Demid Dabizha 30891

ESP32-CAM Water Meter Reading Device

Abstract

This report presents a project aimed at developing a smart water metering system using computer vision technology. The project utilizes an ESP32-CAM module to capture images of a water meter and applies comuter vision to read the meter's digits. The system integrates with IoT infrastructure to enable remote monitoring and automatic reading of water consumption, facilitating efficient water usage management and billing. Additionally, the project includes designing a custom casing using Fusion 360, ensuring a durable and suitable housing for the device. The implementation aims to enhance accuracy, reduce labor, and provide real-time data access for better resource management.

1. Introduction

1.1 Background

Water metering is essential for monitoring water usage, managing resources, and billing consumers. Traditional methods of reading water meters manually are labor-intensive and prone to human error. By leveraging computer vision, it is possible to automate this process, thereby improving accuracy and efficiency. This technology is especially effective for use in cases where water meters are located in hard-to-reach places, for example, in a basement or hidden in a wall.

Computer vision, can interpret visual information and convert it into digital data. This advancement allows for the automation of tasks that were previously manual, significantly enhancing the efficiency and reliability of data collection.

1.2 Objective

The objective of this project is to design and implement a smart water metering system using the ESP32-CAM module. The system will capture images of a water meter, process these images to extract numerical readings, and transmit the data for remote monitoring. The ultimate goal is to provide a cost-effective, accurate, and scalable solution for modern water management systems.

2. Materials and Methods

2.1 Components

ESP32-CAM-MB: Docking Station for ESP32-CAM: The ESP32-CAM-MB is a docking station designed specifically for the ESP32-CAM compact camera module. This board features a USB-to-serial interface, simplifying the programming process. It includes two physical buttons, a reset button and a download button, which facilitate easy operation. The ESP32-CAM module plugs directly into this board using pin headers, creating a complete camera module that is both WLAN- and Bluetooth-

capable, and can be programmed directly via USB without the need for soldering or additional development boards.

ESP32-CAM: Compact Camera Module: The ESP32-CAM is a compact camera module that integrates seamlessly with the ESP32-CAM-MB. It is equipped with a high-resolution camera capable of capturing clear images for processing. This module features built-in WiFi and Bluetooth capabilities, making it ideal for IoT applications. The ESP32-CAM handles image capture and data transmission, enabling remote monitoring and control.

MicroSD Card (32GB): A 32GB MicroSD card is used for storing captured images and configuration files. It offers ample storage space and fast read/write speeds, ensuring efficient handling of image data. It is recomended to use 16 GB SD card, in my case 32 GB worked fine.

USB A to Micro USB Cable: This cable is used to connect the ESP32-CAM-MB to a power source and for initial setup. It ensures a stable power supply and facilitates programming and debugging by providing a direct connection to a computer or other programming device.

WiFi Network: A WiFi network is used for transmitting data from the ESP32-CAM-MB to a remote server or cloud service. It provides a stable and fast connection, enabling reliable data transmission. This network allows for remote monitoring and control, giving users access to water meter readings from anywhere. It also supports the integration of multiple devices, creating a comprehensive smart metering system.

Custom Casing: A custom casing, designed using Fusion 360, houses the ESP32-CAM-MB and the water meter. This casing protects the electronic components from environmental factors such as dust, moisture, and physical damage. It ensures proper alignment of the camera with the water meter for accurate image capture. The design is compact and providing easy access for maintenance and adjustments.

2.2 Setup and Configuration

Installation Overview

The setup of the ESP32-CAM water meter reading device involves several steps:

1. **Hardware Preparation**: Assemble and connect the necessary components.
2. **Firmware Installation**: Flash the firmware onto the ESP32-CAM.
3. **SD Card Preparation**: Flash the firmware on the SD card.
4. **System Startup**: Power on the device and verify its operation.

Each step is detailed below to guide through the installation and configuration process.

All needed codes for flashing can be found here: AI-on-the-edge-device

Hardware Preparation and ESP Firmware Installation

1. **Connecting the Hardware**: Mount the ESP32-CAM onto the ESP32-CAM-MB. Ensure all pins are connected correctly. There is no need for additional wires when using the ESP32-CAM-MB.

