AI-powered Intrusion Detection System

User Manual

Sathwik Kannam
Michel Jensen

AI-powered Intrusion Detection System

User Manual

Sathwik Kannam
Michel Jensen

Contents

1 Introduction **3**

2 System Overview **3**

2.1 Architecture 4

3 Hardware Requirements **5**

4 Software Requirements **5**

5 IR LEDs usage **6**

6 Installation **7**

7 Usage **10**

8 Troubleshooting **11**

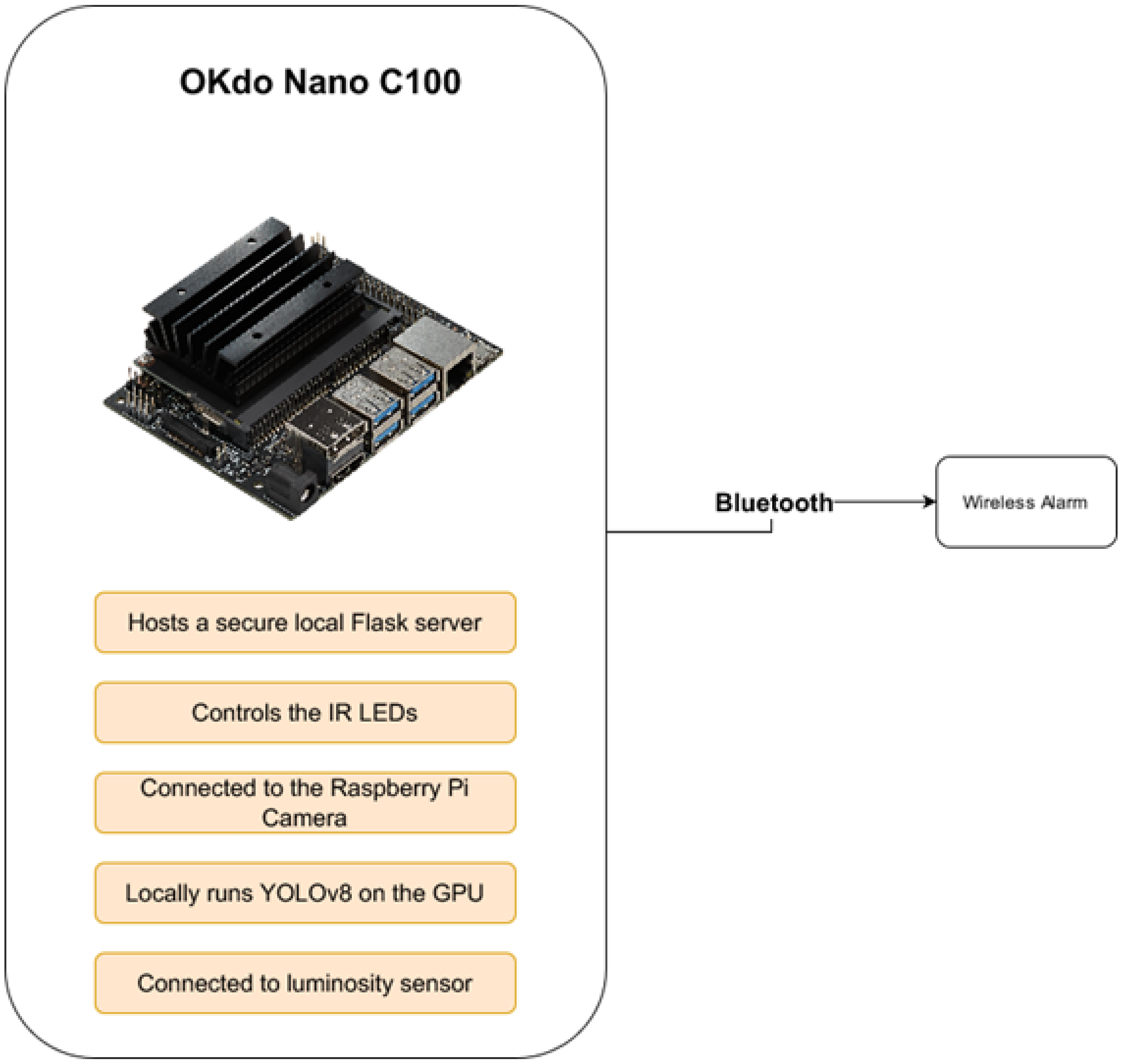
9 Support and Contact Information **12**

1 Introduction

This manual guides you through the operation of an embedded intrusion detection system designed to identify and report unauthorized personnel within a designated area. The system utilizes the power of YOLOv8, an advanced object detection model, to analyze live video feeds and effectively detect unauthorized individuals attempting to enter the protected zone.

