İstanbul Bilgi Üniversitesi

Smart Door Lock with Face Recognition

CMPE 331 Software Engineering Concepts
Final Report

Omar Al-Shareef 119200019

Hamza Sallam 120200013

Yaman Arar 120200096

Mhd Nour Khalifa 120200127

Mohamed Hamed 120200155

Almustafa Alshekfeh 119202008

December, 2022

Contents

Chapter 1

Introduction

The smart door lock project is a system designed to securely control access to a physical space using facial recognition technology and a Raspberry Pi. The system utilizes the Python programming language and the opencv library to process video feed from a camera and compare images to a database of authorized individuals. If a match is found, the door is unlocked and access is granted to the user. This system provides both convenience and security by eliminating the need for traditional keys or access codes and using advanced facial recognition technology to verify the user's identity.

The Raspberry Pi serves as the central processing unit for the system, handling the tasks of capturing and analyzing the video feed from the camera. The opencv library is used to perform the facial recognition, which involves extracting and comparing unique facial features of the user with those stored in the database. This technology is highly accurate and can reliably identify individuals, even in challenging lighting conditions or when the user is wearing glasses or a face mask. The smart door lock project represents a significant advancement in the field of physical access control, offering a convenient and secure alternative to traditional methods.

Chapter 2

Project Details

2.1 Motivation

With this system, users would be able to securely unlock their doors simply by looking at the lock, without having to remember and enter a complicated combination or carry around a physical key. This could make it easier for users to access their homes and businesses, while also providing an additional layer of security to prevent unauthorized access..

Another motivation for this project could be the potential for it to have a positive impact on the environment. By eliminating the need for physical keys, a face recognition door lock could help reduce waste and the use of resources in the manufacturing and distribution of these items. Additionally, the use of advanced technology in the lock could make it more durable and long-lasting, further reducing the need for replacement keys and locks over time.

By implementing this technology in smart door lock systems,the aim is to provide a reliable and innovative solution for controlling access to a physical space. Additionally, the use of a Raspberry Pi allows for a cost-effective and customizable approach to access control, making it accessible to a wide range of users.

2.2 Aim and Objectives

The smart door lock project was designed to provide secure and convenient access control to a physical space using facial recognition technology. The system includes a user-friendly interface for managing authorized users, an opencv-based facial recognition algorithm, and optimizations for efficient video processing and recognition. The project aims to assess the real-world effectiveness of the system and make any required enhancements. These goals were pursued in order to create a reliable and innovative access control solution.

One of the primary objectives of the smart door lock project was to improve security by using advanced facial recognition technology. This technology involves extracting and comparing unique facial features to those stored in a database, allowing the system to accurately identify authorized users even in challenging lighting conditions or when the user is wearing glasses or a face mask. In addition to providing increased security, the use of facial recognition also eliminates the need for traditional keys or access codes, making the system more convenient for users. The integration of these features into the smart door lock system represents a significant advancement in the field of physical access control.

Aims and Objectives:
1-To develop a secure and convenient system for controlling access to a physical space using facial recognition technology.

2-To create a user-friendly interface for enrolling new users and managing the database of authorized individuals.

3-To implement a reliable and accurate facial recognition algorithm using the opencv library.

4-To optimize the performance of the system for fast and efficient processing of the video feed and facial recognition tasks.

5- To evaluate the effectiveness of the system in a real-world setting and make any necessary improvements or modifications.

2.3 Group Members and Tasks

Table 2.1: Group Members and Their Tasks

Mhd Nour Khalifa	CMPE	He worked on the hardware and prototype design of the project. He carried out the hardware part of the project. In particular, he made the connection of the Raspberry pi
Yaman Arar	CMPE	He worked on the hardware part of the project. In particular, he carried out the connection of the Solenoid Lock to the circuit.
Hamza Sallam	CMPE	He worked on the software part of the project. In particular, he wrote the codes for the Facial Recognition system and for the lock.
Omar Alshareef	CMPE	he managed the tasks given to the members. he worked with Hamza Sallam on the software part. He prepared the outline of the all written reports and distributed them to the group members, and then finalized the report.
Mohamed Hamed	CMPE	he worked on the software part of the project. he installed the necessary OpenCV modules and Raspberry pi interface
Almustafa Alshekfeh	EEEN	He worked on the hardware part of the project he established the circuit diagram and made the necessary connection of hardware to Raspberry Pi

2.4 Technologies

Every Project needs technologies to function properly. In the Technologies part, there will be a brief introduce some of the Technologies that our Project is containing.

