ACCESS CONTROL SOLUTION USING SENSOR DATA FUSION FOR SECURITY PURPOSES

Submitted By:

- Capt Daud Asghar, BETE 54A
- Capt Kamran Raza, BETE 54A
- Capt Ali Akbar, BETE 54A
- Capt Muhammad Daniyal Malik, BETE 54A

Supervisor:

- Col ® Ateeq Ahmed

Submitted to the faculty of Department of Electrical Engineering,
Military College of Signals, National University of Sciences and Technology,
in partial fulfillment for the requirements of B.E Degree in Electrical Engineering July 2021

CERTIFICATE OF CORRECTIONS & APPROVAL

Certified that work contained in this thesis titled, "**ACCESS CONTROL SOLUTION USING SENSOR DATA FUSION FOR SECURITY PURPOSES**" carried out by **Capt Daud Asghar, Capt Kamran Raza, Capt Ali Akbar and Capt Muhammad Daniyal Malik** under the supervision of **Col ® Ateeq Ahmed** for partial fulfillment of Degree of Bachelors of Electrical Engineering, in Military College of Signals, National University of Sciences and Technology, Islamabad during the academic year 2020-2021 is correct and approved. The material that has been used from other sources it has been properly acknowledged / referred .

Approved by

Supervisor

Col ® Ateeq Ahmed

Date:

DECLARATION

No portion of work presented in this thesis has been submitted in support of another award or qualification in either this institute or anywhere else

Plagiarism Certificate (Turnitin Report)

This thesis has been checked for Plagiarism. Turnitin report endorsed by Supervisor is attached.

Signature of Supervisor

Col ® Ateeq Ahmed

Acknowledgements

We are thankful to Allah Subhana-Watala to have guided us throughout this work at every step. Indeed we could have done nothing without His help.

We are profusely thankful to our beloved parents who supported us throughout in every department of life.

I would also like to express special thanks to our supervisor Col ® Ateeq Ahmed for his help throughout our thesis and also for course of Linear Circuit Analysis courses which he has taught us in 2nd semester. We can safely say that we haven't learned any other engineering subject in such depth than the ones which he has taught.

Abstract

The science of identifying people by their physical characteristics is called **biometrics**. Biometrics is becoming an important field in science for several reasons, not the least of which is the heightened demand for security in a variety of situations.

Some kinds of biometric identification systems, like those that use patterns of blood vessels in the retina (the back of the eye) are very accurate. But generally, in order for a system like that to identify you, you have to voluntarily enroll yourself in its central database . Then, when you try to gain access to a facility or to classified information, the system takes a scan of your retina and matches it to that stored file to verify your identity.

‘That works great when you're trying to selectively identify a small group of people with special access privileges. But you can't positively identify anybody who isn't enrolled in the system; you can tell who they're not, but not who they are. Furthermore, you generally have to interact closely with these systems in order for them to check your identity (for example, walk up to a retinal scanner and push some buttons). Obviously, someone with malicious intentions would avoid that so, if you want to watch out for suspicious or dangerous individuals from a distance, these systems won't.

Gait recognition is one kind of biometric technology that can be used to monitor people without their cooperation . Some researchers are working on visually-based systems that use video cameras to analyze the movements of each body part knee, the foot, the shoulder, and so on . Marshall's team uses a radar-based system . As an individual walks toward the system, they're bombarded with invisible radio waves . Each individual's walking speed and style will make those waves bounce back a little differently. The result is a kind of composite signature that characterizes the overall feel of their walk.

This information couldn't pick out someone on the Most Wanted List of Law Enforcement Agencies if they wandered into an airport but it could be used to spot people who are moving around in suspicious ways, which may include repetitive walking patterns (suggesting they're

"casing out" a target) or movements that don't appear natural given their physicality. It could also be used in conjunction with other biometric systems to verify people identities and perhaps even weed out imposters .

Table of Contents

Certificate of Correction & Approval..... **iii**

Declaration..... **iv**

Plagiarism Certificate (Turnitin Report).....Error! Bookmark not defined.

Acknowledgements ..... **v**

Abstract..... **vi**

Table of Contents ..... **viii**

INTRO…….1-2

CHAPTER1: GAIT RECOGNITION…….……..…….….12-16

CHAPTER2:FACIAL RECOGNITION……...…..17-19

CHAPTER3: FINGERPRINT SENSOR……..…..……20-23

CHAPTER4:INTEGRATION/WORKING…...……..……24-26

HARDWARE AND SOFTWARE ……27-31

CONCLUSIONS …….32

ANNEXURE
(CODES)…….33-41

REFERENCES.....**42**

PLAGIARISM REPORT…….43

INTRODUCTION

Security at the entrance gates is an issue of grave seriousness for any institution. Now-adays, every company or institution issues some ID cards that are checked at the gates manually but the notion of using fake ID cards is always there. So, in tandem with manual checking of cards, there should be some other source of recognizing the person entering at the gates. Another major issue of concern, at the gates, is the human life i.e. life of security personnel or security guards is at a high risk. Thus there raises a calamitous need for the automated security system which requires no human assistance.

With the increasing importance of personal identification for the purpose of security in remote transactions in today's world of electronic communication, biometric identification techniques are rapidly evolving into a pervasive method for personal verification. Among the different biometrics proposed for such purpose, face recognition, fingerprints, hand prints, voice prints, retinal images, handwriting samples and etc.

