Soil Moisture Detection Using Raspberry Pi

Abstract

Soil moisture monitoring is essential for optimizing water use in agriculture, gardening, and plant care. Inadequate moisture can lead to plant stress, reduced yields, or plant death, while over-watering can promote root rot and reduce soil aeration. This project utilizes a Raspberry Pi microcontroller and a soil moisture sensor to measure soil moisture levels in real-time and display or alert users about the moisture status. Unlike automated irrigation systems, this project focuses solely on detecting and reporting soil moisture without any automatic watering response. This setup is ideal for users looking to monitor soil conditions and make informed watering decisions, and it provides a foundation for future enhancements such as automated watering or Internet of Things (IoT) integration for remote monitoring.

Objectives and Description

Objectives:

1. Accurate Moisture Monitoring: Provide a reliable, real-time soil moisture monitoring solution.
2. User-Friendly Data Display: Present moisture levels in a format that users can easily interpret.
3. Adaptability for Future Upgrades: Enable potential integration with IoT platforms or automated watering systems.

Project Description:

The system is designed to detect and report soil moisture levels through the use of a Raspberry Pi and a soil moisture sensor. The Raspberry Pi acts as the main processing unit, converting sensor data into digital values that are compared against set moisture thresholds. The system can provide visual or alert-based notifications, allowing users to respond to soil moisture levels as needed. In this project, the sensor is used strictly for monitoring and does not trigger any watering mechanism. This system is beneficial for small-scale gardeners, researchers, and hobbyists interested in soil monitoring, and it can serve as a prototype for larger agricultural applications.

Methodology

The project follows a series of systematic steps:

1. Hardware Setup:
    - The Raspberry Pi is set up as the primary controller, interfacing with the soil moisture sensor through GPIO (General-Purpose Input/Output) pins.
    - A soil moisture sensor is embedded in the soil to measure its moisture content, which varies with the amount of water present.
    - Additional components, such as resistors and jumper wires, may be needed to complete the wiring connections between the Raspberry Pi and the sensor.
2. Data Collection:
    - The soil moisture sensor generates an analog signal proportional to the moisture content in the soil.
    - This analog signal is read by the Raspberry Pi, which can be configured to interpret these readings as digital values (using an Analog-to-Digital Converter if necessary).
3. Data Processing:
    - Using Python programming, the analog sensor output is converted into digital values that represent moisture levels.
4. Output Display:
    - The processed moisture data can be displayed on the user screen.

Components

1. Soil Hygrometer:
    - This sensor measures soil moisture based on electrical conductivity. Moisture levels impact the soil's electrical resistance, with higher moisture reducing resistance.
    - The hygrometer has two probes, one acting as a reference and the other as the measurement probe, allowing it to gauge the moisture level accurately.
2. Soil Moisture Sensor:
    - This sensor outputs analogue readings that vary with the amount of water present in the soil.
    - The readings can range from 0 (dry soil) to 1023 (highly saturated soil), and the Raspberry Pi interprets these readings to indicate moisture levels.
3. Raspberry Pi Board:
    - The Raspberry Pi microcontroller is responsible for managing sensor data and converting it into readable values.
    - GPIO pins on the board allow it to connect to various sensors, making it versatile for multiple applications.
    - The Raspberry Pi is programmed using Python, a language known for its simplicity and compatibility with Raspberry Pi hardware.

Pictures of components:

Fig 1. Raspberry Pi Board

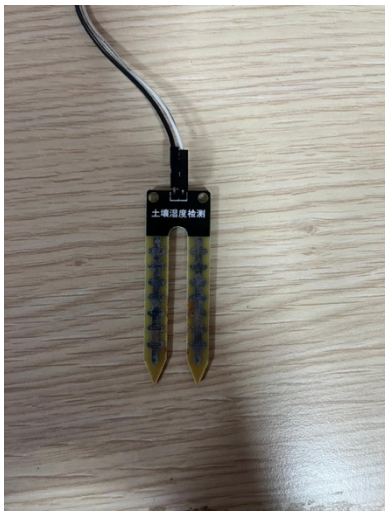


Fig 2. Soil Moisture Sensor

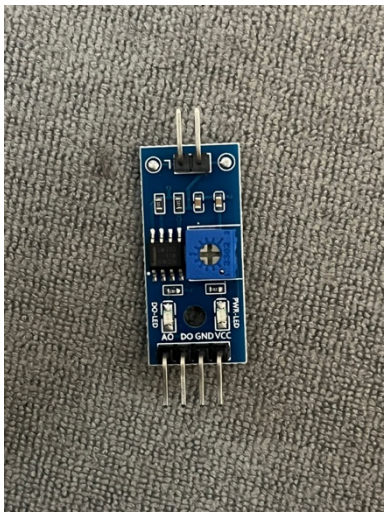


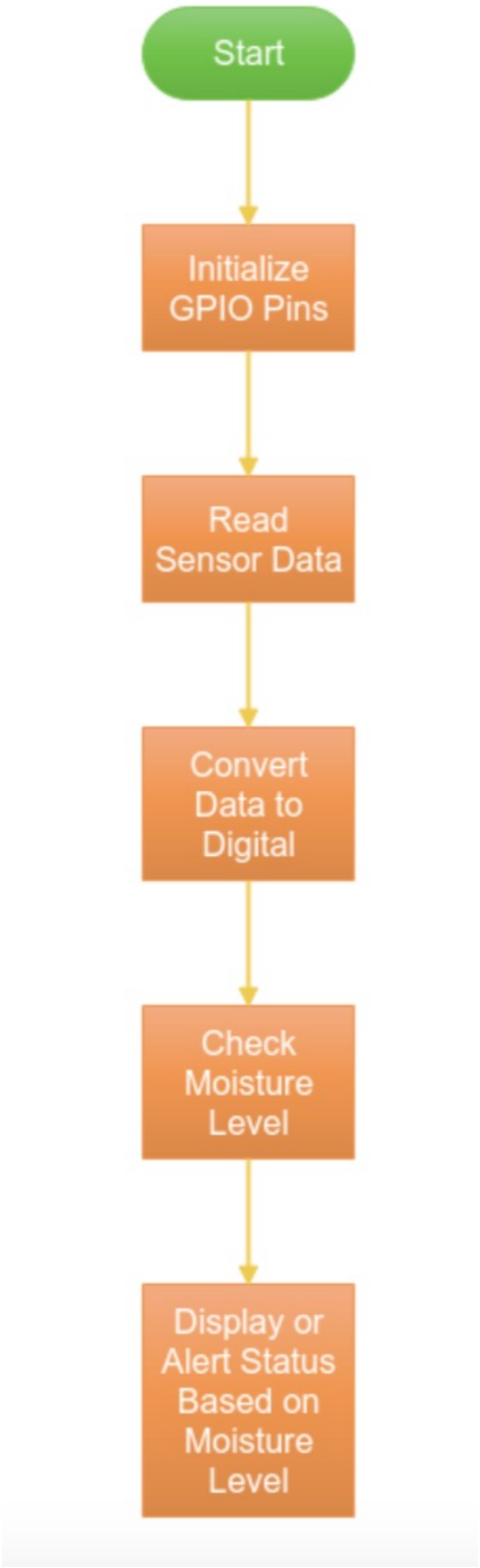
Fig 3. Soil Hygrometer

Algorithm and Flowchart

Algorithm:

1. Initialize GPIO Pins: Set up the Raspberry Pi GPIO pins to receive data from the soil moisture sensor.
2. Read Analog Data: Retrieve the analogue output from the soil moisture sensor.
3. Convert to Digital: If an Analog-to-Digital Converter (ADC) is used, convert the analogue signal to a digital value; otherwise, interpret the GPIO data directly.
4. Analyze Moisture Levels:
    - Compare the digital reading to predefined thresholds:
        - Dry Soil: Display or alert with a "Low Moisture" warning.
        - Moist Soil: Indicate "Moderate Moisture" for regular conditions.
        - Saturated Soil: Indicate "High Moisture."
5. Display Results: Show the status on an LCD screen or send an alert to the user.
6. Repeat: Continuously loop the process to update the data in real time.

Flowchart:


Start
Initialize GPIO Pins
Read Sensor Data
Convert Data to Digital
Check Moisture Level
Display or Alert Status Based on Moisture Level

Code and Explanation

```
import RPi.GPIO as GPIO

import time

GPIO.setmode(GPIO.BCM)

sensor_pin = 17

GPIO.setup(sensor_pin, GPIO.IN)

try:

    while True:

        if GPIO.input(sensor_pin) == GPIO.LOW:

            print("Water detected")

        else:

            print("No water detected")

        time.sleep(1)

except KeyboardInterrupt:

    print("Program terminated by user")

finally:

    GPIO.cleanup()
```

Explanation of code:

- RPi.GPIO is a Python library for controlling GPIO pins on the Raspberry Pi. It allows us to interact with the pins in a simple and organized way.
- time is a built-in Python module that provides time-related functions, such as pausing the program execution.
- GPIO.BCM refers to the Broadcom chip-specific numbering system, where each pin is identified by its actual Broadcom GPIO number rather than its physical location on the board.