- Software design was associated with using Raspberry Pi hardware, which is the most important part of our Project.

- USB Web Camera for capturing faces

- Solenoid Lock for opening/closing the door .

- Relay module which is necessary for the connection of solenoid lock .

- LCD display for displaying messages

2.5 Challenges

1. installing the OpenCV module in raspberry pi
   - **Problem:** To implement the face recognition system, the OpenCV library must be used to help train the program using deep learning. the raspberry pi computer has only 1 GB of RAM and can't process the heavy module installation
   - **solution:** reinstall it in an external hard disk and install it back to raspberry pi
2. finding a camera for our project
   - **problem:** since Arduino doesn't support USB devices, It was very hard to find a camera that connects directly to a breadboard. The part wasn't available in turkey and it is very expensive.
   - **solution:** switched from Arduino to raspberry pi to use a USB webcam
3. failed to find the user's name
   - **problem:** In order to print the name of the user after unlocking the door, the system must extract an id once a registered face is recognized. the system unlocks the door without knowing the name of the user and that had to be fixed
   - **solution:** added an ID to the name of the photos saved in the database and implemented an algorithm to extract the ID from the photo path and assign the name according to that ID

2.6 Timeline

Table 2.2: Timeline

WEEKS	ACTIVITIES
Week 2-3	Deciding the project topic Deciding the roles of group members Deciding the materials for project needs
Week 4	Preparing the necessary software Supplying the materials based on their price, and performance
Week 5	Preparing the proposal report Choosing the most useful codes and hardware design
Week 6	Starting to code on Raspberry pi Deciding on the last design of the hardware
Week 8	**MIDTERM WEEK**
Week 10	Starting to code on Raspberry pi Starting to learn the necessary Python libraries Deciding on the last design of the hardware
Week 11	Testing the facial recognition feature and determining their value on the Raspberry pi testing the functionality of the door lock Finalization of the project
Week 12	Preparing and practising on the presentation

2.7 Requirement List

Table 2.3: Requirement List

Number	Priority	Requirement
Req1	4	The system should detect faces from the webcam which is connected to the raspberry pi
Req2	5	The system should unlock the door to the registered person's face
Req3	3	The system should analyze the face in some seconds only.
Req4	2	The system should print the person's name in the LCD display once authorized
Req5	3	The system should consider different environments such as dim light, face mask, different angles, etc
Req6	4	The Hardware design must be user-friendly and easy to install to any door

Chapter 3

Approach and Methodology

The approach and methodology for our smart door lock project involved the following steps:.

1. Research and selection of appropriate hardware and software for the system, including the Raspberry Pi and the opencv library for Python.
2. Design and development of the user interface for enrolling new users and managing the database of authorized individuals.
3. Implementation of the facial recognition algorithm using the opencv library and testing of the system's accuracy and performance.
4. Integration of the system with the hardware components, including the camera and door lock.
5. Testing of the complete system in a real-world setting to assess its effectiveness and identify any issues or areas for improvement.
6. Iterative development and refinement of the system based on the results of testing.

Throughout the development process, we followed a systematic and methodical approach to ensure the reliability and accuracy of the system. We conducted extensive testing at each stage to ensure that the system met our goals and objectives, and made any necessary adjustments or improvements based on the results of the testing. The end result is a smart door lock system that is secure, convenient, and easy to use.

Figure 3.1

Chapter 4

Hardware Design

4.1 Camera and LCD display

The box has made of cardboard and it is totally handmade. As shown in Fig 4.1 there is a web camera on top of the box and Lcd display attached to the door

Figure 4.1: Prototype of the project. Rasbberry pi is hidden inside the Box

Because the project is detecting faces, a web camera has been used in the project to capture video feeds. it is located on the top of the box as shown in Fig 4.1. The main aim is detecting faces so the door can open to the user.

