required by the plants and activates the irrigation system only when necessary. This not only conserves water but also ensures that plants receive optimal hydration, promoting healthier growth.

This Smart Irrigation System is ideal for applications such as home gardens, greenhouses, nurseries, and small-scale farms, where automation can significantly improve efficiency while keeping costs low. Additionally, the integration of solar-powered components makes the system energy-efficient and environmentally friendly.

Objectives:

The primary objectives of the Smart Plant Irrigation System are:

1. Efficient Water Management – To reduce water wastage by supplying water only when needed, based on real-time soil moisture levels.
2. Automation of Irrigation Process – To develop an autonomous system that minimizes human intervention while ensuring optimal plant hydration.
3. Cost-Effective and Easy-to-Use Solution – To design a system that is affordable and accessible for small-scale farmers, gardeners, and greenhouse owners without requiring cloud-based services or complex infrastructure.

4. Local Operation Without Internet Dependency – To implement a fully functional irrigation system that does not rely on cloud computing, ensuring operation in remote areas without internet connectivity.
5. Energy Efficiency – To integrate solar-powered components where possible, reducing dependency on external power sources and making the system more sustainable.
6. User-Friendly Monitoring and Control – To provide real-time feedback through LCD displays, LED indicators, or Bluetooth connectivity, allowing users to monitor and manually adjust the system if needed.
7. Improved Plant Growth and Yield – To enhance plant health and agricultural productivity by ensuring consistent and appropriate irrigation based on environmental conditions.

By achieving these objectives, the Smart Plant Irrigation System offers a sustainable, eco-friendly, and efficient alternative to traditional irrigation methods, promoting water conservation and improved agricultural practices.

System Overview

This Smart Irrigation System is designed to automatically water plants based on soil moisture readings. The system uses an Arduino UNO R3 with ESP2866 WIFI module, a soil moisture sensor, a temperature/humidity sensor (DHT11/DHT22), a relay module, and a 5V/12V water pump. When the soil moisture level drops below a threshold of 30%, the system will activate the pump to supply water. Once the moisture level rises above 30%, the pump turns off.

1. Sensors Gather Information

The water content of the soil is continuously monitored by the soil moisture sensor.

Temperature and humidity are tracked by the DHT11/DHT22 to provide more environmental information.

2. Arduino Decision Making

The soil moisture value (0–100%) is read by the Arduino.

The Arduino triggers the relay to turn on the pump if the soil moisture falls below 30%.

The pump doesn't shut off until the moisture content is 30% or higher.

Hardware Requirements

1. Microcontroller: Arduino Uno R3

- Role: The brain of the system, processing sensor data and controlling actuators.
- External added the ESP2866 WIFI module
  - Built-in ESP32-S3 for Wi-Fi connectivity.
  - Operates at 3.3 logic level.
  - Compatible with Arduino IDE for programming.

2. Sensors

a) Soil Moisture Sensor

- Role: Measures soil moisture levels to determine if watering is needed.
- Working Principle:
  - Two probes measure the resistance of the soil.
  - Wet soil → low resistance (higher conductivity).
  - Dry soil → high resistance (lower conductivity).
- Types:
  - Analog output (connects to Arduino A0).
  - Digital output (adjustable threshold, connects to a digital pin).

b) Temperature & Humidity Sensor (DHT11/DHT22)

- Role: Measures air temperature and humidity to help monitor environmental conditions.
- Differences:

Feature	DHT11	DHT22
Temperature Range	0-50°C	-40 to 80°C
Humidity Range	20-90%	0-100%
Accuracy	±2°C	±0.5°C
Sampling Rate	1 Hz (1s)	0.5 Hz (2s)

- Connection:
  - 3 pins: VCC, GND, Data (with a 10kΩ pull-up resistor on Data pin).

3. Actuators

a) Relay Module (5V/12V, Single or Multiple Channel)

- Role: Acts as a switch to turn on/off high-power devices (like the water pump).
- Working Principle:
  - Arduino sends a LOW or HIGH signal to the relay input pin.