Upon identifying an intrusion, the system takes immediate action. It triggers a remote alarm notification, alerting designated personnel of the potential security breach. Additionally, for enhanced situational awareness, the system enables remote inspection of the area. This functionality is achieved through a secure Flask server running locally on the network, allowing authorized users to view the live video feed and assess the situation in real time. This combination of features empowers proactive security measures and facilitates the effective monitoring of the protected area. The system can be in a dark environment due to the Infrared LEDs surrounding the camera.

2 System Overview


OKdo Nano C100
Hosts a secure local Flask server
Controls the IR LEDs
Connected to the Raspberry Pi Camera
Locally runs YOLOv8 on the GPU
Connected to luminosity sensor
Bluetooth
Wireless Alarm

This embedded security system is designed for real-time object detection and intrusion monitoring.

The system's core is the NVIDIA Jetson Nano Developer Kit [NVIDIA Jetson Nano Developer Kit], a compact single-board computer designed to run artificial intelligence applications on edge devices. The Jetson Nano's powerful GPU is ideally suited to execute the YOLOv8 object detection model locally, enabling fast and near real-time performance – a critical factor for security systems.

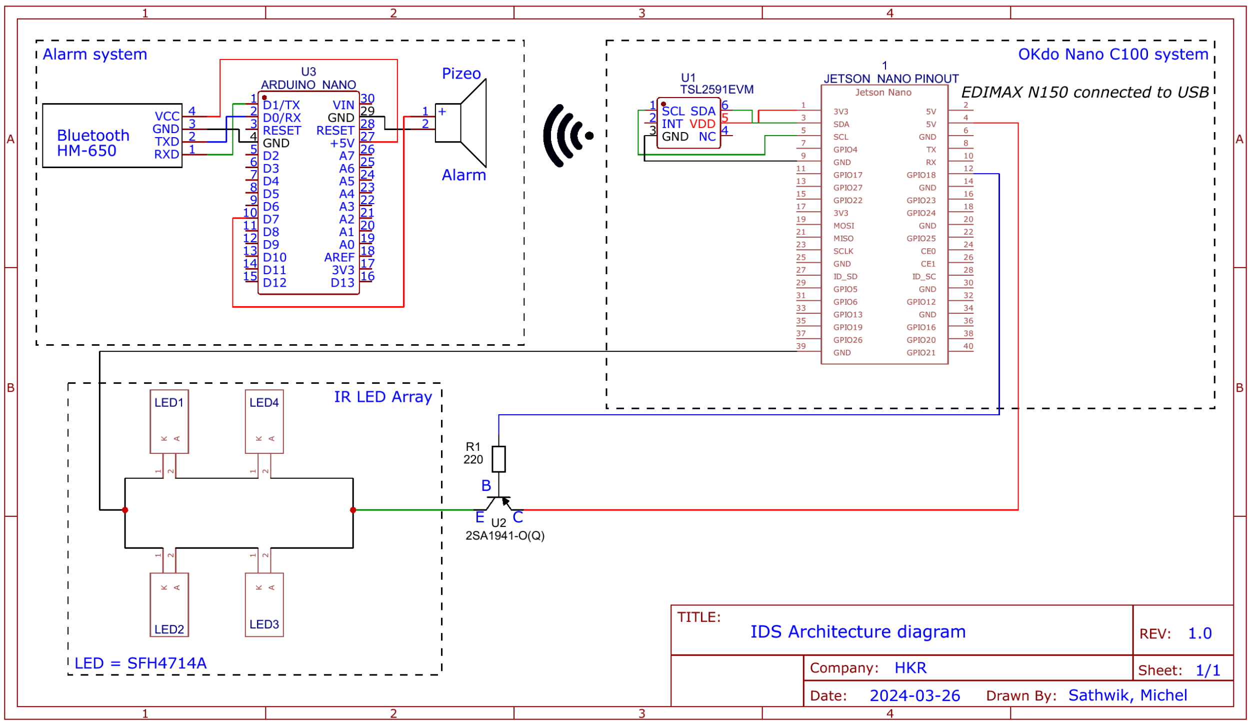
A Flask server is implemented for visualization, simulation, and manual control. This server creates a web interface accessible from a computer or mobile device. Users can view live camera footage, simulate security breaches for testing purposes, and remotely turn the security system on or off via the web interface.