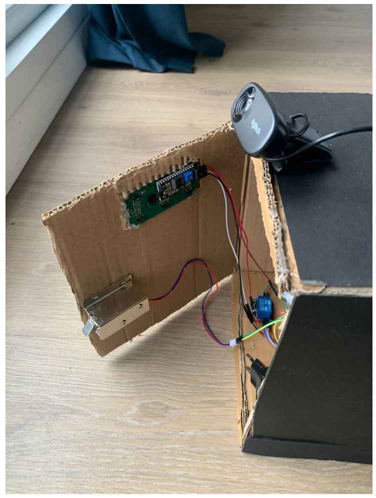

Figure 4.2: Inside of the Box

As shown in Fig 4.2, a solenoid lock is attached to the door to simulate a door lock with a latch, it is connected to a relay module to give the necessary power supply for the lock, the relay module is connected to the raspberry pi with jumper wires, Thanks to this, we can install the raspberry pi away from the door for convenience, the camera is connected directly to the raspberry pi via USB, which can be a problem. However, there are cameras that work using WiFi or via a breadboard as a bridge to use jumper wires for connection, finally, we have the LCD display module that is also connected to raspberry pi with Jumper wires. In order to start the system. The project needs a microcontroller for all these processes, which is the raspberry pi model B with an interface. raspberry pi must be connected to a computer or a monitor to run the necessary python files. The design looks like a house with a door to test its functionality, if a user is verified, the door will open and a welcome message will display. The box is totally handmade using only cardboard and a sealing wax gun.