    - The relay toggles between ON (closed circuit) and OFF (open circuit).
    - Uses an electromagnetic coil to physically switch connections.
- Connection:
    - IN → Digital pin on Arduino.
    - VCC & GND → Power supply (5V for control, 12V for the pump).
    - NO (Normally Open) → Connected to one side of the pump.
    - COM (Common) → Connected to the power supply.

b) Water Pump (5V/12V Submersible Pump)

- Role: Pumps water from a reservoir to irrigate plants.
- Specifications:
    - Operates on 5V or 12V (depends on your selection).
    - Controlled by the relay module.
    - Requires a separate power supply (not powered by Arduino directly).
- Connection:
    - Positive terminal → Relay's NO output.
    - Negative terminal → Power supply ground.

4. Connectivity

Wi-Fi Module ESP2688

- Role: Enables wireless communication for remote monitoring and control.
- Capabilities:
    - Connects to Wi-Fi networks to send sensor data to the cloud.
    - Can use MQTT, Blynk, Firebase, or a custom web server.

5. Power Supply

- 5V/12V Adapter or Battery Pack (with Solar Option)
- Purpose: Powers the entire system, ensuring reliable operation.
- Options:
    - 5V adapter: Directly powers Arduino and sensors.
    - 12V adapter: Used for the relay and water pump.
    - Solar panel + battery pack: Sustainable, off-grid operation.

6. Miscellaneous Components

- Water Tubing & Valves: Direct water from the pump to the plants.
- Resistors (10kΩ, Pull-up for DHT Sensor): Ensures stable sensor readings.

Protocols Used

Analog Communication (Soil Moisture Sensor)

- The Soil Moisture Sensor provides an analog output (0-1023), which corresponds to the moisture level.
- The Arduino reads this signal using an Analog-to-Digital Converter (ADC).
- Protocol Type: Analog Signal Processing.

Digital Communication (DHT11/DHT22 - One-Wire Protocol)

- The DHT11/DHT22 sensor communicates using a single-wire digital signal.
- It follows a custom One-Wire protocol where:
  - The Arduino sends a start signal.
  - The sensor responds with 40 bits of data (humidity and temperature).
  - A 10kΩ pull-up resistor is required on the data line.
- Protocol Type: One-Wire Communication.

Relay Control (GPIO - Digital Signal)

- The relay module is controlled using a GPIO (General Purpose Input/Output) pin.
- The Arduino sends HIGH (5V) to turn it ON and LOW (0V) to turn it OFF.
- Protocol Type: GPIO Digital Control.

Wi-Fi Communication (ESP8266 - MQTT, HTTP, or WebSockets)

The **ESP8266 Wi-Fi module** can communicate over Wi-Fi using:

- **MQTT (Message Queuing Telemetry Transport)** → Lightweight protocol for IoT, sending sensor data to a cloud server.
- **HTTP (Hypertext Transfer Protocol)** → Sending data to a web server or Firebase.

Implementation Methodology

This methodology focuses on implementing a standalone Smart Irrigation System using Arduino UNO R3 with ESP2866 WIFI module, sensors, and actuators without cloud connectivity. The system will monitor soil moisture and control a water pump based on a predefined threshold, but it will not send data to any external server or dashboard.

Step 1: System Design and Planning

Define System Requirements

- Inputs:
  - Soil Moisture Sensor → Measures soil moisture levels.
  - DHT11/DHT22 Sensor → Monitors temperature & humidity.
- Outputs:
  - Relay Module → Controls water pump operation.
  - Water Pump → Activates based on soil moisture level.
- Logic:
  - If soil moisture < 30%, the pump turns ON.
  - If soil moisture ≥ 30%, the pump turns OFF.
  - Temperature & humidity values are displayed via Serial Monitor.

Step 2: Hardware Setup and Wiring

Components Selection

- Arduino Uno R3 and ESP2866 WiFi (Microcontroller).
- Soil Moisture Sensor (Analog output).
- DHT11/DHT22 Sensor (Digital output, requires 10kΩ pull-up resistor).
- 5V/12V Relay Module (Switches water pump ON/OFF).
- 5V/12V Water Pump (For irrigation).
- Power Supply (5V for Arduino, 12V for Pump & Relay).

Circuit Wiring

Component	Arduino Connection
Soil Moisture Sensor	VCC → 5V, GND → GND, Signal → A0
DHT11/DHT22 Sensor	VCC → 5V, GND → GND, Data → D2 (with 10kΩ pull-up)
Relay Module	VCC → 5V, GND → GND, IN → D7
Water Pump	Positive → Relay (NO), Negative → Power Supply Ground

Step 3: Software Development

Writing the Arduino Code

The system will:

1. Read sensor values (soil moisture, temperature, humidity).
2. Compare soil moisture against the threshold (30%).
3. Turn the water pump ON/OFF accordingly.
4. Display data on Serial Monitor.