The system leverages various sensors and actuators to achieve comprehensive security monitoring. A luminosity sensor detects ambient light levels. The sensor automatically triggers the infrared LEDs when complete darkness falls, ensuring continuous visual monitoring even in low-light conditions. This allows the security camera to capture clear video footage regardless of the lighting situation.

For alerting purposes, the system utilizes Bluetooth communication. When the YOLOv8 model detects an intrusion or human presence, a signal is transmitted locally via Bluetooth to a remote battery-powered alarm; therefore, no setup is required for the alarm. This triggers the alarm, notifying relevant authorities of a potential security breach. The Bluetooth communication allows for a quick and localized alert system.

This overview provides a high-level look at the core components and functionalities of the embedded security system. More detailed descriptions of each component and its operation will be covered in later sections of this manual.

2.1 Architecture



This section outlines the architecture of an intrusion detection system comprised of three primary components: an OKdo Nano c100 system, an alarm system, and the **IR LED array** (covered in Section 5).

I. **OKdo Nano C100 System**:

The OKdo Nano c100 system serves as the central processing unit for the intrusion detection system. It features two peripheral devices connected to the board:

- **Luminosity Sensor**: This sensor detects ambient light levels and communicates with the OKdo Nano c100 using the I2C communication protocol. The Serial Clock (SCL) and Serial Data (SDA) lines are connected to their respective pins on the board. Additionally, the sensor requires a 5V power supply obtained from the dedicated 5V pin on the board. The ground pin is also connected to ensure proper circuit operation.
- **EDIMAX N150**: The EDIMAX N150 enables both WiFi and Bluetooth support to the Jetson Nano. This allows us to host Flask servers and communicate with the alarm system using Bluetooth (BLE).

II. **Alarm System**:

The alarm system generates audible or visual alerts upon intrusion detection. It consists of three main components:

- **Arduino Uno ATMega328P**: This microcontroller serves as the processing unit for the alarm system.
- **Bluetooth Module - (HM-650)**: A Bluetooth module is connected to the Tx and Rx pins of the Arudino. To establish a null modem configuration, the Transmit (Tx) and Receive (Rx) pins of the Zigbee module are swapped. This module facilitates wireless communication with the central unit (Jetson)
- **Alarm**: The alarm device (piezo) is connected to digital Pin 7 of the Arduino Uno. digitalWrite() is employed to control the activation of the alarm output.