4.2 Circuits

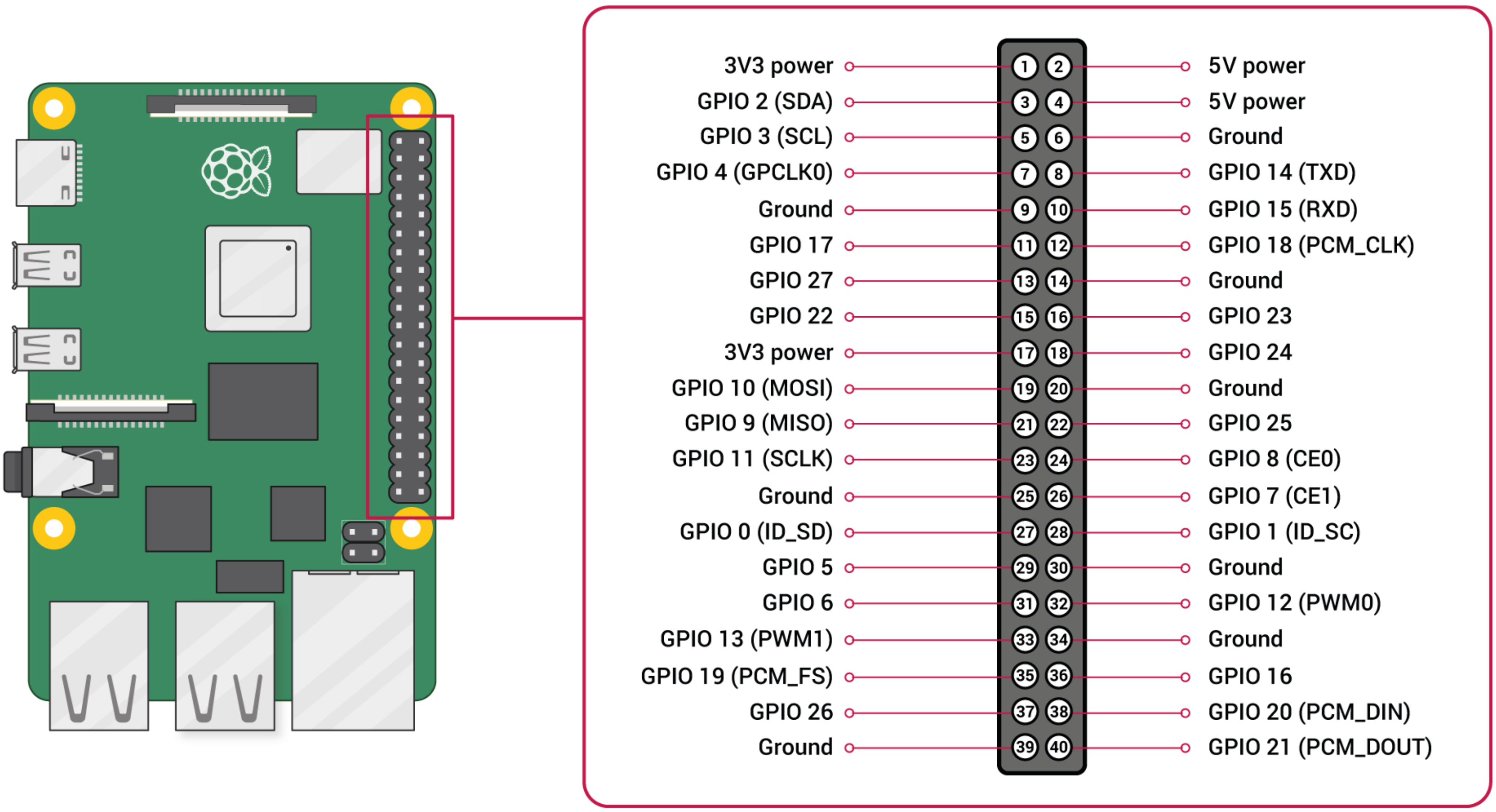

Figure 4.3: Raspberry pi GPIO

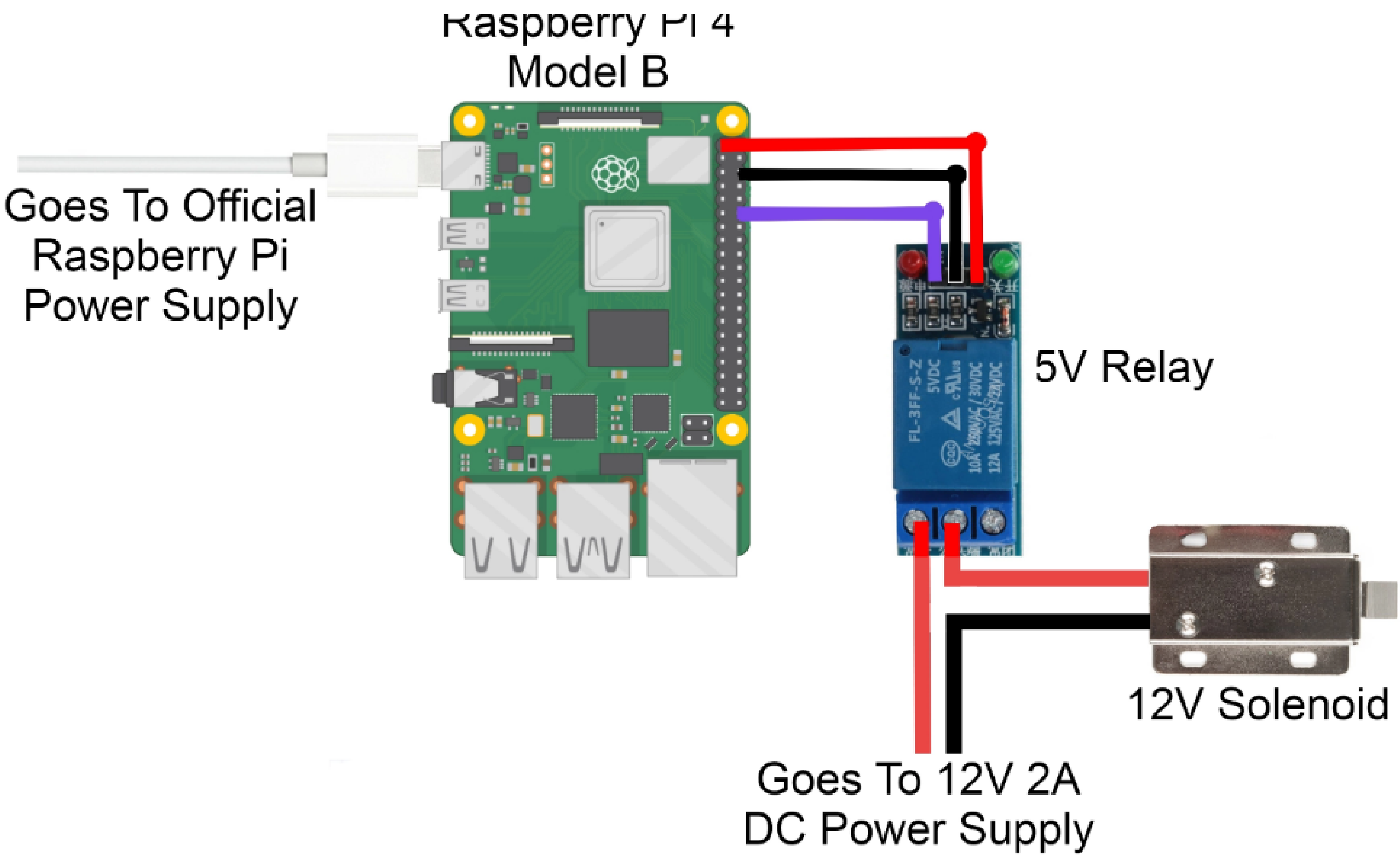

Figure 4.4: The lock circuit diagram

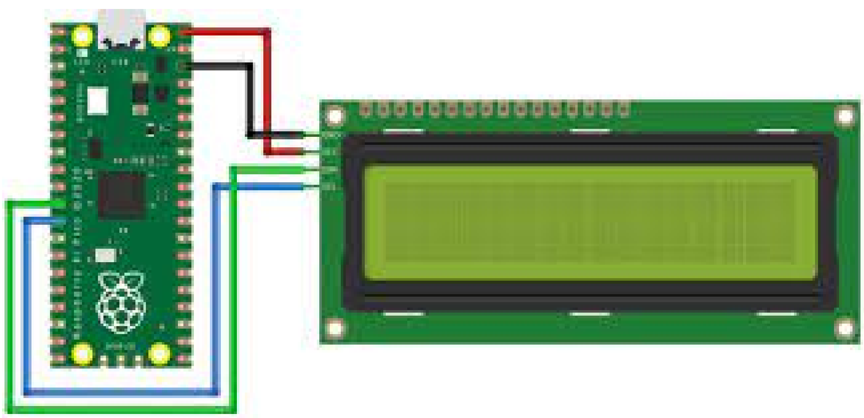

Figure 4.5: The LCD display circuit diagram

Chapter 5

Software Design

For the project, we used Linux-based Raspberry pi with Python3 IDE. source codes were written in python. 3 python files are used, one for registering the face to the data set,one for training the faces, and one for the testing and recognizing the faces which is the main file

5.1 Registering Faces Phase

```
import cv2
import drivers
cam = cv2.VideoCapture(0)
detector=cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

display=drivers.Lcd()
```

Figure 5.1

In Fig 5.1, cv2(OpenCV) module is imported, which is a library for computer vision tasks. also, drivers module is imported which is a model for using the LCD display. it contains predefined methods for displaying text, and characters in the LCD display.
The code then creates a VideoCapture object from the cv2 module, which allows the code to access the default video camera on the computer.

Next, the code creates a CascadeClassifier object from the cv2 module, which is a machine learning model trained to detect objects in images. The CascadeClassifier is initialized with a pre-trained model file called 'haarcascadefrontalfacedefault.xml'. which is a model that has been trained to detect frontal faces in images.

```python
while(True):
        ret, img = cam.read()
        gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces = detector.detectMultiScale(gray, 1.3, 5)

        for (x,y,w,h) in faces:
                sampleNum=sampleNum+1
                cv2.imwrite("dataset/User."+ str(Id)+"."+str(sampleNum) + ".jpg", gray[y:y+h,x:x+w])
                cv2.rectangle(img,(x,y),(x+w,y+h),(255,132,216),2)
                cv2.waitKey(100)
        cv2.imshow('Face',img)
        cv2.waitKey(1)
        display.lcd_display_string('registering face',1)
        display.lcd_display_string('please wait',2)
        if sampleNum>25:
                break
```

Figure 5.2

In Figure 5.2; This code is setting up an infinite loop that captures video frames from the default video camera, converts them to grayscale, and uses the detector object to detect any faces in the frame.

For each face that is detected, the code saves a copy of the face as an image file in the "dataset" directory, with a file name of the form "User.Id.sampleNum.jpg", where Id and sampleNum are variables that are incremented each time a face is detected. The code also draws a rectangle around the detected face in the original video frame.

The code then displays the video frame with the rectangle drawn on it and displays a message on the Lcd object. The loop continues until 25 faces have been detected, at which point it breaks out of the loop.