Advantages and Disadvantages of Biometrics

Advantages:

Following are the advantages of Biometrics

- Increase security - provide a convenient and low-cost additional tier of security
- Reduce fraud by employing hard-to-forge technologies and materials.e.g. minimize the opportunity for ID fraud, buddy punching
- Eliminate problems caused by lost IDs or forgotten passwords by using physiological attributes.e.g. prevent unauthorized use of lost, stolen or "borrowed" ID cards

- Reduce password administration costs
- Replace hard-to-remember passwords which may be shared or observed
- Integrate a wide range of biometric solutions and technologies, customer applications and databases into a robust and scalable control solution for facility and network access
- Make it possible, automatically, to know *who* did *what, where* and *when!*
- Unequivocally link an individual to a transaction or event.

Disadvantages of a Biometric System

Following are the disadvantages of Biometrics

- The finger print of people working in chemical industries is often affected therefore these companies should not use the finger print mode of authentication.
- It is found that with age, the voice of a person becomes different. Besides, when the person has flu or throat infection the voice changes or if there is too much noise in the environment this method may not authenticate correctly. Therefore this method of verification is not workable all the time.
    - For people affected with diabetes, the eyes get affected resulting in differences.

Non Biometric sources in Security system

Non Biometrics is the science and technology of analyzing numeric data. Diferent non biometric sources such as Bar Code Reader, Magetic Stipe, Smart Cards, and RFID are used for the security purposes for analyzing, identifing and authentiaing persons.

CHAPTER 1: GAIT RECOGNITION

In this project you can find implementation of deep neural network for **people identification from video** by the characteristic of their **gait** . The processing is very robust against various covariate factors such as *clothing*, *carrying conditions*, *shoe types* and so on . Feel free to use this network in your project or extend it in some way.

Requirements

The code was written in Python 3.5, but it is probably also compatible with other versions .

Python packages

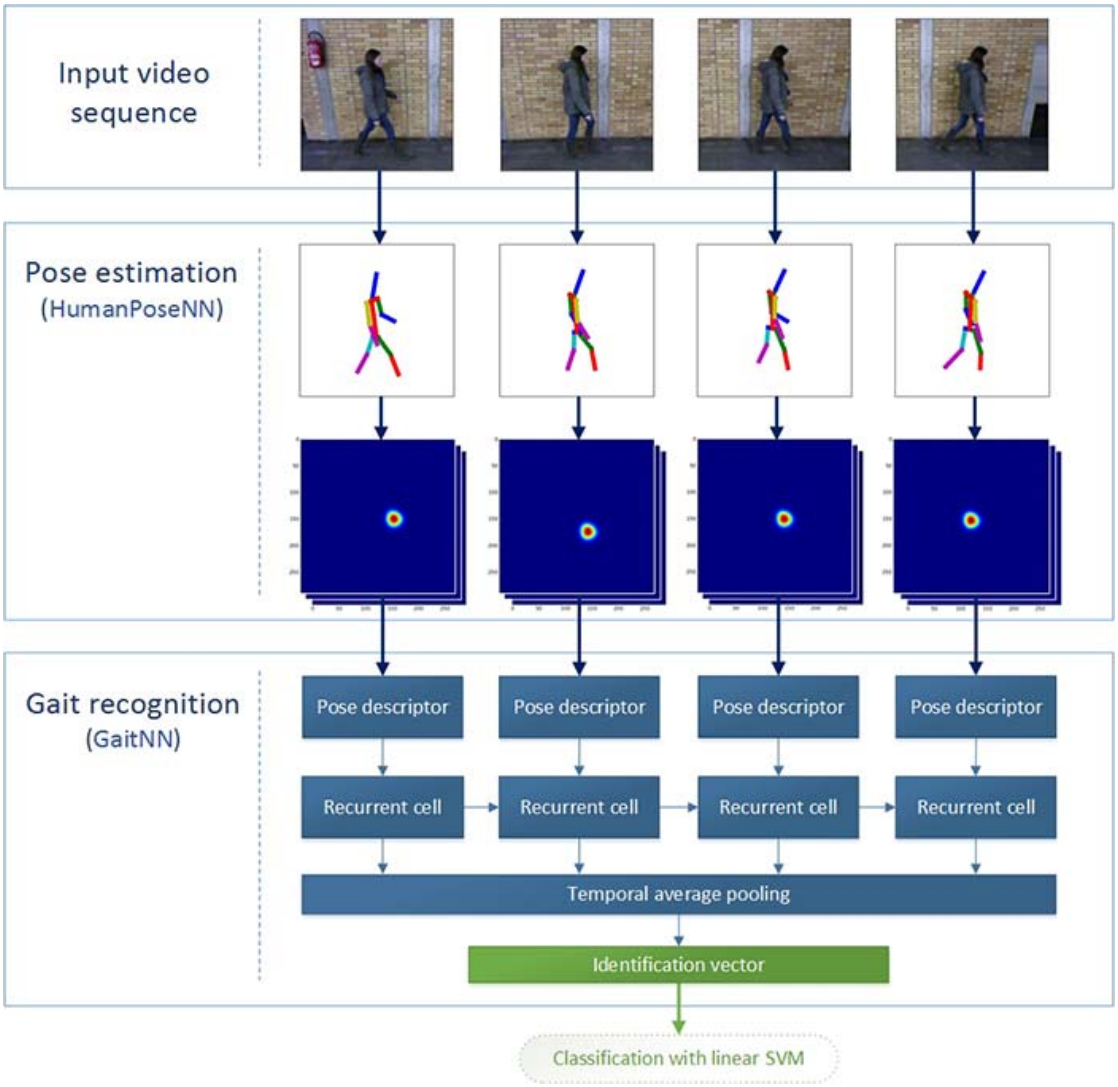
- TensorFlow 1.3
- numpy, scipy, PIL

Basic information about architecture

The network takes *raw RGB video frames* of a pedestrian as an input and produces one-dimensional vector - **gait descriptor** that exposes as an identification vector . The identification vectors from gaits of each two different people should be **linearly separable** . Whole network consists of two sub-networks connected in cascade - HumanPoseNN and GaitNN .