3 Hardware Requirements

I. OKdo Nano C100 Development kit [1]

II. SFH-4714A [2]

III. Adafruit TSL2591 High Dynamic Range Digital Light Sensor - STEMMA QT [3]

IV. Raspberry Pi Camera Module 2 NoIR [4]

V. GSM60E05-P1J (Power supply rated for 5V 6A) [5]

VI. 2SA1941-O(Q) (100W Transistor) [6]

VII. EDIMAX N150 [7]

VIII. Bluetooth 4.0 Module (HM-650)

4 Software Requirements

I. OKdo Nano C100 Developer Kit Official Software Image

II. **OpenCV** library for Python 3

III. **ultralytics** package for Python 3

IV. **Flask** web framework for Python 3

V. **Flask-Login** library for Python 3

VI. **csv** library for Python 3

VII. **secrets** library for Python 3

VIII. **OpenSSL** to create private keys and certificates.

IX. **bleak** for Bluetooth (BLE) communication

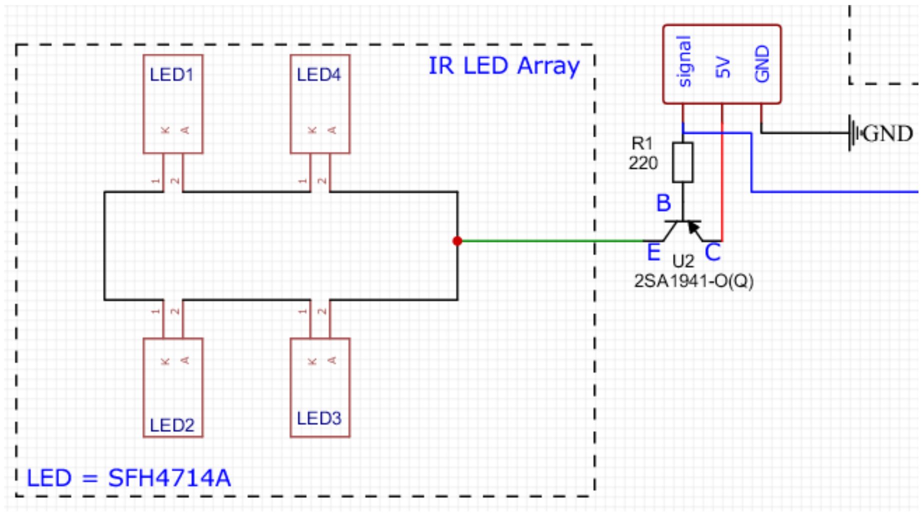
X. **python_tsl2591** for I2C communication with luminosity sensor.

XI. **pyzmq** for byte streams between Python 3.6.9 to Python 3.8.0

XII. **numpy** to convert byte streams to frame/images.

Check the documentation for a detailed overview of the software requirements, including versions.

5 IR LEDs usage



I. Power Supply and Connection

- Connect the base pin to the Pin D18 of the Jetson Nano and the Collector pin to the Ground pin of the Jetson Nano.

II. IR LED Array Configuration

- The four SFH-4714A IR LEDs (labeled LED1 to LED4) are connected in **parallel**.

III. Transistor Control

- A transistor (2SA1941-O(Q)) regulates current flow through the IR LED array and hence can be used to control the activation of the IR LEDs.
- Connect the transistor's pins as follows:
  - Base (B): This pin should be connected to the signal input, which can come from the Jetson (Pin D18). This signal input will be used to control how the IR LED array operates.
  - Emitter (E): Connected to the IR LED array.
  - Collector (C): Connected to 5V of the Jetson Nano.

IV. Safety Precautions

- Heat: IR (infrared) LEDs can generate heat during operation, mainly when used for extended periods. This heat generation is a normal byproduct of their functionality. To ensure optimal performance and longevity, it's crucial to avoid installing the system in environments with high ambient temperatures. Excessive heat can shorten the lifespan of IR LEDs and potentially impact their performance. Lastly, high humidity can trap heat around the IR LEDs, further exacerbating temperature rise. If possible, maintain a moderate humidity level in the operating environment.
- Current Limiting: Do not exceed the specified current limits for the LEDs.
- Polarity: Observe correct polarity during connections.

6 Installation

To use the software, you would need to create two Python environments with different versions. You need to install base Python 3.6.9 system-wide and create a Python environment with 3.8.0.

I. Connect the Wireless module.

- Connect the EDIMAX N150 to a USB port on the Jetson.
- Follow the guide in [10] for driver installation.
- Then connect to WiFi or use an Ethernet port.

II. Power supply.

- Connect GSM60E05-P1J power supply to the Jetson Nano.

III. Adafruit TSL2591

- Connect the sensor using the schematic provided in Section 2.1

IV. Setting up the Raspberry Pi Camera Module.

- Connect the Raspberry Pi Camera Module to the CSI port on the Okdo nano c100.

V. Setting up the Alarm system.

- Connect HM-650 to the alarm system. Connect Tx to Rx and Rx to Tx for null modem configuration.
- Connect 5V and GND to the Arduino.
- Flash the code provided in alarm/main.ino directory to Arduino.

VI. Python 3.6.9

- Install Python 3.6.9 system-wide
- Install and build OpenCV 4.5.0 from source in this system-wide Python version, follow the guide in [11].
- Install pyzmq, and numpy with this Python version.

VII. Setup Python 3.8.0 Environment

- Install Python 3.8.0

```
$ sudo apt install python3.8 python3-pip
```

- Setup the environment (Replace 'myenv' with a name of the environment)

```
$ python3.8 -m venv <myenv>
$ source <myenv>/bin/activate
```

VIII. Alternative: Conda environment

- Create an environment

```
$ conda create -n <myenv> python=3.8.0 ipython
$ conda activate <myenv>
```

IX. Installing Python Libraries (Assumed that you have a Python 3.8.0 environment activated):

- Open a terminal window on the OKdo Nano C100.
- Update the package lists:

```
$ sudo apt update
```

- Install the Python libraries using the requirements.txt. This approach ensures there are no compatibility issues.

```
$ pip install requirements.txt
```

X. PyTorch and torch-vision guide for GPU acceleration (run YOLOv8 on the GPU)

- Follow the guide in [8] to install it in the Python 3.8.0 Environment.