5.2 Training Phase

```python
def getImagesAndLabels(path):
#get the path of all the files in the folder
    imagePaths=[os.path.join(path,f) for f in os.listdir(path)]
#create empty face list
    faces=[]
#create empty ID list
    Ids=[]
#now looping through all the image paths and loading the Ids and the images
    for imagePath in imagePaths:
#loading the image and converting it to gray scale
        pilImage=Image.open(imagePath).convert('L')
#Now we are converting the PIL image into numpy array
        imageNp=np.array(pilImage,'uint8')
#getting the Id from the image
        Id=int(os.path.split(imagePath)[-1].split(".")[1])
        faces.append(imageNp)
        print (Id)
        Ids.append(Id)
        cv2.imshow("training",imageNp)
        cv2.waitKey(10)
    return faces,Ids

faces,Ids = getImagesAndLabels(path)
recognizer.train(faces, np.array(Ids))
recognizer.save('trainer.yml')
```

Figure 5.3

In Figure 5.3; we defined a function that takes a directory path as input and returns two lists: one containing images and the other containing corresponding user IDs.

The function first creates a list of file paths for all the files in the specified directory. It then sets up two empty lists called faces and Ids.

The function then loops through each file path in the list of file paths. For each file path, it opens the image, converts it to grayscale, and converts it to a NumPy array. It then extracts the user ID from the file name of the image and adds the user ID to the Ids list. It also adds the NumPy array representation of the image to the faces list. Finally, the function displays the image using OpenCV's imshow function.
After the loop is finished, the function returns the faces and Ids lists.

Finally , we train the face and their corresponding IDs using the training method from the 'cv2.face.LBPHFaceRecognizer-create()' Object, is a face recognition model that works by comparing the texture patterns around the eyes, nose, and mouth of a face. The trained model is saved to 'trianer.yml' file.

5.2.1 LBPHFaceRecognizer

The LBPHFaceRecognizer is a face recognition model that is part of the cv2 (OpenCV) library. It stands for "Local Binary Patterns Histograms Face Recognizer".

This model works by extracting texture patterns around the eyes, nose, and mouth of a face, and representing them as a histogram of local binary patterns (LBP). It compares these histograms to those of other faces in a database to determine whether or not the faces match.

The LBPHFaceRecognizer has several parameters that can be adjusted to fine-tune its performance. These include the radius and number of neighbors used to construct the LBP patterns, and the threshold used to determine whether or not two faces match.

To use the LBPHFaceRecognizer, you first need to create an instance of it and then train it on a set of face images and corresponding labels. Once the model is trained, you can use it to predict the labels (i.e., the identities) of new faces by passing it the new face images as input

5.2.2 CascadeClassifier

The CascadeClassifier object is part of the cv2 (OpenCV) library and is used to detect objects in images or video. It works by training a classifier on positive and negative samples of the object, and then using the trained classifier to detect instances of the object in new images.

In this case, the CascadeClassifier object is being used to detect faces in images. It is trained using the "haarcascadefrontalfacedefault.xml" file, which is a pre-trained classifier for detecting front faces. This file is an XML file that contains the trained classifier, which is a set of Haar-like features (a type of feature used in object detection).

To use the CascadeClassifier object, you first need to create an instance of it and pass it the path to the XML file containing the trained classifier. You can then use the detectMultiScale method of the CascadeClassifier object to detect objects in an image. This method returns a list of rectangles (x, y, width, height) that enclose the detected objects.

5.3 Recognizing Faces and Unlocking the Door

```python
import cv2
import drivers
from time import sleep
import RPi.GPIO as gp
gp.setwarnings(False)
gp.setmode(gp.BCM)
gp.setup(18,gp.OUT)
recognizer=cv2.face.LBPHFaceRecognizer_create();
recognizer.read('trainer.yml')
cascadePath = "haarcascade_frontalface_default.xml"
faceCascade = cv2.CascadeClassifier(cascadePath);
cam = cv2.VideoCapture(0)
display=drivers.Lcd()
font = cv2.FONT_HERSHEY_SIMPLEX
fontscale = 1
fontcolor = (255, 255, 255)
display.lcd_display_string("Door Locked",1)
```