Spatial features from the video frames are extracted according to the descriptors that involve **pose of the pedestrian**. These descriptors are generated from the first sub-network - HumanPoseNN defined in human_pose_nn module. HumanPoseNN can be also used as a standalone network for regular **2D pose estimation problem** from still images.

Responsibility of the second sub-network - GaitNN defined in gait_nn module is the further processing of the generated spatial features into one-dimensional **pose descriptors** with the use of a residual convolutional network . **Temporal features** are then extracted across these *pose descriptors* with the use of the multilayer recurrent cells - **LSTM** or **GRU**. All temporal features are finally aggregated with **Average temporal pooling** into one-dimensional **identification vector** with good discriminatory properties. As already mentioned in the text above, the human identification vectors are linearly separable with each other and can therefore be classified with e.g. **linear SVM** .



Gait Recognition

The code below shows how to generate the identification vector form the input data video_frames. For the best results, all frames should include the **whole** person visible from the **profile view**. The person should be located approximately in the center of each frame.

Initialize computational graphs of both sub-networks

```
net_pose = HumanPoseIRNetwork()
net_gait = GaitNetwork(recurrent_unit = 'GRU', rnn_layers = 2)

# Load pre-trained models
net_pose.restore('path/to/pose_checkpoint.ckpt')
net_gait.restore('path/to/gait_checkpoint.ckpt')

# Create features from input frames in shape (TIME, HEIGHT, WIDTH, CHANNELS)
spatial_features = net_pose.feed_forward_features(video_frames)

# Process spatial features and generate identification vector
identification_vector = net_gait.feed_forward(spatial_features)
```

Pose Estimation

The first sub-network HumanPoseNN can be also used as a standalone network for 2D **pose estimation problem**. This can be done in such a way:

```
net_pose = HumanPoseIRNetwork()

# Restore pre-trained model
net_pose.restore('path/to/pose_checkpoint.ckpt')

# input_images should contain RGB images (299 x 299) to be processed.
# The images in batch should be stacked along the first dimension, so the shape of input_images
# has to be (BATCH, 299, 299, 3)
coords_y, coords_x, probabilities = net_pose.joint_positions(input_images)
```

where coords_y, coords_x and probabilities stores estimated joint coordinates in **Y axis**, **X axis** and **probability** of each estimate, respectively. All these tensors have shape (BATCH, 16), where the second dimension is the body joint. The order of the body joints in the second dimension is as follows:

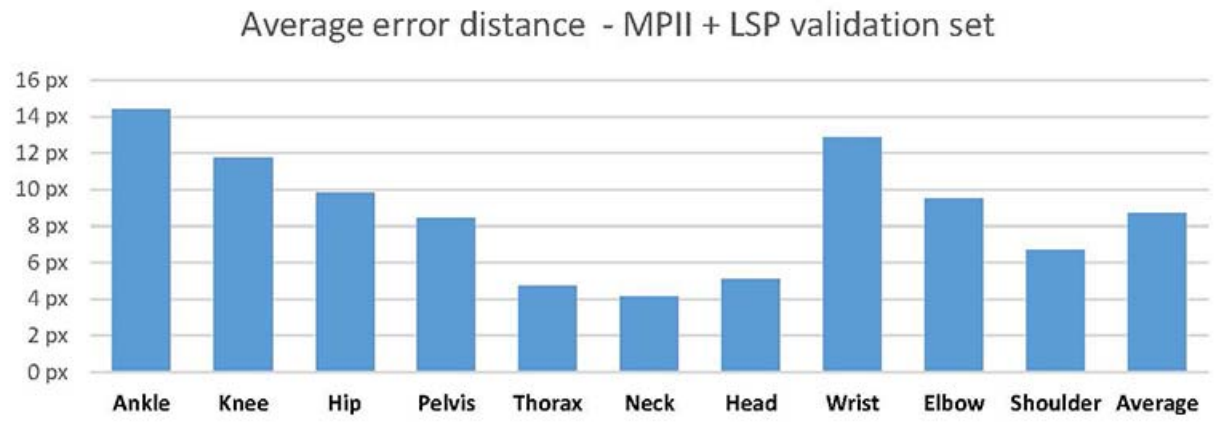
1. right ankle
2. right knee
3. right hip
4. left hip
5. left knee
6. left ankle
7. pelvis
8. thorax
9. upper neck
10. head top *(in human3.6m - head middle)*
11. right wrist
12. right elbow
13. right shoulder
14. left shoulder