XI. Creating the User Account.

- Create a new user account for the embedded system application. Replace "username" and "password" with your desired credentials:

```
$ sudo adduser <username> \\
sudo passwd <username>
```

XII. Cloning the Application Code

- Clone the Git repository containing the application code (provided) to the home directory of the newly created user:

```
$ git clone https://github.com/eggheadtwins/YOLOTHESIS.git
```

XIII. Security Certificate:

- These steps require creating a security certificate for HTTPS communication with the Flask application.
- Generate a private key and a self-signed certificate using OpenSSL:

```
$ sudo openssl req -x509 -newkey rsa:4096 \\
-nodes -out security/flask_cert.pem -keyout security/flask_key.pem -days 365
```

- Place the generated certificate and private key files in a " security " directory within the application folder.

XIV. Running the Application

- Navigate to the application directory:

```
$ cd /home/<username>/YOLOTHESIS
```

- When you clone the Git repository, default credentials are already implemented. However, creating a custom account for added privacy is highly recommended. To do this, create a new file named security/user.txt and add the username and password on a single line separated by a comma (,). Ensure there are no spaces around the credentials:

  ```
  <username>,<password>
  ```

- Start the Flask application:

  ```
  $ python3 main.py
  ```

- Access the application web interface in a web browser using the OKdo Nano C100's IP address and port (port 5000 by default). You will need to log in with the credentials specified in the security/user.txt file.