2. **Power Supply**: The ESP32-CAM requires a 5V power supply, typically provided via a USB power source.
3. **Entering Flashing Mode**: This is very important step. To enter flashing mode, its needed to connect a wire between GND and GPIO0 on the ESP32-CAM-MB (I used male jumper cable to connect GND and GPIO0). This is necessary for the initial firmware flashing.
4. **Flashing the Firmware**:
    - Download the required firmware files from the GitHub repository in a zip file.
    - Connect the ESP32-CAM-MB to the computer via a USB cable.
    - With use of Arduino IDE upload the firmware onto the ESP32-CAM.

SD Card Preparation

Note: it is recomeneded to use 16GB SD Card. But 32GB worked fine for me.

1. **Downloading Configuration Files**: Obtaining the `sd-card.zip` file from the GitHub repository and extracting its contents.
2. **Editing WiFi Configuration**:
    - Opening the `wlan.ini` file from the extracted contents.
    - Entering the WiFi SSID and password to allow the ESP32-CAM to connect to the network.
3. **Copying Files to SD Card**:
    - Copying the edited files onto the SD card.
    - Inserting the SD card into the ESP32-CAM.

System Startup

1. **Powering On**: Insert the SD card into the ESP32-CAM and power on the device.
2. **Verification**: After powering on, the onboard red LED will blink to indicate the connection status:

- **5 fast blinks**: Connection pending
- **3 slow blinks**: WLAN connection established

1. **Checking Connection**: Ensure the ESP32-CAM is connected to WiFi network by verifying the IP address.

2.3. Software Configuration

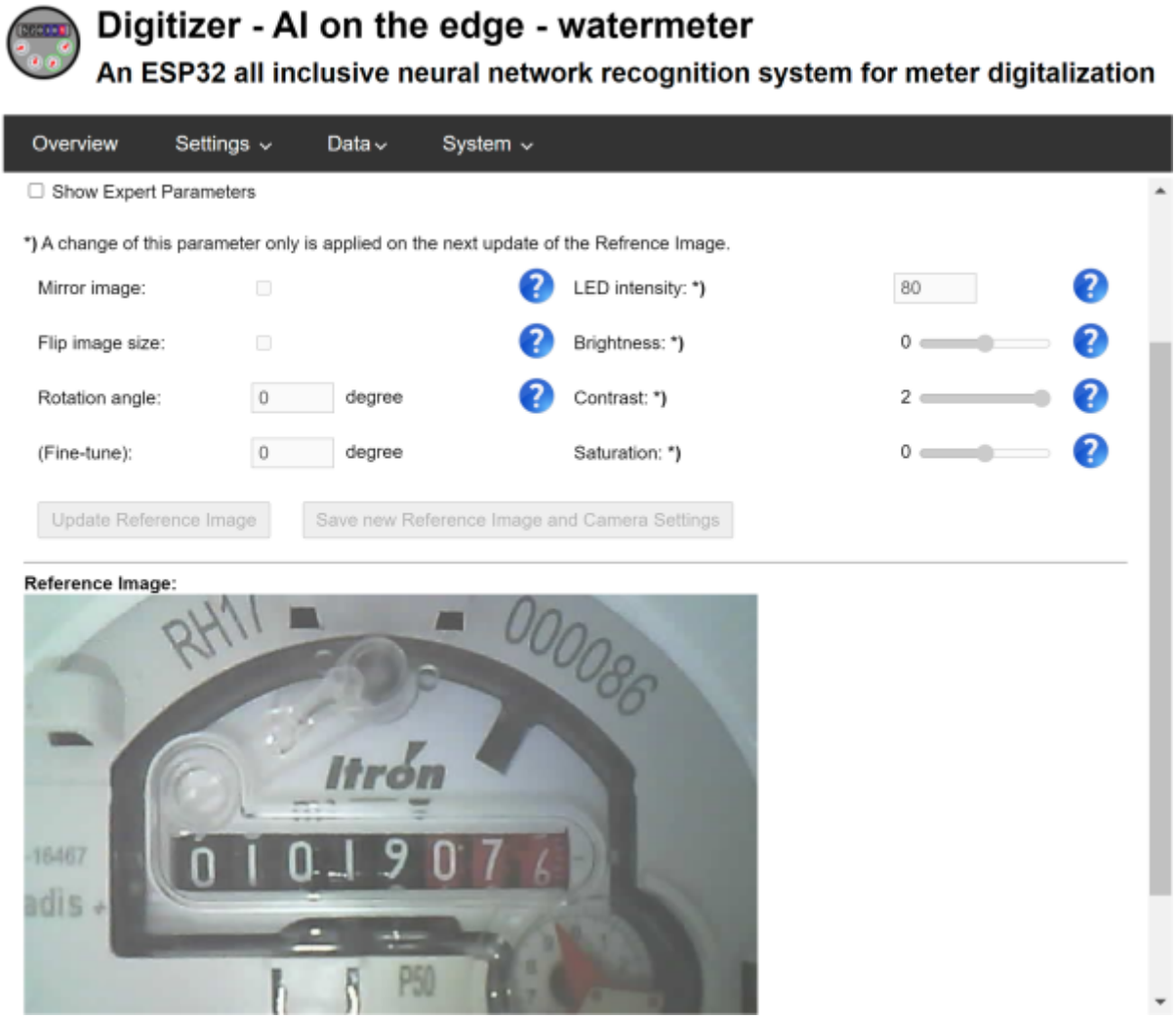
After verifying that the hardware setup is working correctly, the next step is the software configuration. This involves accessing the device's web interface, capturing a reference image, and setting up alignment references and Regions of Interest (ROIs).

