Face Recognition Smart Door Lock System Using Raspberry Pi

Table of Contents

- 1.1 Overview
- 1.2 Purpose
- 1.3 Hardware Components
- 1.4 Software Components
- 1.5 Prerequisites

System Architecture and Flow

- 2.1 System Overview
- 2.2 Flowchart of the System Operation
- 2.3 Key Features
- 2.4 Demonstration Overview

Hardware Setup

- 3.1 Connecting LCD Model to Raspberry Pi
- 3.2 Connecting DC Motor for Smart Door Lock
- 3.3 Motor Driver IC Connections
- 3.4 Connecting LED and PR Sensor
- 3.5 Buzzer, Camera, and Fingerprint Sensor Connections
- 3.6 Switch Pin Connections

Circuit Diagrams

- 4.1 LCD Connection Diagram
- 4.2 Motor Driver IC Connection Diagram

- 4.3 Buzzer, Camera, and Fingerprint Sensor Connection Diagram
- 4.4 Switch Pin Connection Diagram

❖ Programming Setup

- 5.1 Installing Necessary Libraries (cv2, face recognition)
- 5.2 Downloading Code from Google Drive
- 5.3 Creating a New Folder on Raspberry Pi
- 5.4 Configuring Switch Pin and Other Components
- 5.5 Setting Up Image Storage and Comparison
- 5.6 Initializing Face Recognition System

❖ Code Explanation

- 6.1 Defining Input and Output Pins
- 6.2 Accessing Camera using cv2.VideoCapture
- 6.3 Face Recognition Code (rahul_image, travel_phase_encoding, brow_face_encoding)
- 6.4 Using Known Faces Array and Passing User Names
- 6.5 Utilizing LCD Functions for Face Detection Messages

❖ Testing and Debugging

- 7.1 Testing Each Component

- 7.2 Debugging Common Issues
- 7.3 Troubleshooting Tips

❖ Demonstration

- 8.1 Running the Complete Project
- 8.2 Testing Face Recognition
- 8.3 Sending Email with Image on Door Opening
- 8.4 Verifying Presence Using PR Sensor
- 8.5 Demonstrating Unknown Visitor Scenario

❖ Conclusion and Future Enhancements

- 9.1 Project Summary
- 9.2 Lessons Learned
- 9.3 Possible Improvements and Additions

❖ References

- List of sources used for the project
  Appendix
- Additional information, diagrams, or code snippets

Introduction

Using a strong facial recognition algorithm, the Face Recognition Smart Door Lock algorithm with Raspberry Pi seeks to improve home security. Authorised professionals can use this method to inform homes about unexpected visitors. The application combines the Raspberry Pi with a number of hardware elements, including buzzers, LCDs, DC motors, cameras, LEDs, and PR sensors.

Purpose

The Raspberry Pi can be connected to the hardware component using facial recognition to create a smart door lock system. Use facial recognition algorithms to control access. Notify the homeowner by email or SMS if the camera identifies unfamiliar guests. Add a PR sensor to detect visitors.

1.3 Hardware components

- The Raspberry Pi
- LCD display
- DC motor
- Camera (USB).
- led
- Public Interface Sensor
- motor carrier IC

- fingerprint sensor
- Change images

1.4 Software Products

- Python (for Raspberry Pi system).
- OpenCV (for image processing).
- Front introduction library
- cv2 (Python library for computer monitoring).

1.5 Prerequisites

- Basic knowledge of Raspberry Pi and Python programming.
- Obtaining required hardware parts.
- Installation of necessary Python libraries.

2. System Architecture and Flow

2.1 Content of the Plan

The system creates facial recognition smart doors by integrating hardware with a Raspberry Pi. The process entails putting a message on an LCD screen, scanning visitors' faces with a camera, and making accessibility recommendations based on face recognition.

2.2 Flowchart of the System

A detailed flow chart showing the sequence of events from guest arrival to door entry or denial.

2.3 Key Features

- Face recognition for authorized access.
- Real-time image capture and notification for unknown visitors.
- PR sensor for detecting home occupancy.
- Integration with email for visitor image sharing.

3. Hardware Setup

3.1 Connecting LCD Model to Raspberry Pi

Step 1: Gather Components

- Raspberry Pi
- LCD Display
- Jumper wires

Step 2: Identify Pins

- Examine how the LCD and Raspberry Pi are configured with regard to pins.
- Attach the LCD's VCC and GND pins to the Raspberry Pi's matching 5V and GND pins.