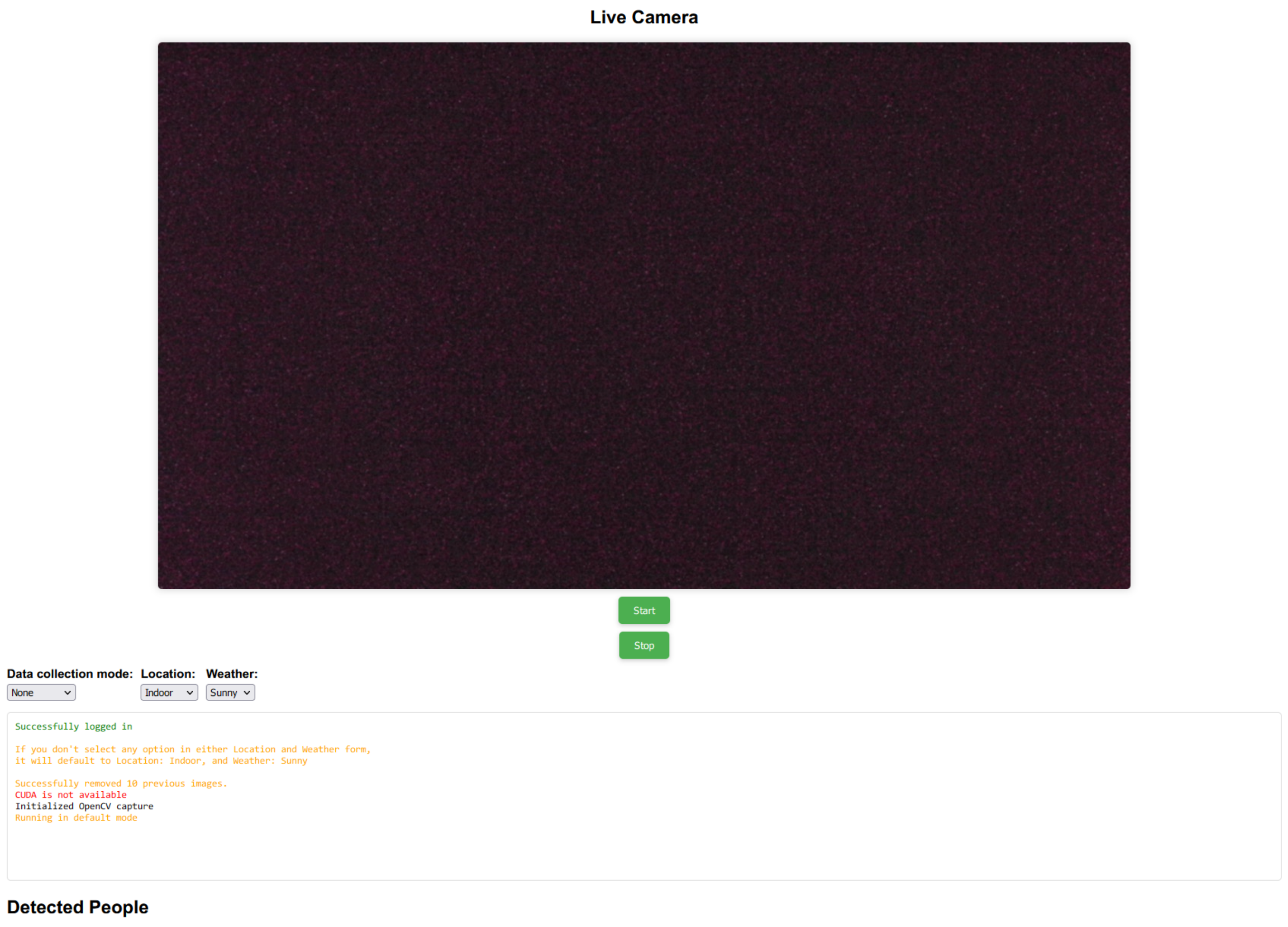
XV. System Placement

- To optimize the efficiency and effectiveness of the system, place the system where the camera covers a large area. Therefore, it is recommended to be placed on a ceiling targeting the entry points of the area (e.g., doors).
- Ensure covering the luminosity sensor to avoid invalid sensory values. This will cause errors when reading reliable luminance data.

XVI. Terminating the csicam.py process

- You need to manually terminate the csicam.py using the Terminal to kill the socket to rerun the subprocess.

7 Usage



Data collection mode: None
Location: Indoor
Weather: Sunny

```
Successfully logged in

If you don't select any option in either Location and Weather form,
it will default to Location: Indoor, and Weather: Sunny

Successfully removed 10 previous images.
CUDA is not available
Initialized OpenCV capture
Running in default mode
```

I. Web Interface

- The primary method of interaction with the embedded system is through a web interface accessible on a web browser. Open a web browser on any device connected to the same network as the OKdo Nano C100 and navigate to the following URL: https://ipadress:5000.

II. Authentication

- Upon accessing the web interface, you will be prompted to log in using the credentials specified during application setup (refer to the Installation Steps - Creating the User Account section).

III. Main Dashboard

- Live camera feed: This section displays a real-time video stream captured by the Raspberry Pi Camera Module.
- Object detection results: Overlaid on the live camera feed or displayed separately, this section may showcase detected objects and their classifications based on the implemented algorithms (e.g., person detection, animal detection).
- Logging system: Under the live camera feed, there is a log system that provides warnings, errors, or any successful operations, allowing you to remotely debug the system without having to disconnect the system.

- Ultimately, when you press the stop button, images of detected people are presented under the logs, which gives you an overview of the detections found throughout the runtime.

IV. Data collection

- The main dashboard has options for data collection modes: Luminance, Weather, and Indoor.
- When set to Luminance, the system will measure the luminance whenever a person is detected and store the luminance and confidence at detecting the person in the CSV file.
- When set to Weather mode, you must set the weather parameter in the weather box based on the weather outside. When clicked START, the system will run for 10 minutes and calculate the average confidence of all human detections. Lastly, the selected weather parameter and the average confidence is stored a CSV file.
- Similarly, when set to Location mode, you must set the location parameter to whether the physical system is indoors or outdoors, and the rest follows the same weather mode.
- Lastly, the clear box refers to whether to empty the CSV files before starting a new session.

V. CSV files downloads

- Whenever you click stop and use the data collection modes, you will get an option below the logging system; when clicked, it will automatically download the CSV file to your machine.
- WARNING: There is an issue with downloading CSV when connecting to the localhost on another device.

8 Troubleshooting

I. Cannot access the web interface

- Verify that the Okdo nano c100 is powered on and connected to the network.
- Double-check the web interface URL and ensure you are using the correct IP address for your device.
- If using a firewall, ensure it is configured to allow traffic on port 5000 (the default port for the Flask application).

II. Web interface loads slowly:

- High network traffic or limited processing power on the Okdo nano c100 can cause slow loading times.