Accessing the Web Interface

1. **Connect to the Device**: Open a web browser and enter the IP address assigned to the ESP32-CAM.
2. **User Interface**: The web interface will display setup settings, including image capture and configuration options.

Capturing a Reference Image

1. **Adjusting Camera Position**:
    - **Distance**: Position the ESP32-CAM at a distance where the entire water meter display is visible.
    - **Focus Adjustment**: Remove the glue on the camera lens and adjust the focal length until the image is clear.
2. **Image Quality Settings**:
    - **Brightness, Contrast, and Saturation**: Adjust these settings to ensure a high-quality image.
    - **Horizontal Alignment**: Ensure the numbers on the water meter are horizontally aligned.

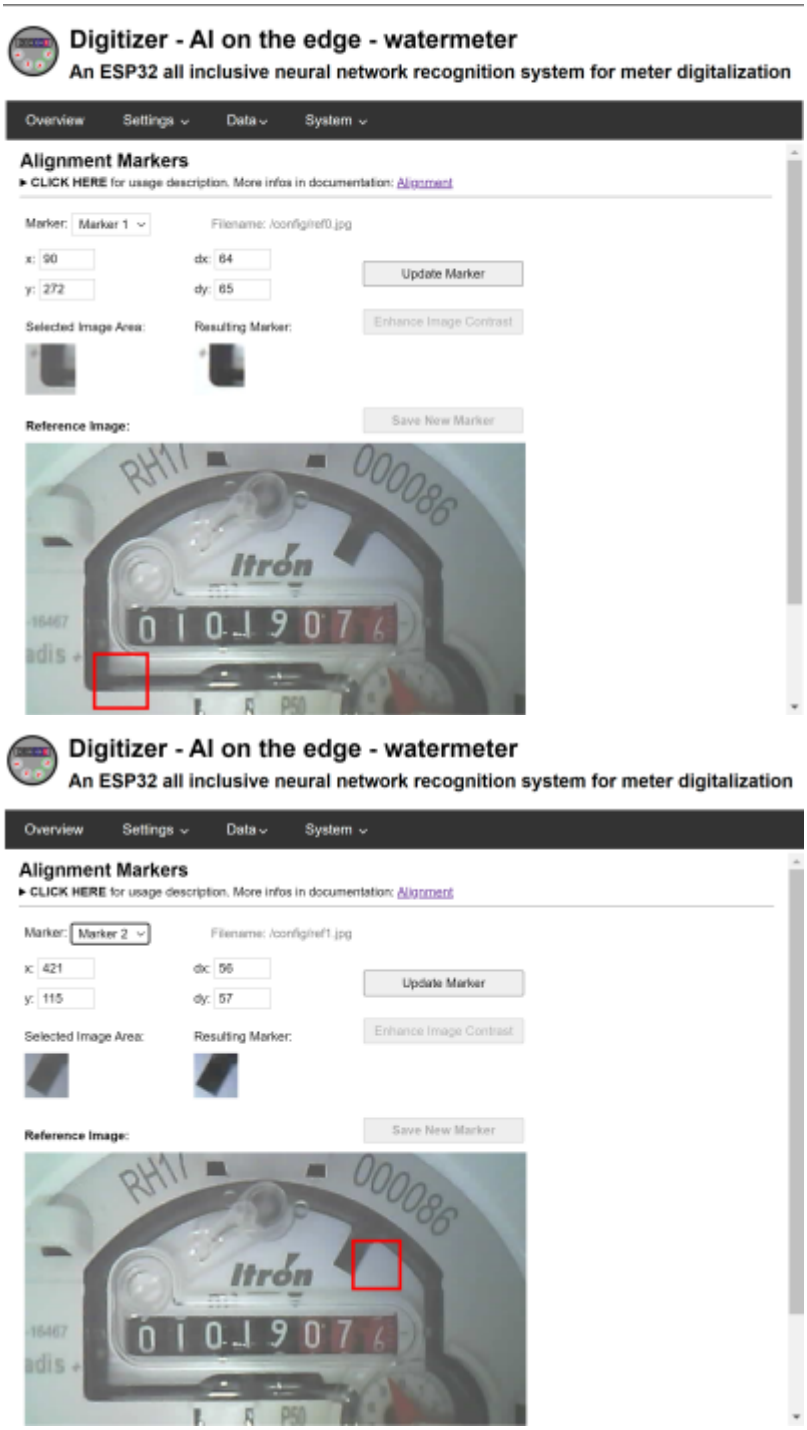


3. **Dealing with Reflections**:
   - **Camera Rotation**: Rotate the ESP32-CAM to minimize reflections on the meter.
   - **Diffuser Attachment**: Attach a diffuser, such as a piece of felt or parchment paper, to the LED to reduce reflections.
   - **LED Intensity**: Lower the LED intensity to further minimize glare.

Setting Up Alignment References

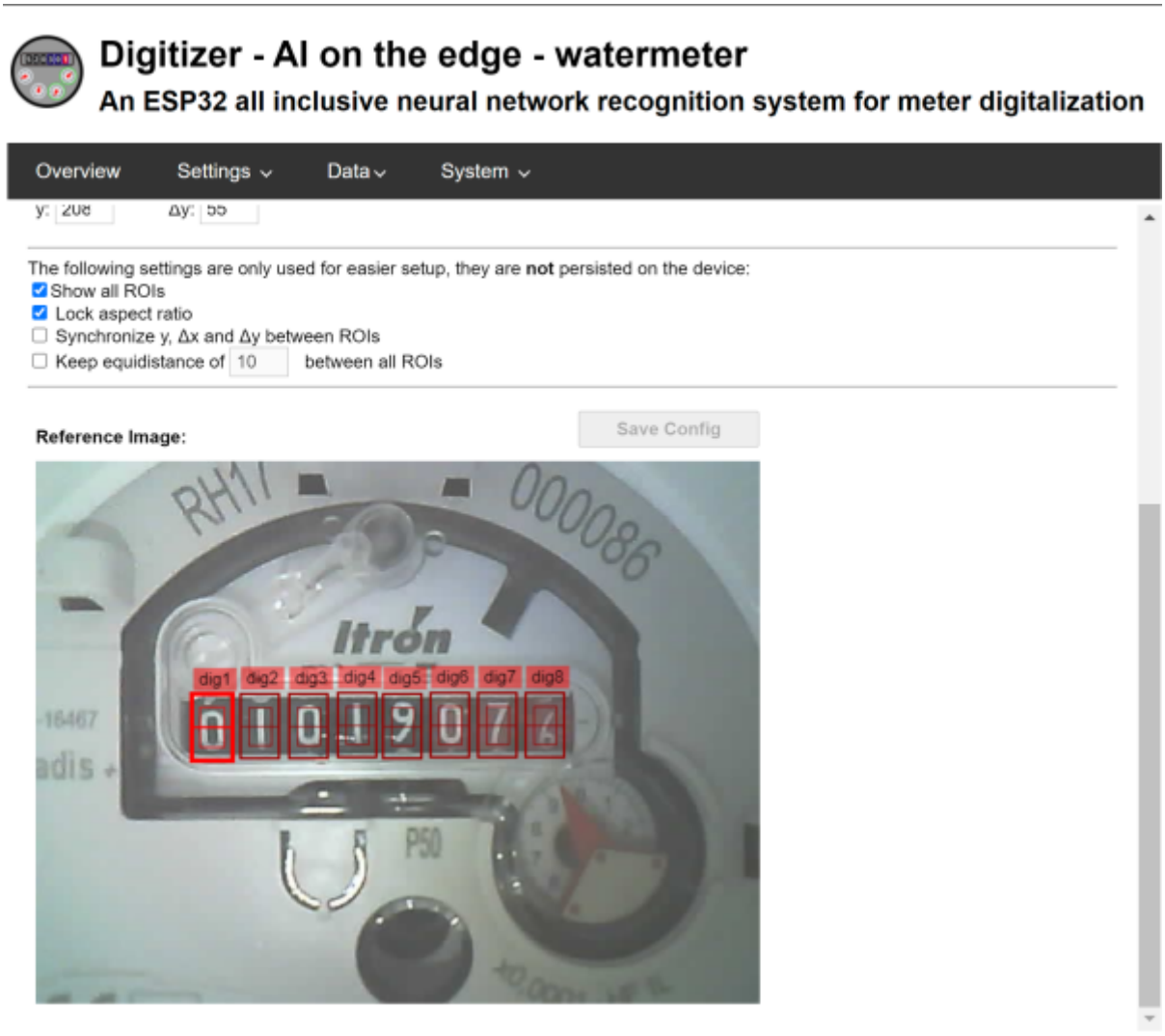
Alignment references ensure that each captured image is correctly aligned with the reference image, improving the accuracy of readings.