- sensor_pin = 17: Assigns GPIO pin 17 to sensor_pin, which will read the data from the moisture sensor.
- GPIO.setup(sensor_pin, GPIO.IN): Configures GPIO pin 17 as an input pin. This means the Raspberry Pi will read incoming signals from this pin.
- try Block: The program is wrapped in a try block to allow safe exiting by the user.
- while True Loop: Runs indefinitely, constantly checking the sensor status.
- if GPIO.input(sensor_pin) == GPIO.LOW:
  - GPIO.input(sensor_pin) reads the current state of sensor_pin.
  - If the sensor output is GPIO.LOW (meaning a low voltage or signal from the sensor), the program interprets this as "water detected" and prints that message.
  - If the sensor reads a HIGH state, it indicates "no water detected."
- time.sleep(1): Pauses the loop for 1 second between readings to avoid excessive rapid polling.
- except KeyboardInterrupt: This block handles a KeyboardInterrupt (e.g., pressing Ctrl + C), which stops the loop and prints a message saying the program was terminated by the user.
- finally Block: After exiting the loop, GPIO.cleanup() is called to reset the GPIO pin configuration, preventing issues in subsequent runs of the program.

Demonstration

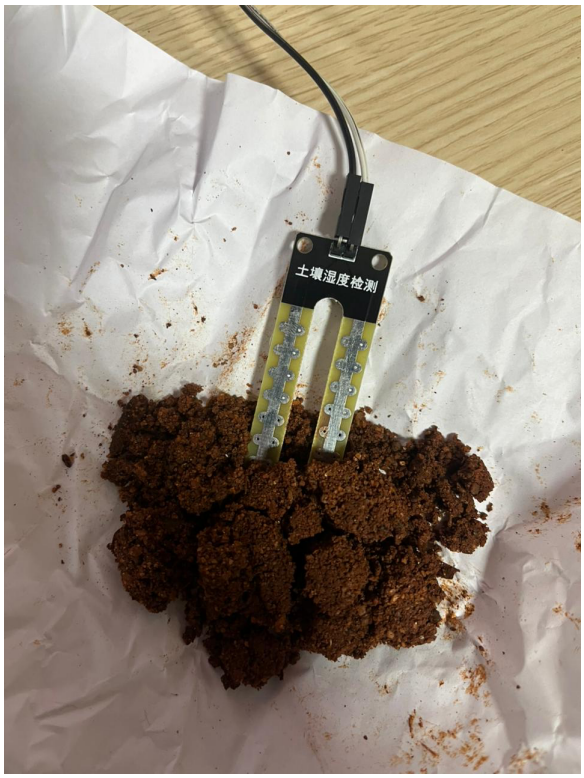


Fig 4. Soil sensor in soil

Fig 5. Demonstration

Output

- When the code is running, the output will depend on the sensor's reading. If moisture is detected (LOW state on sensor_pin), it prints:

  water detected

- If no moisture is detected (HIGH state on sensor_pin), it prints:

  no water detected

- The output will refresh every second as long as the program is running. If the program is stopped by the user, it will display:

  program terminated by user

Future Enhancements

1. IoT Integration:
   - Connect the system to an IoT platform to allow remote monitoring of soil moisture levels.
   - This could allow users to view real-time data from a mobile app or web dashboard, making it convenient for large farms or remote areas.
2. Automated Irrigation:
   - Expand the system to include a relay and water pump for automated irrigation based on soil moisture data.
   - When moisture levels fall below a set threshold, the pump could be activated to water the soil until optimal moisture is reached.
3. Data Analytics and Prediction:
   - Store moisture data over time to analyze patterns and predict irrigation needs.
   - Using machine learning algorithms, the system could eventually predict when watering will be necessary based on historical data and weather forecasts.
4. Mobile App Integration:
   - Develop a mobile application that receives real-time alerts on soil moisture levels, allowing users to monitor and respond to changes immediately.
   - Users could adjust moisture thresholds through the app to match specific plant needs.

Conclusion

The soil moisture detection system built using a Raspberry Pi and soil moisture sensor offers a cost-effective, easy-to-build solution for monitoring soil moisture levels. It provides timely insights into soil conditions, helping users make informed decisions about watering. This project is suitable for small-scale gardens, research labs, or hobbyists, and can be expanded for larger agricultural operations with further development. Future enhancements like IoT integration and automated irrigation make it a promising starting point for creating comprehensive soil health management systems. Through its flexible design, the project provides a practical and scalable approach to soil moisture monitoring.