15. left elbow
16. left wrist

Pre-trained models

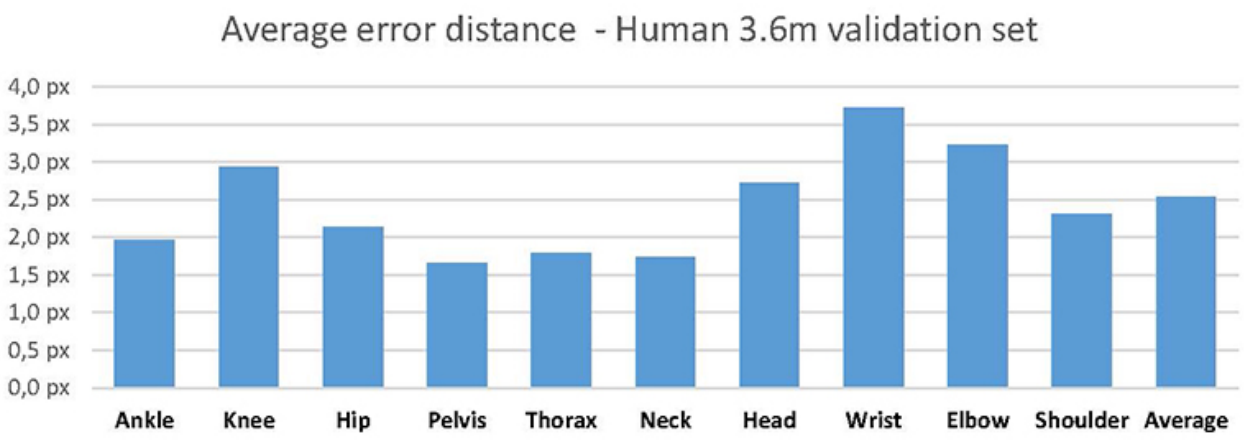
HumanPoseNN: MPII + LSP

The checkpoint MPII+LSP.ckpt was trained on images from MPII and LSP database. In the graph below you can see the average distance between predicted and desired joints on a **validation set of about 6 000 images** .



HumanPose NN: Human 3.6m

The checkpoint Human3.6m.ckpt was trained on the database Human 3.6m and only on the **walking** sequences of our two group members which was used for a validation purposes and the average distance between predicted and desired joints is shown in the following graph. As you can see, errors are smaller compared to MPII+LSP. It is because desired poses in Human 3.6m was labeled more precisely using motion capture system, so the a trained network can more accurately estimate the human pose. The second reason is that Human 3.6m sequences are very monotonous and thus human pose estimation is less challenging .


Average error distance - Human 3.6m validation set
4,0 px
3,5 px
3,0 px
2,5 px
2,0 px
1,5 px
1,0 px
0,5 px
0,0 px
Ankle
Knee
Hip
Pelvis
Thorax
Neck
Head
Wrist
Elbow
Shoulder
Average

CHAPTER 2: FACIAL RECOGNITION

A facial recognition system is a technology capable of matching a human face from a digital image or a video frame against a database of faces, typically employed to authenticate users through ID verification services, works by pinpointing and measuring facial features from a given image Background, Scope and Motivation .

In computer vision, one essential problem we are trying to figure out is to automatically detect objects in an image without human intervention . Face detection can be thought of as such a problem where we detect human faces in an image There may be slight differences in the faces of humans but overall, it is safe to say that there are certain features that are associated with all the human faces . There are various face detection algorithms but Viola-Jones Algorithm is one of the oldest methods that is also used today and we will use the same later in the article

Face detection is usually the first step towards many face-related technologies, such as face recognition or verification. However, face detection can have very useful applications. The most successful application of face detection would probably be photo taking When you take a photo of your friends, the face detection algorithm built into your digital camera detects where the faces are and adjusts the focus accordingly Now that we are successful in making such algorithms that can detect faces, can we also recognise whose faces are they?

Face recognition is a method of identifying or verifying the identity of an individual using their face. There are various algorithms that can do face recognition but their accuracy might vary Here I am going to describe how we do face recognition using deep learning So now let us understand how we recognise faces using deep learning . We make use of face embedding in which each face is converted into a vector and this technique is called deep metric learning.

Face Detection: The very first task we perform is detecting faces in the image or video stream. Now that we know the exact location/coordinates of face, we extract this face for further processing ahead .

Feature Extraction: Now that we have cropped the face out of the image, we extract features from it. Here we are going to use face embedding to extract the features out of the face. A neural network takes an image of the person's face as input and outputs a vector which represents the most important features of a face In machine learning, this vector is called embedding and thus we call this vector as face embedding. Now how does this help in recognizing faces of different persons While training the neural network, the network learns to output similar vectors for faces that look similar. For example, if I have multiple images of faces within different timespan, of course, some of the features of my face might change but not up to much extent. So in this case the vectors associated with the faces are similar or in short, they are very close in the vector space. Now after training the network, the network learns to output vectors that are closer to each other(similar) for faces of the same person(looking similar). We are not going to train such a network here as it takes a significant amount of data and computation power to train such networks. We will use a pre-trained network trained by Davis King on a dataset of ~3 million images. The

network outputs a vector of 128 numbers which represent the most important features of a face. Now that we know how this network works, let us see how we use this network on our own data. We pass all the images in our data to this pre-trained network to get the respective embeddings and save these embeddings in a file for the next step .

Comparing faces: Now that we have face embeddings for every face in our data saved in a file, the next step is to recognise a new t image that is not in our data. So the first step is to compute the face embedding for the image using the same network we used above and then compare this embedding with the rest of the embeddings we have. We recognise the face if the generated embedding is closer or similar to any other embedding .