1. **Define Alignment Structures**: Select two distinct structures on the water meter to act as alignment references.
2. **Image Adjustment**: The software will use these references to shift and rotate each captured image to match the reference coordinates.



Defining Regions of Interest (ROIs)

ROIs are specific areas of the image where the digits of the water meter are located.



1. **Define ROIs for Each Digit**:
   - **Digit Location**: Carefully define a separate ROI for each digit on the water meter display.
   - **Precision**: Ensure the ROIs are accurately placed to capture each digit clearly.

3. Casing Design

Design Tools and Software

The casing for the ESP32-CAM module was designed using Autodesk Fusion 360 and prepared for 3D printing with UltiMaker Cura. These tools were selected for their robust design capabilities and ease of use in preparing models for 3D printing.

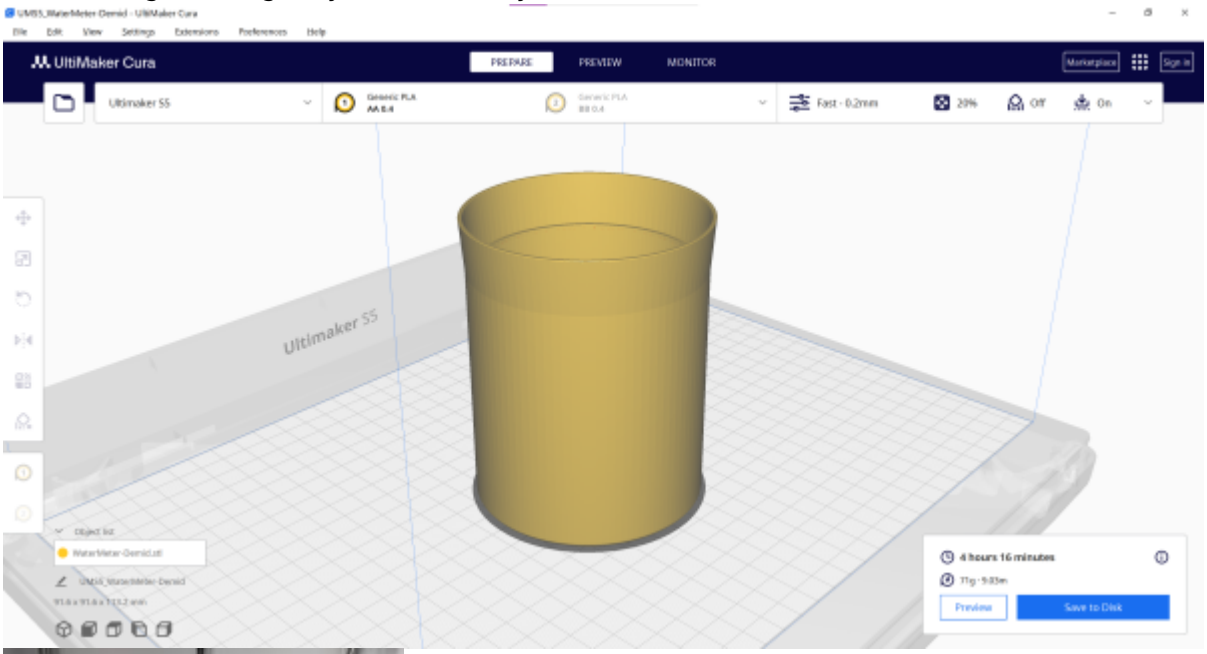
Design Considerations

The casing was designed specifically to fit the given water meter, ensuring a snug and secure fit. Several key factors were taken into consideration during the design process:

1. **Distance Between ESP32-CAM and Water Meter**: Ensuring the correct distance between

the ESP32-CAM and the water meter was crucial for capturing clear images of the meter readings. The design includes precise measurements to maintain this distance consistently.

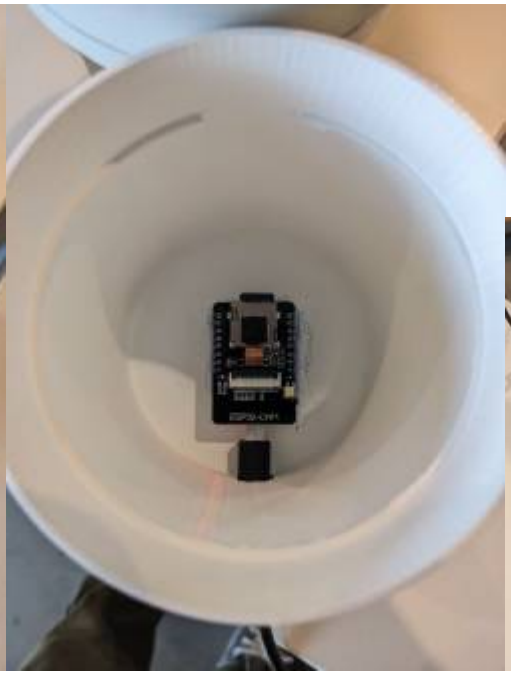
2. **Easy Removal**: The casing is designed to be easily removable, allowing for maintenance and adjustments without hassle. It can be detached and reattached without tools, ensuring convenience for the user.
3. **Power Supply Access**: The casing includes a hole for the power supply, enabling easy connection and disconnection of the power source. This ensures that the device remains powered without compromising the integrity of the casing.
4. **Durability and Protection**: The material and design of the casing ensure that the ESP32-CAM module is protected from environmental factors such as dust and moisture. This helps in maintaining the longevity and reliability of the device.