Figure 5.4

In Figure 5.4; The code imports necessary libraries explained earlier in addition to RPi.GPIO module for interacting with the general-purpose input/output (GPIO) pins on the Raspberry Pi. The pre-trained model is loaded to recognize faces. and sets up some variables for use in displaying text on the LCD display

```python
def unlock(id):
        cv2.putText(im,str(id), (x,y+h),font,fontscale,fontcolor)
        display.lcd_clear()
        display.lcd_display_string('Door Unlocked',1)
        display.lcd_display_string('welcome '+str(id)+' :)',2)
        gp.output(18,1) #unlock the door
        sleep(5)
        gp.output(18,0)
        display.lcd_clear()
```

Figure 5.5

In Figure 5.5; the Unlock function unlocks the door by sending current using gp.output() method and displays a welcome message to the user whith his name. it also prints the user's name in the camera feed. this function will be called once a face is recognized

```
while True:
    ret, im =cam.read()
    gray=cv2.cvtColor(im,cv2.COLOR_BGR2GRAY)
    faces=faceCascade.detectMultiScale(gray, 1.3,5);
    for(x,y,w,h) in faces:
        cv2.rectangle(im,(x,y),(x+w,y+h),(33,255,134),2)
        id, conf = recognizer.predict(gray[y:y+h,x:x+w])
        print(conf)

    if(conf < 50):
            if(id ==1):
                id= "hamza"
                unlock(id)
                break
            if(id ==5):
                id="omar"
                unlock(id)
                break
            if(id ==2):
                id="david"
                unlock(id)
                break
            if(id==3):
             id="mustafa"
             unlock(id)
             break
    else:
        id="unknown"
        cv2.putText(im,str(id), (x,y+h),font,fontscale,fontcolor)
```

Figure 5.6

For each face that is detected, the code draws a rectangle around the face in the video frame and uses the recognizer object to predict the identity of the person in the face. It stores the predicted identity in the id variable and the confidence of the prediction in the conf variable.

If the confidence of the prediction is less than 50, the code checks the predicted identity against a set of known identities (1, 5, 2, and 3). If the predicted identity matches one of the known identities, the code calls the unlock function and passes the name of the person as an argument.
if the confidence of the prediction is more than 50, the face is not registered and the door will keep locked and 'unknown' label is displayed on the face

Chapter 6

Test Cases

Table 6.1: Test Cases

Test Case Type	Description	Test Step	Expected Result	Status
debugging	code must run without errors.	run the python files.	code should run and the camera is open for capturing faces.	PASS
Functionality	system must detect right faces	stand in front of the device.	door must open if face is registerd door must still be closed if face is not recognized.	PASS
Accuracy	Ensure system works in sensitive environments.	test it in dim light and different angles	System should work.	PASS
Functionality	Lcd screen display messages correctly.	wait for the door to open	displays a welcome message with the user's name.	PASS

Chapter 7

Vision

7.1 Social, Environmental and Economic Impact

The social impact of the smart door lock with face recognition is significant, as it has the potential to improve security and reduce the risk of burglaries and other crimes. By using face recognition technology, the door lock can accurately identify authorized individuals and prevent unauthorized access, helping to protect the safety of residents and property. Additionally, the smart door lock may also have a positive impact on the environment by reducing energy consumption, as it can be programmed to automatically lock or unlock based on the presence of authorized individuals.

The economic impact of the smart door lock is also significant, as it has the potential to save homeowners money on their energy bills and insurance premiums. By using the door lock to control energy consumption and enhance security, homeowners may be able to reduce their energy usage and potentially qualify for lower insurance rates. In addition, the smart door lock may also be attractive to businesses, as it can help to improve security and reduce the risk of losses due to theft or break-ins.

Overall, the smart door lock with face recognition has the potential to have a positive impact on society, the environment, and the economy. Its ability to enhance security and reduce energy consumption makes it a valuable tool for homeowners and businesses alike, and its potential to save money on energy and insurance costs makes it an attractive investment.

7.2 What can be done in the future?

There are several directions that the smart door lock with face recognition project could take in the future. One potential area for improvement is in the accuracy and speed of the face recognition algorithm. By continuously training and improving the algorithm, we could further reduce the risk of false positives and false negatives, resulting in an even more secure and reliable door lock. Additionally, we could also explore other types of biometric authentication, such as fingerprint recognition or voice recognition, to provide even more robust security.