What is OpenCV

In the field of Artificial Intelligence, Computer Vision is one of the most interesting and Challenging tasks. Computer Vision acts like a bridge between Computer Software and visualizations around us. It allows computer software to understand and learn about the visualizations in the surroundings. For Example: Based on the color, shape and size determining the fruit. This task can be very easy for the human brain however in the Computer Vision pipeline, first we gather the data, then we perform the data processing activities and then we train and teach the model to understand how to distinguish between the fruits based on size, shape and color of fruit.

Currently, various packages are present to perform machine learning, deep learning and computer vision tasks. By far, computer vision is the best module for such complex activities. OpenCV is an open-source library. It is supported by various programming languages such as R, Python. It runs on most of the platforms such as Windows, Linux and MacOS .

Advantages of OpenCV:

- OpenCV is an open-source library and is free of cost.
- As compared to other libraries, it is fast since it is written in C/C++.
- It works better on System with lesser **RAM**
- T supports most of the Operating Systems such as Windows, Linux and MacOS.

Face Recognition using Python

In this section, we shall implement face recognition using OpenCV and Python. First, let us see the libraries we will need and how to install them :

- OpenCV
- dlib
- Face_recognition

OpenCV is an image and video processing library and is used for image and video analysis, like facial detection, license plate reading, photo editing, advanced robotic vision, optical character recognition, and a whole lot more .

The dlib library, maintained by Davis King, contains our implementation of “deep metric learning” which is used to construct our face embeddings used for the actual recognition process .

The face recognition library, created by Adam Geitgey, wraps around dlib’s facial recognition functionality, and this library is super easy to work with and we will be using this in our code. Remember to install dlib library first before you install face recognition.

CHAPTER 3: FINGERPRINT SENSOR

Fingerprint modules come with FLASH memory to store the fingerprints and work with any microcontroller or system with TTL serial. These modules can be added to security systems, door locks, time attendance systems, and much more.

Prices for this sensor greatly vary from $10 to $50. We recommend checking the Fingerprint sensor module on Maker Advisor that compares the price in different stores. The fingerprint sensor modules featured on Maker Advisor should be compatible with this guide.

Specifications

Here's the specifications of the fingerprint sensor module we're using (you should check your sensor datasheet or the specifications provided by your supplier – they shouldn't be much different than these):

- Voltage supply: DC 3.6 to 6.0V
- Current supply: <120mA
- Backlight color: green
- Interface: UART
- Bad rate: 9600
- Safety level: five (from low to high: 1,2,3,4,5)
- False Accept Rate (FAR): <0.001% (security level 3)
- False Reject Rate (FRR): <1.0% (security level 3)
- Able to store 127 different fingerprints

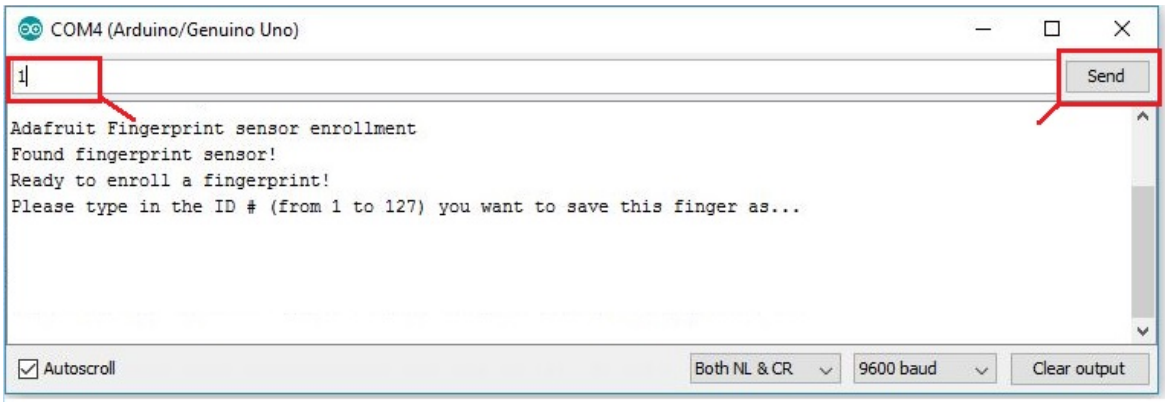
Enroll a New Fingerprint

Having the fingerprint sensor module wired to the Arduino, follow the next steps to enroll a new fingerprint . Make sure you've installed the Adafruit Fingerprint Sensor library previously.