4. Results

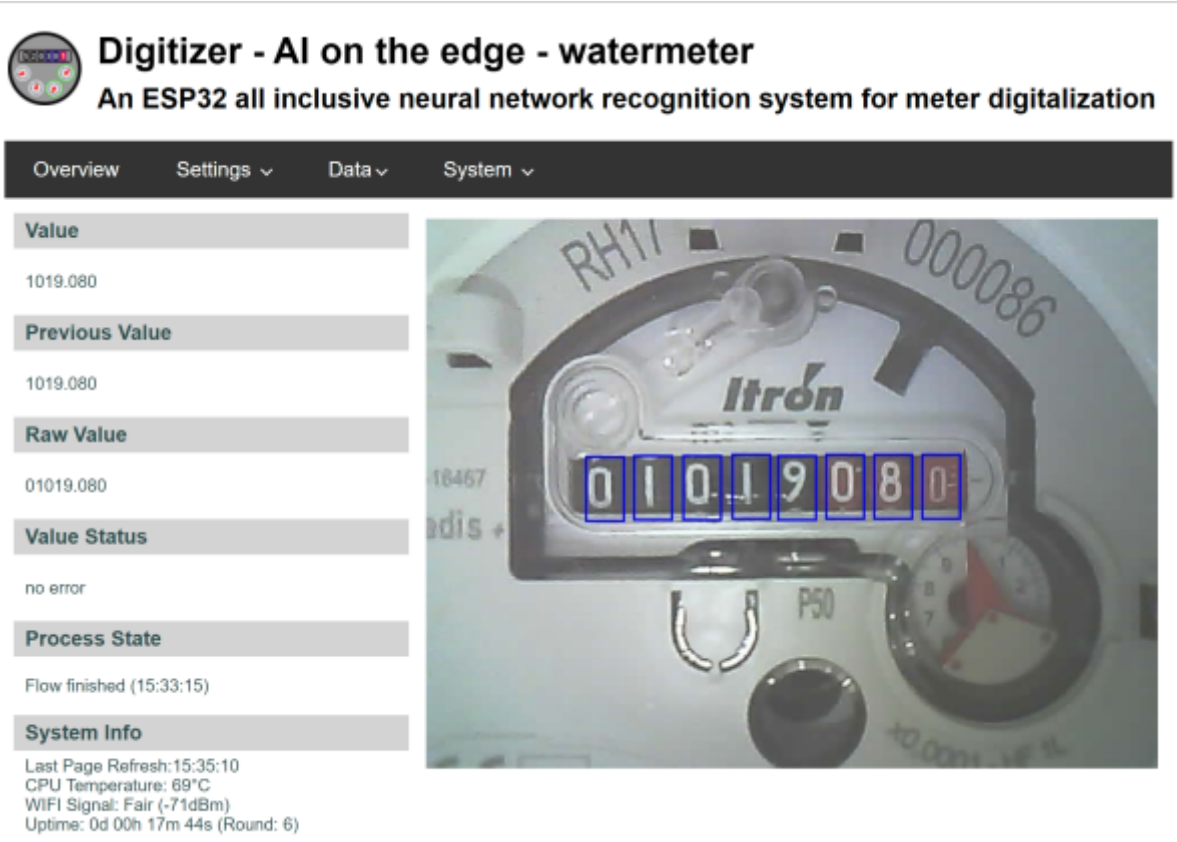
Final Physical Setup

The final physical setup of the smart water metering system includes the ESP32-CAM module mounted securely in its custom-designed casing. The setup is installed over the water meter.