III. OKdo Nano C100 degradation

- The OKdo Nano C100 is an efficient microcontroller. However, extensive usage may cause it to be a thermal throttle due to the GPU-accelerated software. To mitigate this, consider purchasing a new fan for the heatsink and replacing the thermal paste of the GPU and CPU.

IV. Login issues:

- Ensure you are entering the username and password correctly as specified during setup.
- Check for typos or case sensitivity issues.

V. No video stream.

- Verify that the Raspberry Pi Camera Module is properly connected to the CSI port on the Okdo nano c100.
- Check if the camera module is enabled in the application software. Refer to the application documentation (In GitHub) for specific instructions.

VI. Unresponsive IR LEDs

- Likely, the power supply to the IR LEDs is slightly loose, causing it to disconnect. Ensure it is properly connected and isolated.
- If the LEDs are properly connected and still unresponsive, they are likely degraded due to extensive use and heat dissipation. Therefore, consider replacing the LEDs.

VII. Not running on the GPU

- If you get an error message in the logging system in the Flask server stating "CUDA not available," then YOLOv8 is not running on the Jetson's GPU.
- Make sure you have the correct version of PyTorch as NVIDIA Jetpack 4.6, and PyTorch works only with Python 3.6. However, YOLOv8 can't work with Python under 3.8.
- Therefore, compile a compatible PyTorch version necessary for running YOLOv8 on the GPU (instructions provided in [8]) or refer to the official PyTorch guide on how to do this.
- Else, create an issue in Ultralytics GitHub [9] or contact NVIDIA for possible solutions.

VIII. Alarm not ringing.

- The Bluetooth connection between the Jetson and HM-650 may have disconnected; therefore, try bringing the Bluetooth module closer to the Jetson.

IX. No camera preview

- Likely, the socket in csicam.py is still running. Therefore, try killing the job/process in the terminal.

X. General Troubleshooting Tips

- Reboot the system: A simple restart often resolves temporary glitches or software errors.

9 Support and Contact Information

I. Sathwik Kannam: sathwik.kannam0191@stud.hkr.se

II. Michel Jensen: michel.jensen0008@stud.hkr.se

You can also refer to the documentation.

References

[1] OKdo, *OKdo Nano C100 Developer Kit powered by NVIDIA® Jetson Nano Module*, Available: https://www.okdo.com/us/p/okdo-nano-c100-developer-kit-powered-by-nvidia-jetson-nano-module/, OKdo, 2024

[2] Mouser, *SFH-4714A Product Detail*, Available: https://www.mouser.se/ProductDetail/ams-OSRAM/SFH-4714A, Mouser, 2024

[3] Adafruit, *Adafruit TSL2591 High Dynamic Range Digital Light Sensor - STEMMA QT*, Available: https://www.adafruit.com/product/1980, Adafruit, 2023

[4] Raspberry Pi, *SC0875*, Available: https://www.raspberrypi.com/products/pi-noir-camera-v2/, Raspberry Pi, 2024

[5] Mouser, *GSM60E05-P1J*, Available: https://www.mouser.se/ProductDetail/709-GSM60E05-P1J, Mouser, 2024

[6] Mouser, *2SA1941-O(Q)*, Available: https://www.mouser.se/ProductDetail/Toshiba/2SA1941-OQ?qs=s6nfgz3miVwN9gIQs%2By%2BCw%3D%3D, Mouser, 2024

[7] Edimax, *2-in-1 N150 Wi-Fi Bluetooth 4.0 Nano USB Adapter*, Available: https://www.edimax.com/edimax/merchandise/merchandise_detail/data/edimax/global/wireless_adapters_n150/ew-7611ulb/, Edimax, 2024

[8] EdwardoSunny, *Jetson-Nano-YOLOv8-Setup* , Available: https://github.com/EdwardoSunny/Jetson-Nano-YOLOv8-Setup, GitHub, 2024

[9] ultralytics, *ultralytics*, Available: https://github.com/ultralytics/ultralytics, GitHub, 2024

[10] eleccelerator, *Jetson Nano Edimax EW-7611ULB*, Available: https://eleccelerator.com/wiki/index.php?title=Jetson_Nano_Edimax_EW-7611ULB, eleccelerator, 2019

[11] AastaNV, *JEP*, Available: https://github.com/AastaNV/JEP/blob/master/script/install_opencv4.5.0_Jetson.sh, GitHub, 2019