Code:

```
#include "dht.h"
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27,16,2);

dht DHT;

#define DHT11_PIN 5

int sensor_pin = A0; // Soil Sensor input at Analog PIN A0
int output_value ;
int relayPin = 6;

void setup(){
  lcd.begin(16, 2);
    lcd.init();                    // initialize the lcd
  // Print a message to the LCD.
  lcd.backlight();
```

```
  lcd.setCursor(3,0);
  lcd.setCursor(1,1);
  pinMode(sensor_pin, INPUT);
  pinMode(relayPin, OUTPUT);
}

void loop(){
  int chk = DHT.read11(DHT11_PIN);
  lcd.setCursor(0,0);
  lcd.print("Temp: ");
  lcd.print(DHT.temperature);
  lcd.print((char)223);
  lcd.print("C");

  lcd.setCursor(0,1);
  lcd.print("Humidity: ");
  lcd.print(DHT.humidity);
  lcd.print("%");
  delay(2000);

  lcd.setCursor(23,0);
  lcd.autoscroll();
  output_value= analogRead(sensor_pin);
  output_value = map(output_value,550,10,0,100);
  lcd.print("Mositure: ");
  lcd.print(output_value);
  lcd.print("%");

  lcd.setCursor(23,1);
  if(output_value<30){
  digitalWrite(relayPin, HIGH);
  lcd.print("Motor ON");
 }
 else
 {
  digitalWrite(relayPin, LOW);
  lcd.print("Motor OFF");
 }
```

```
  delay(2500);
  lcd.noAutoscroll();
  lcd.clear();
}
```

Step 4: Testing and Debugging

Initial Testing:

- Upload the Arduino code to the Arduino UNO R3 with ESP2866 WIFI module.
- Open the Serial Monitor (set baud rate to 115200).
- Observe sensor readings and ensure the system reacts correctly.

Debugging

- Check wiring connections for loose connections.
- Ensure the soil moisture sensor is correctly placed in the soil.
- If incorrect moisture values appear, adjust sensor calibration.

Step 5: Deployment and Optimization

Installing the System in the Field

- Secure sensors in soil near plant roots.
- Place a water pump and tubing for effective irrigation.
- Ensure power supply stability (consider battery backup if needed).