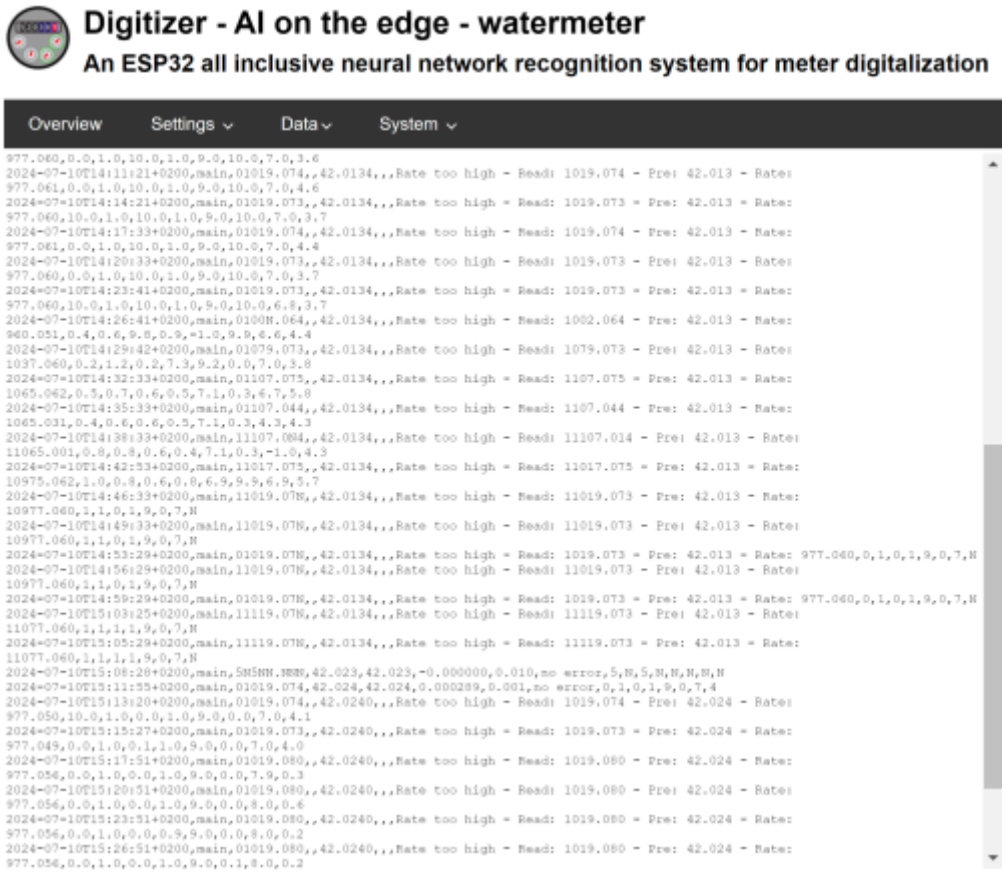

System Functionality

The system successfully captures images of the water meter, processes these images to detect and read the meter's numerical values using Computer vision, and displays the readings on a user-friendly interface. It accurately detects and highlights the numbers on the water meter, ensuring precise reading and data collection. The detected readings are displayed on the interface accessed via the IP address, providing real-time water consumption data for remote monitoring and management. All captured images and processed data are stored for further analysis.



Data Storage and Analysis

All captured data is stored in a structured format, allowing for historical analysis and tracking of water consumption over time. This data can be used to identify usage patterns, detect anomalies, and generate reports for efficient water management and billing.



5. Discussion

Accuracy and Reliability

The implementation of the Computer vision on the ESP32-CAM demonstrated high accuracy in reading the numerical values from the water meter. The system's ability to correctly detect and interpret the digits is critical for reliable water usage monitoring. However, occasional inaccuracies were observed due to variations in lighting conditions and reflections from the water meter cover. But this can be fixed by the methods mentioned earlier in “Dealing with Reflections”.

Integration with Home Assistant

Integrating the system with Home Assistant adds significant value by enabling easy monitoring and management of water consumption. Connecting the ESP32-CAM to Home Assistant is straightforward; in the settings under “MQTT,” entering the Home Assistant address allows for automatic detection and display of water consumption data in cubic meters (m^3). This seamless integration makes it easier for users to incorporate water usage monitoring into their existing smart home setups.

Challenges Encountered