Step 3: Connect Data Pins

- Connect the data pins of the LCD to the GPIO pins on the Raspberry Pi.
- Refer to the datasheets or documentation for your specific LCD model for pin details.

Step 4: Power Up

- Power up the Raspberry Pi and ensure that the LCD is receiving power.
- Verify the connection by displaying a sample message on the LCD using a simple Python script.

3.2 Connecting DC Motor for Smart Door Lock

Step 1: Gather Components

- Raspberry Pi
- DC Motor
- Motor Driver IC
- Jumper wires

Step 2: Motor Driver Connections

- Connect the input pins of the motor driver IC to GPIO pins on the Raspberry Pi.
- Connect the output pins of the motor driver IC to the DC motor.

Step 3: Power Supply

- Connect the power supply to the motor driver IC, ensuring it matches the motor specifications.
- Connect the common ground of the Raspberry Pi to the common ground of the motor driver IC.

Step 4: Test Motor

- Write a simple Python script to test the motor by running it in both directions.
- Ensure the motor responds appropriately to the script.

3.3 Motor Driver IC Connections

Step 1: Understanding Motor Driver IC

- Identify the input and output pins of the motor driver IC.
- Connect the input pins to GPIO pins on the Raspberry Pi for control.

Step 2: Wiring to the Motor

- Connect the output pins of the motor driver IC to the DC motor.
- Ensure that the power supply to the motor driver IC is appropriate for the motor's voltage and current requirements.

Step 3: Connect to Raspberry Pi

- Establish connections between the motor driver IC and the Raspberry Pi using jumper wires.
- Follow the provided circuit diagram to ensure accurate connections.

3.4 Connecting LED and PR Sensor

Step 1: LED Connection

- Connect the LED's longer leg (positive) to a GPIO pin on the Raspberry Pi.
- Connect the shorter leg (negative) to the common ground of the Raspberry Pi.

Step 2: PR Sensor Connection

- Connect the PR sensor's output pin to a GPIO pin on the Raspberry Pi.
- Connect the power (VCC) and ground (GND) pins of the PR sensor to the corresponding pins on the Raspberry Pi.

Step 3: Verify Connections

- Run a simple Python script to toggle the LED on and off based on the input from the PR sensor.
- Confirm that the PR sensor is detecting motion by observing changes in its output.

3.5 Buzzer, Camera, and Fingerprint Sensor Connections

Step 1: Buzzer Connection

- Connect the positive (longer) leg of the buzzer to a GPIO pin on the Raspberry Pi.
- Connect the negative (shorter) leg to the common ground of the Raspberry Pi.

Step 2: Camera Connection

- Connect the USB camera to any available USB port on the Raspberry Pi.

Step 3: Fingerprint Sensor Connection

- Connect the VCC, GND, and output pins of the fingerprint sensor to the corresponding pins on the Raspberry Pi.

Step 4: Verify and Test

- Test the buzzer by running a Python script to produce a sound.
- Verify the camera functionality by capturing a sample image.
- Ensure the fingerprint sensor is recognized by checking its output using a Python script.

3.6 Switch Pin Connections

Step 1: Identify Switch Pins

- Understand the pin configuration of the switch model.
- Identify the power, ground, input, and output pins.

Step 2: Wiring the Switch

- Connect the power and ground pins of the switch to the corresponding pins on the Raspberry Pi.
- Connect the input and output pins based on the switch's functionality.

Step 3: Switch Model Configuration

- If necessary, configure the switch model as described in the video, such as shorting two pins on the Zero PCB board.

Step 4: Test Switch

- Write a Python script to test the switch's functionality.
- Ensure that the switch input is recognized by the Raspberry Pi.

4. Circuit Diagrams

4.1 LCD Connection Diagram

#diagram

4.2 Motor Driver IC Connection Diagram

#diagram

4.3 Buzzer, Camera, and Fingerprint Sensor Connection Diagram

#diagram

4.4 Switch Pin Connection Diagram

#diagram

5. Programming Setup

6. Code Explanation

7. Testing and Debugging

7.1 Testing Each Component

Guidelines for testing each hardware component individually to ensure proper functionality.

7.2 Debugging Common Issues

Common issues that may arise during hardware setup and programming, with solutions.

7.3 Troubleshooting Tips

Additional troubleshooting tips for addressing potential challenges.