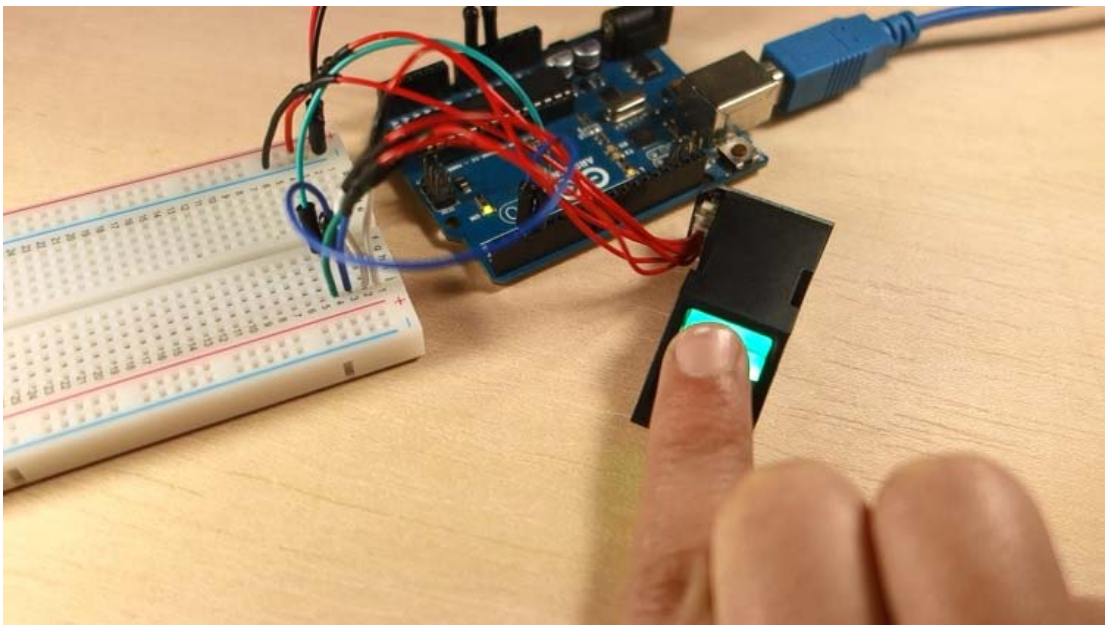
1. In the Arduino IDE, go to **File** > **Examples** > **Adafruit Fingerprint Sensor Library** > **Enroll**.

2. Upload the code, and open the serial monitor at a baud rate of 9600.

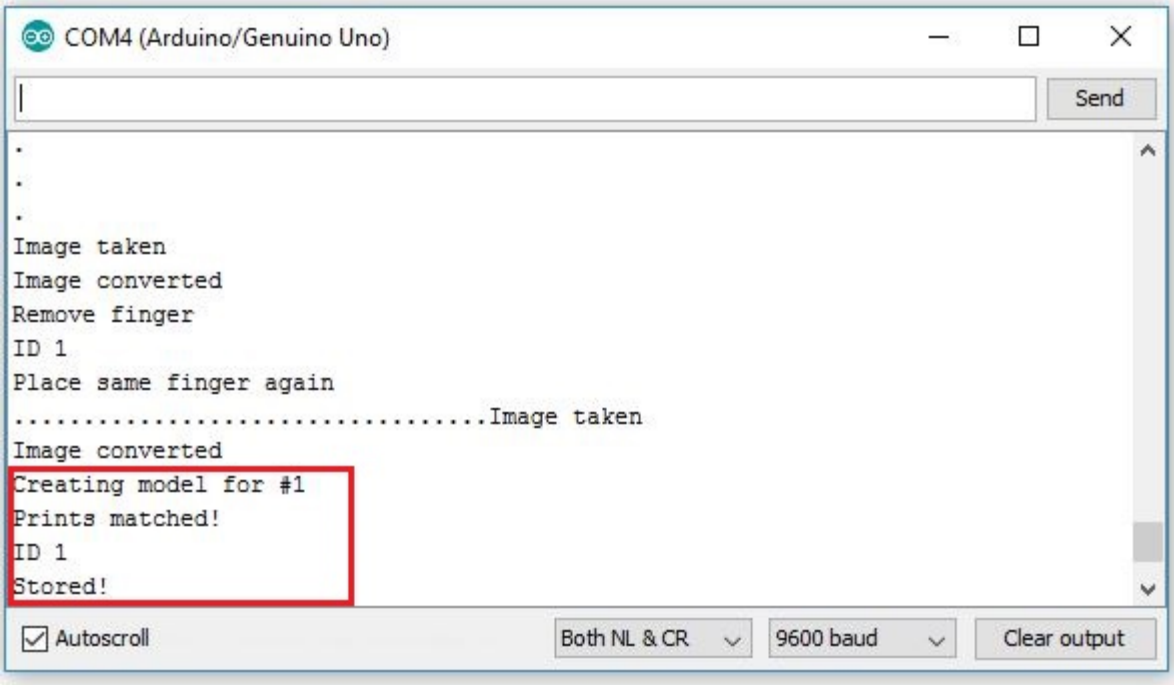
3. You should enter an ID for the fingerprint. As this is your first fingerprint, type 1 at the top left corner, and then, click the **Send** button.



4. Place your finger on the scanner and follow the instructions on the serial monitor.



You'll be asked to place the same finger twice on the scanner. If you get the "**Prints matched!**" message, as shown below, your fingerprint was successfully stored. If not, repeat the process, until you succeed.



