**USER MANUAL**

Table of Contents

1. Introduction

2. System Overview

3. Hardware Setup

4. Software Installation

5. System Configuration

6. Operating Instructions

7. Maintenance and Troubleshooting

8. Safety Guidelines

9. Frequently Asked Questions (FAQ)

1. Introduction

Welcome to the Intelligent Sliding Door System. This advanced system uses AI-powered object detection to automate door operations, enhancing safety, efficiency, and user experience in various environments such as academic institutions, hospitals, commercial buildings, and airports.

2. System Overview

The Intelligent Sliding Door System consists of:

* Raspberry Pi (with Picamera2 support)
* USB Camera (640x480 resolution, RGB888 format)
* LED indicators:
* Green LED: Door open
* Red LED: Door closed
* Yellow LED: Detection in progress
* Blue LED: System standby
* Buzzer: Audio feedback for door actions
* YOLO object detection model

3. Hardware Setup

Connect the Picamera2 to the Raspberry Pi.

Set up the LED indicators using the GPIO pins:

* Green LED: GPIO 17
* Red LED: GPIO 27
* Yellow LED: GPIO 22
* Blue LED: GPIO 23

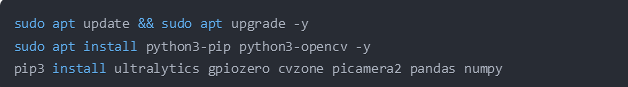
Connect the buzzer to GPIO 24.

Ensure all ground connections are properly made.

Position the camera to capture the approach area in front of the sliding door.

4. Software Installation

* Install Raspberry Pi OS on your Raspberry Pi.
* Open a terminal and run the following commands:



* Clone the project repository and navigate to the project directory.
* Download the pre-trained YOLO model file 'best.pt' and place it in the project directory.
* Ensure you have the 'coco2.txt' file in the project directory, containing the class list.

5. System Configuration

The system is pre-configured with the following settings:

* Camera resolution: 640x480
* Frame processing rate: Every 5th frame
* Door open duration: 5 seconds

To modify these settings, edit the Python script directly.

6. Operating Instructions

Data Collection and Annotation

For optimal performance of the Intelligent Sliding Door System, it's important to train the YOLO model with a diverse and well-annotated dataset. This section outlines the process for collecting and annotating data to improve or retrain the model.

Data Collection

1. Capture a diverse set of images and videos of people approaching the sliding door:
   * Include various lighting conditions (day, night, artificial lighting)
   * Capture different approach angles (front, side, diagonal)
   * Ensure diversity in individuals (height, body type, clothing, etc.)
   * Include scenarios with multiple people
   * Capture both positive cases (people intending to use the door) and negative cases (people passing by)
2. Aim for a dataset of at least 850 images to ensure good model performance.
3. Save the images in a common format (e.g., JPEG) and organize them in a dedicated folder.

Data Annotation

1. Use annotation tools such as CVAT (Computer Vision Annotation Tool) or Roboflow for labelling the collected images.
2. Label the following classes:
   * 'front': For individuals approaching the door directly from the front
   * 'side': For individuals approaching from the side or passing by
3. Draw bounding boxes around each person in the images.
4. Ensure consistent labelling across all images.
5. Export the annotations in both COCO and YOLO formats for compatibility.

Data Preparation

1. Resize all images to 640x480 pixels to match the camera input size.
2. Apply data augmentation techniques to increase dataset variability:
   * Rotate images
   * Adjust brightness and contrast
   * Apply slight blur or noise
3. Split the dataset into training, validation, and testing sets (e.g., 70% training, 15% validation, 15% testing).

Model Training

1. Use Google Colab or Kaggle Notebooks for training if local GPU resources are limited.
2. Upload your prepared dataset and annotations to the chosen platform.
3. Use the following hyperparameters as a starting point:
   * Learning rate: 0.001
   * Batch size: 16
   * Epochs: 20 to 150 (monitor validation performance to determine optimal number)
4. Train the YOLOv8n model using the prepared dataset.
5. Evaluate the model performance using precision, recall, and F1 score metrics.
6. Fine-tune the model, if necessary, by adjusting hyperparameters or collecting more data for challenging scenarios.

Model Deployment

1. Once satisfied with the model's performance, export it in the appropriate format (e.g., 'best.pt').
2. Replace the existing 'best.pt' file in your Intelligent Sliding Door System's project directory with the new model file.
3. Update the 'coco2.txt' file if any changes were made to the class labels.
4. Restart the system and monitor its performance with the new model.

By following these steps for data collection, annotation, and model training, you can continuously improve the accuracy and reliability of your Intelligent Sliding Door System.

To start the system:

Open the directory where the project is saved and launch the ‘main.py’ file using the Thonny application for Python on the Raspberry Pi 5. Click the START button to execute.

The system will initialize, and the Blue LED will light up to indicate standby mode. When a person approaches:

* Yellow LED: Lights up during detection
* Green LED: Indicates the door is open
* Red LED: Indicates the door is closed
* Buzzer: Sounds when the door opens

The system automatically detects individuals and their approach direction:

* Front Approach: The door will open automatically.
* Side Approach: The door will remain closed.

To stop the system, press 'q' in the camera window or use Ctrl+C in the terminal. The Thonny application for Python on the Raspberry Pi 5 also has a STOP button associated.

7. Maintenance and Troubleshooting

* Regularly clean the camera lens for optimal performance.
* Check all GPIO connections periodically to ensure they're secure.
* Update the software regularly by pulling the latest changes from the repository.
* If the system fails to start, check for error messages in the terminal.
* Ensure the 'best.pt' model file and 'coco2.txt' are present in the project directory.

8. Safety Guidelines

* Ensure the area around the door is clear of obstructions.
* Do not modify the hardware setup while the system is running.
* In case of emergency, use the manual override to open the door.
* Regularly test the system's detection capabilities and LED indicators.
* Do not expose the system to water or moisture.

9. Frequently Asked Questions (FAQ)

Q: The system isn't detecting people accurately.

A: Ensure proper lighting and camera positioning. You may need to retrain the YOLO model with more diverse data.

Q: How can I adjust the buzzer open duration?

A: Modify the `time.sleep(5)` value in the `control\_door` function of the Python script.

Q: Can the system differentiate between multiple people?

A: Yes, the system processes each detected person individually.

Q: How can I improve performance?

A: Consider increasing the processing rate by adjusting the frame skip count in the main loop.

Notes

The system is designed to open the door when a person is detected approaching from the front and close it when no person is detected, side view detected or when a person is detected moving away. Ensure the Raspberry Pi is properly configured for camera support (enable camera interface in raspi-config). Modify GPIO pin assignments in the code if using different pin configurations.

For further assistance, please contact our technical support team.