Chapter 8

Equipments

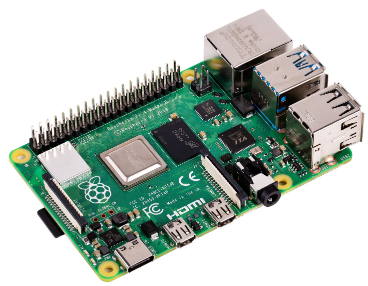

Figure 8.1: Raspberry pi

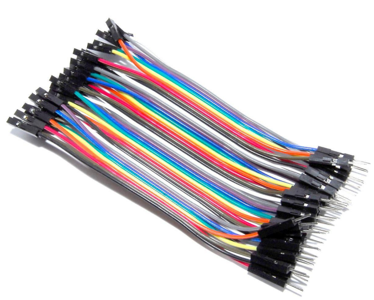

Figure 8.2: Jumper wires

Figure 8.3: Relay Module

Figure 8.4: HDMI Cable

Figure 8.5: Power Supply

Figure 8.6: Monitor

Figure 8.7: DC adapter

Figure 8.8: Web Camera

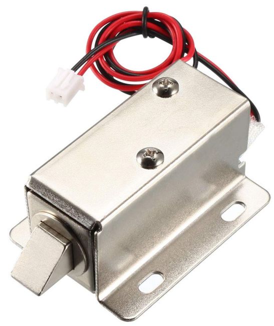

Figure 8.9: solenoid Lock

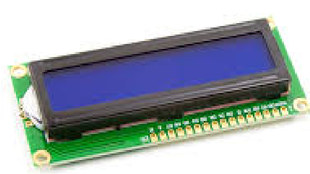

Figure 8.10: LCD display

Chapter 9

Cost Table and Analysis

The project costs 315 TL for the group. The group members paid it from their own pocket.
The team already had most of the stuff.
yaman had the raspberry pi and the relay module // Hamza had the web cam and the HDMI cable // Mhd Nour Khalifa had the cardboard and wax gun // Omar had the LCD display Also, backups of some materials were taken just in case.

Table 9.1: Cost Table

MATERIAL	**COST**
solenoid lock	240 TL
Dc power adapter	30 TL
Jumper wires	20 TL
Power Supply	25 TL
relay model	FREE
HDMI cable	FREE
LCD Display	FREE
web Camera	FREE
Raspberry pi 4 model B	FREE
Cardboard	FREE
wax Gun	FREE

Chapter 10

Conclusion

In conclusion, the smart door lock with face recognition project has been a successful and rewarding endeavor. By using a Raspberry Pi and openCV face recognition libraries, we were able to design and implement a functional smart door lock that can accurately identify authorized individuals and prevent unauthorized access.

Throughout the project, we faced several challenges, including choosing the appropriate hardware and software components, debugging and testing the system, and integrating the various components into a cohesive whole. However, through persistence and problem-solving skills, we were able to overcome these challenges and create a working prototype.

The results of our testing demonstrate that the smart door lock is a reliable and secure solution for controlling access to a home or business. Its ability to accurately recognize authorized individuals and prevent unauthorized access can help to improve security and reduce the risk of burglaries and other crimes. Additionally, the smart door lock has the potential to save homeowners money on their energy bills and insurance premiums by reducing energy consumption and enhancing security.

There are many potential directions for future work on the smart door lock project. For example, we could continue to improve the accuracy and speed of the face recognition algorithm, or explore other types of biometric authentication such as fingerprint recognition or voice recognition. We could also integrate the smart door lock with other home automation systems, such as Google Home or Amazon Alexa, to provide even more convenience and control. Finally, we could consider expanding the scope of the project to include additional security features, such as cameras or motion sensors, to create a more comprehensive security system.

Overall, the smart door lock with face recognition has been a successful and exciting project, and we are proud of the results we have achieved. We believe that this technology has the potential to make a positive impact on society, the environment, and the economy, and we look forward to continuing to develop and improve upon it in the future.

Chapter 11

References

- Components, Available on: https://www.trendyol.com//

- Setting up RaspBerry pi Availble on : https://youtu.be/BpJCAafw2qE

- OpenCV tutorial, Available on: https://youtu.be/PmZ29Vta7Vc/

- Lcd display installation script, Available on: https://github.com/the-raspberry-pi-guy/lcd/

- Controlling a Solenoid with Raspberry Pi and a Relay, Available on: https://core-electronics.com.au/guides/solenoid-control-with-raspberry-pi-relay/