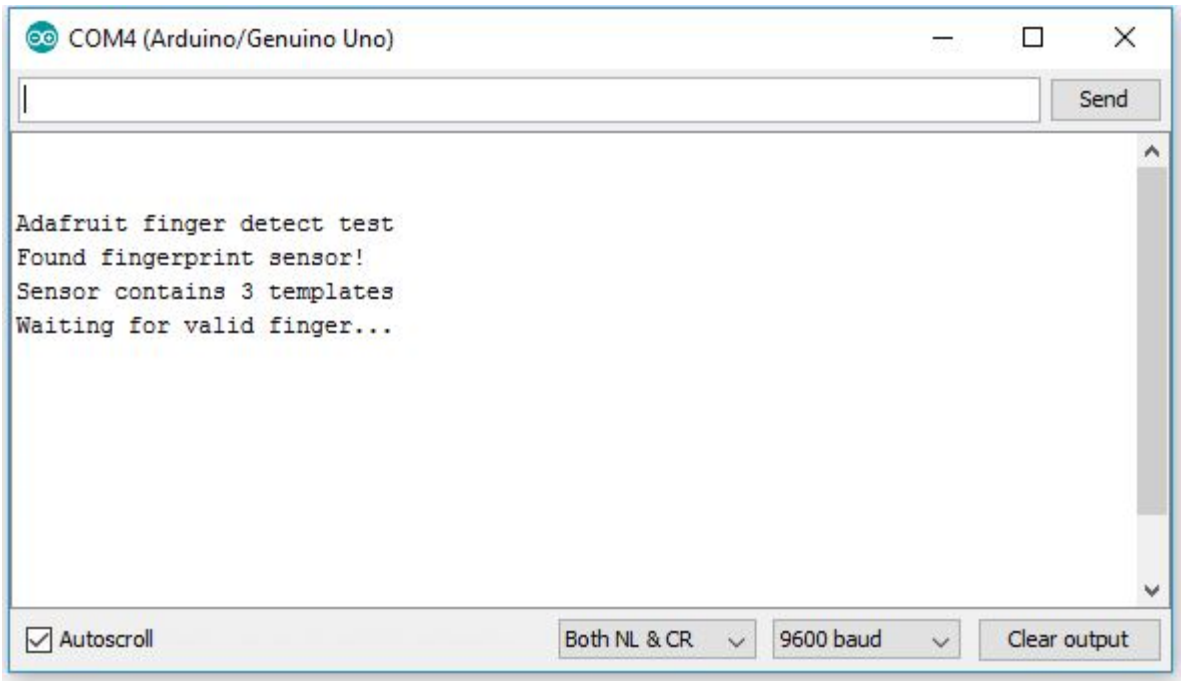
Store as many fingerprints you want using this method.

Finding a Match

You now should have several fingerprints saved on different IDs. To find a match with the fingerprint sensor, follow the next instructions.

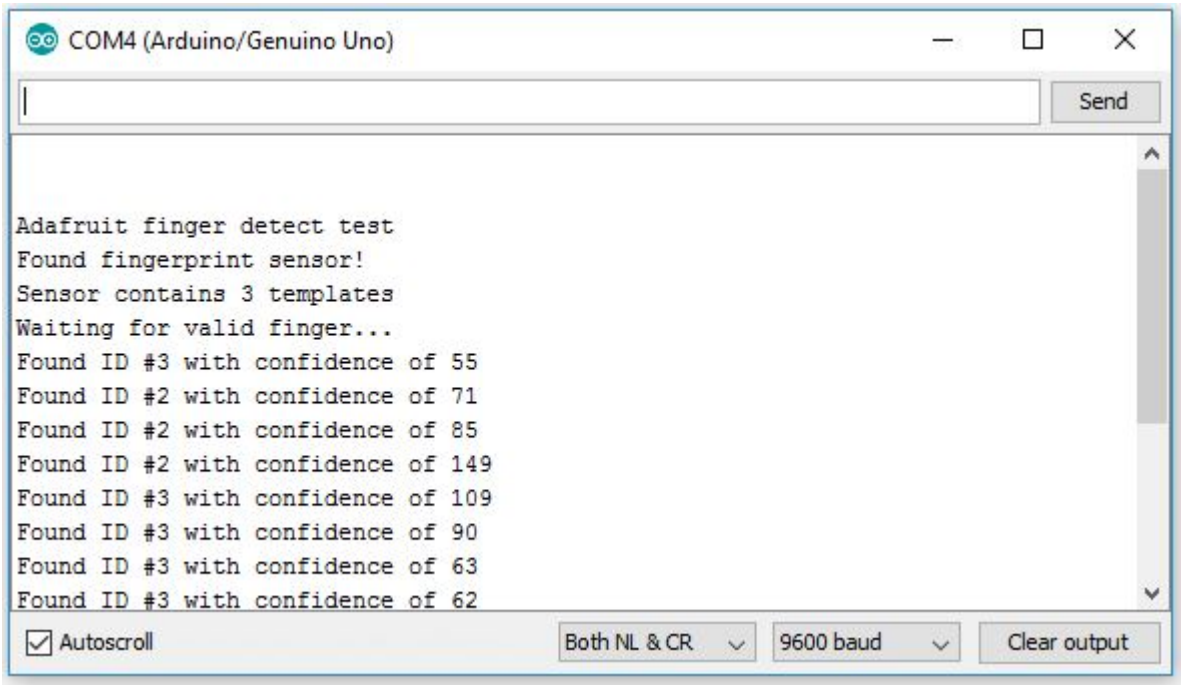
1. In the Arduino IDE, go to **File** > **Examples** > **Adafruit Fingerprint Sensor Library** > **Fingerprint** and upload the code to your Arduino board.