Hardware Issues: One of the primary challenges was getting the ESP32-CAM module to enter flashing mode. Despite trying various methods and configurations, only one out of four ESP32-CAM

modules successfully entered flashing mode. This inconsistency suggests that it's crucial to verify the proper functioning of the ESP32-CAM modules before deployment. Ensuring hardware reliability can save significant time and effort during the setup phase.

Camera Setup: Setting up the camera to achieve the correct focus and alignment was another challenge. The process required careful adjustments to the focal length of the OV2640 camera and dealing with reflections on the water meter's protective cover. Removing the glue on the lens to adjust the focus was delicate and needed precision to avoid damaging the camera.

Alignment References: The initial setup of Alignment References posed significant issues. Every time the ESP32-CAM was powered up, the settings were reset, causing the Regions of Interest (ROIs) to shift and making it difficult to accurately capture the numbers. To address this, the Alignment References were turned off. The custom casing ensured that the ESP32-CAM and water meter maintained a consistent position, eliminating the need for re-alignment with each restart.

6. Conclusion

I managed to make a working module for a water meter that is ready for use in household. The ESP32-CAM for smart water metering has shown great potential in automating water usage monitoring. The system accurately reads water meter numbers, though there were some challenges with lighting and reflections. Integrating it with Home Assistant made the system easier to use and provided real-time data for better water management.

A major advantage of this project is its low cost. The ESP32-CAM and other components are affordable, making it accessible to many people. Also, the setup process, while requiring some basic technical skills, is not too complicated. Clear instructions and open-source firmware make it easier to implement.

Links

https://jomjol.github.io/AI-on-the-edge-device-docs/