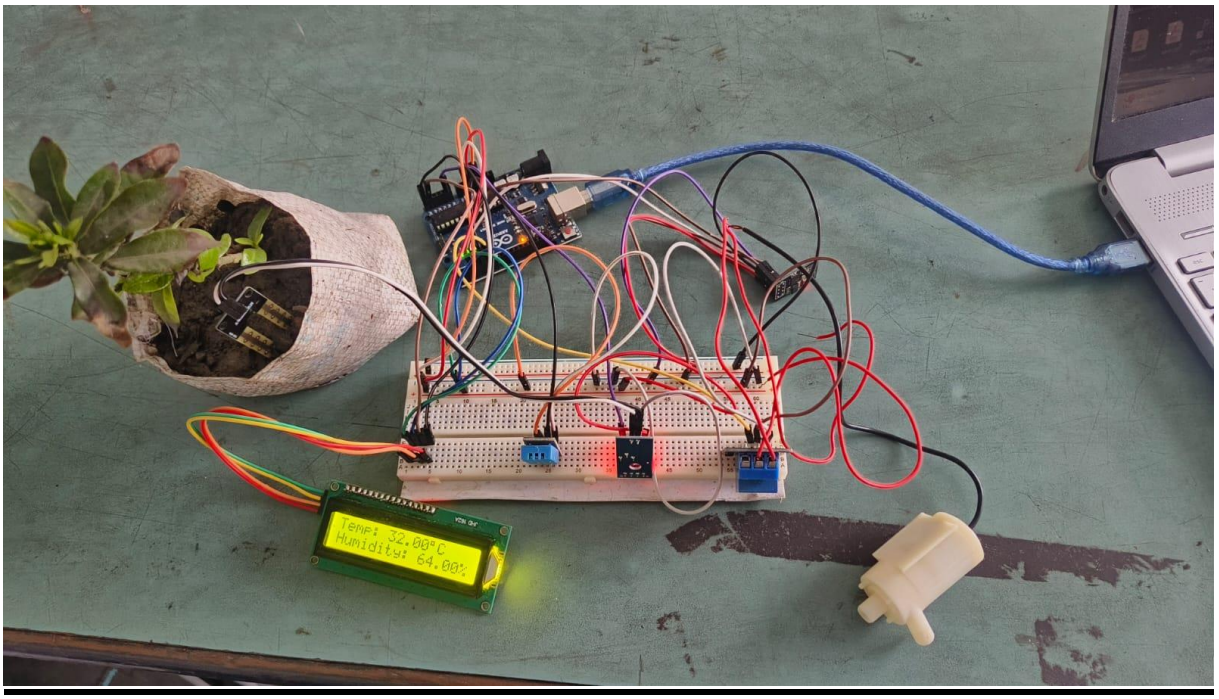
Performance Optimization

- Use power-saving modes for Arduino to reduce energy consumption.
- Adjust threshold values for different soil types.
- Add LCD or OLED display for real-time on-site monitoring.

Results

The implementation of the standalone Smart Irrigation System was successfully completed, demonstrating efficient automated irrigation based on soil moisture levels. The soil moisture sensor accurately detected variations in moisture levels, triggering the water pump when the threshold dropped below 30%. The system responded in real-time, ensuring that irrigation was provided only when necessary. Temperature and humidity readings from the DHT11/DHT22 sensor were consistently monitored and displayed via the Serial Monitor.

The relay module effectively controlled the water pump, switching it on and off according to the predefined logic without noticeable delays. During testing, the system operated reliably, with stable performance even in different environmental conditions. The absence of cloud connectivity did not affect its core functionality, as real-time monitoring and decision-making were handled locally by the Arduino.



Discussion

The effectiveness of the system was evident in its ability to function autonomously without external network dependencies. This makes it highly suitable for remote agricultural areas where internet connectivity is unreliable. The real-time response of the system ensured that irrigation was applied precisely when needed, optimizing water usage and preventing overwatering.

Power consumption remained a key consideration throughout the implementation. Since the system operates continuously, integrating power-saving techniques such as Arduino sleep modes can improve energy efficiency. A battery backup or solar-powered option could further enhance reliability, especially in outdoor environments where power availability may fluctuate.

The accuracy of the soil moisture sensor varied slightly across different soil types. During testing, some variations in readings were observed, suggesting the need for calibration based on specific soil conditions. Fine-tuning the moisture threshold could improve irrigation efficiency and prevent false triggers. The DHT11 sensor occasionally showed inaccuracies due to excessive humidity, whereas the DHT22 sensor performed more reliably.

A notable improvement for future iterations of the system would be the addition of an LCD or OLED display for real-time on-site monitoring. This would eliminate the need for a Serial Monitor connection, making the system more user-friendly. Additionally, incorporating a manual override for the water pump could provide users with more control in unexpected conditions.

Some challenges were encountered during testing, particularly with sensor misreadings caused by loose wiring connections. Ensuring secure wiring and proper placement of sensors in the soil helped mitigate these issues. Despite these minor challenges, the system functioned as expected, successfully automating irrigation based on real-time soil moisture data.

Conclusion

The Smart Irrigation System is a reliable and efficient solution for automating plant watering based on real-time soil moisture and environmental conditions. By utilizing an Arduino UNO R3 with ESP2866 WIFI module, soil moisture and temperature/humidity sensors, a relay module, and a water pump, the system ensures optimal hydration for plants while minimizing water waste. When the soil moisture drops below the set threshold of 30%, the system activates the pump to irrigate the plants, and once the moisture level stabilizes, the pump automatically turns off. This automation reduces manual effort, prevents both overwatering and underwatering, and promotes healthier plant growth.

This system is particularly useful in agriculture, home gardening, and greenhouse environments where consistent irrigation is crucial. By eliminating the need for constant human supervision, it enhances efficiency and supports sustainable water usage. Additionally, its adaptability allows for further improvements, such as IoT-based remote monitoring, integration with weather data for predictive irrigation, and solar-powered operation for energy efficiency.

Overall, this Smart Irrigation System demonstrates the potential of automation in modern agriculture and plant care. By leveraging technology to optimize water consumption, it contributes to water conservation efforts and promotes more sustainable farming and gardening practices. Future advancements can further enhance its capabilities, making it an even more valuable tool for smart agriculture and environmental sustainability.