8. Demonstration

8.1 Overview

The demonstration phase is crucial to showcase the successful integration of hardware components and the flawless execution of the face recognition system using a Raspberry Pi. Ensure you follow these steps to effectively present your project:

8.2 Pre-Demonstration Setup

8.2.1 Physical Setup:

- Confirm that all hardware components are correctly connected to the Raspberry Pi based on the circuit diagrams and pin configurations.
- Power up the Raspberry Pi and ensure all peripherals (camera, fingerprint sensor, etc.) are operational.

8.2.2 Software Setup:

- Load the necessary libraries and dependencies onto the Raspberry Pi.
- Verify that the Python scripts for each component have been successfully uploaded.

8.3 Demonstration Steps

8.3.1 LCD Display:

- Run a script to display a welcoming message on the LCD.
- Emphasize the user-friendly interface of the smart door lock system.

8.3.2 Face Recognition and Door Lock:

- Initiate the face recognition script.
- Demonstrate the door opening when a recognized face is detected.
- Show the LCD display updating to indicate the recognition status.

8.3.3 Additional Functionality:

- Showcase additional features such as the buzzer ringing for unrecognized faces.
- Emphasize the integration of sensors like the PR sensor for presence detection.

8.3.4 Error Handling:

- Simulate scenarios where an unauthorized face is detected or the system encounters an error.
- Explain the error messages on the LCD and demonstrate how the system responds.

8.4 Post-Demonstration Steps

8.4.1 Q&A Session:

- Allow for questions from the audience.

- Provide explanations for any technical aspects or challenges faced during the demonstration.

8.4.2 Feedback Collection:

- Encourage feedback from the audience on the functionality, usability, and any potential improvements.

8.4.3 Future Enhancements:

- Discuss potential enhancements or future features that could be added to the smart door lock system.

9. Conclusion and Future Enhancements

9.1 Project Summary

The face recognition system using a Raspberry Pi for a smart door lock system has successfully demonstrated the integration of hardware components and software modules. The project involved connecting an LCD, DC motor, camera, buzzer, LED, and PR sensor to the Raspberry Pi, creating a functional system that recognizes authorized individuals and grants them access. The key achievements of the project include:

- Implementation of a reliable face recognition algorithm for user authentication.

- Integration of multiple hardware components, including the LCD, DC motor, camera, buzzer, and sensors, to create a cohesive smart door lock system.
- Real-time communication of recognized faces to the user through images sent to their mobile device.
- Effective use of the PR sensor for detecting the presence of individuals at home.
- Provision of a user-friendly interface through the LCD display for communication with visitors.

9.2 Lessons Learned

Throughout the development of this project, several valuable lessons were learned:

- Hardware Integration Challenges: Integrating diverse hardware components required careful consideration of pin configurations, power supplies, and communication protocols. Thorough testing and debugging were essential to ensure proper functionality.
- Software Complexity: Writing code to interface with the camera, perform face recognition, and control the various peripherals demanded a good understanding of Python, Raspberry Pi GPIO, and relevant libraries such as OpenCV and face_recognition.
- User Interface Design: Designing an intuitive and informative user interface on the LCD display involved considerations for effective communication of system status and instructions.
- Error Handling: Implementing robust error handling mechanisms, especially in face recognition scenarios, proved crucial for system reliability.

9.3 Possible Improvements and Additions

While the current system is functional, there are several avenues for future improvements and additions:

- Enhanced Security Features: Implement more advanced face recognition algorithms or consider incorporating additional security measures for a more robust access control system.
- Remote Access and Control: Integrate IoT capabilities to enable remote monitoring and control of the smart door lock system through a mobile app or web interface.
- Machine Learning Integration: Explore the use of machine learning to continually improve face recognition accuracy over time.
- Voice Commands: Implement voice recognition for hands-free control, allowing users to interact with the system verbally.
- Customization Options: Provide users with the ability to customize system settings, such as recognition sensitivity, notification preferences, and display themes.
- Integration with Smart Home Systems: Connect the smart door lock system to popular smart home platforms, allowing for seamless integration with other smart devices.

10. References

Geitgey, A. (2018). "Face Recognition with Python." GitHub. Link

OpenCV Documentation. (2023). "OpenCV Python Tutorials." Link

Raspberry Pi Foundation. (2023). "GPIO Pinout." Link

Fritzing. (2023). "Fritzing Software." Link

Youtube Video: https://www.youtube.com/watch?v=CDwTqcmPzxQ