2. Open the Serial Monitor at a baud rate of 9600. You should see the following message:

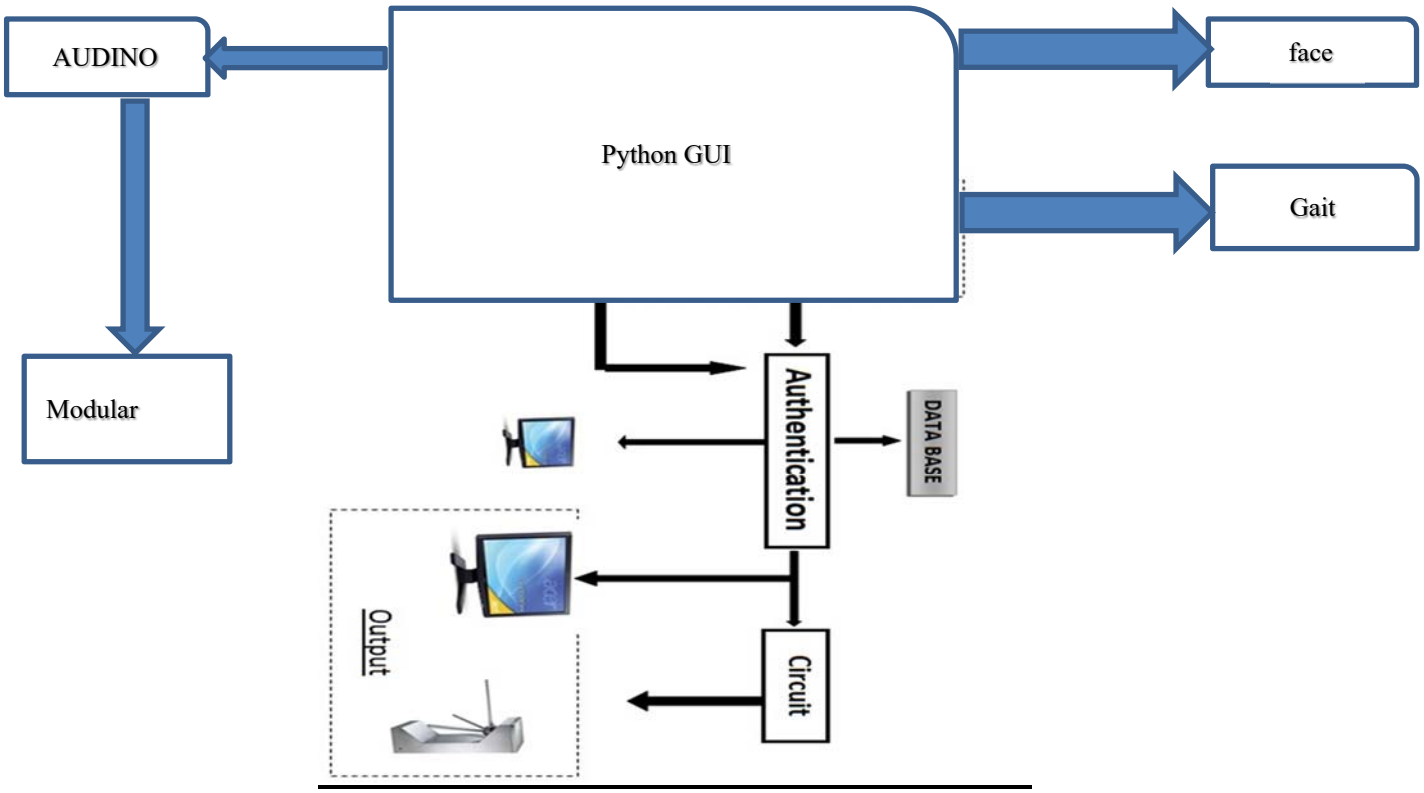


3. Place the finger to be identified on the scan.

4. On the serial monitor, you can see the ID that matches the fingerprint.It also shows the confidence – the higher the confidence, the similar the fingerprint is with the stored fingerprint .



CHAPTER 4: INTEGRATION/ WORKING



The project has been implemented as per above design.

- When a person comes for authentication, First His Gait input is taken.
- If his data is confirmed by the data base then his face and thumb recognition takes place and his data is verified from the facial data base.
- If the person goes through both levels of identification and he is verified and ONLY then he is allowed to pass though the walk through gate. The gate will automatically open for him.
- From the authentication block output is fed into the hardware circuit , which controls the gate .

- Along the hardware circuit output to a display screen is also provided which shows the output that whether the unknown person is recognized and can pass through gate or not .

Objectives

Academic Objectives

1) To study communication between bio devices
2) To learn different interfacing techniques
3) To learn and understand basics of image processing tools and techniques
4) To acquire appropriate coding language for developing windows application form
5) To be able to learn relevant coding for Microcontroller

Industrial Objectives

1) To provide cost effective solution for security system at enterance.
2) To bring combination of biometric and non-biometric sources based technology in everyday use
3) To develop an intelligent integrated security system prototype that may also be used for asset management, security and attendance system.
4) To provide fool proof security at entrance through intelligent face and card recognition
5) To deliver automated security system that requires no human assistance

Limitations

- There are certain limitations of the project

- Only one person can enter at a time at the gate. Multiple entries cannot be detected and authorized at the same time.
- The person entering at the gate should maintain a specific distance from the camera for his picture to be taken correctly and get recognized correctly.
- The output of face and Gait recognition can vary in extreme weather conditions.
- Entry of person is not allowed without his all three teirs

Hardware and Software Specifications

Hardware

Camera selection

Marking recognition accurately demands a good camera with good resolution, pan, and tilt and zoom capability. Good resolution is required for recognition of unknown persons entering at the gates.

DSLR (Digital Single Lens Reflex)

Initially a Digital Single Lens Reflex camera was used for capturing the face image but there were two major drawbacks of using this. Firstly DSLR isn't economically feasible to buy. Secondly it takes pictures manually. It cannot be controlled via controller laptop or computer.

IP Camera

Due to less efficiency of DSLR, IP camera was chosen for face detection but this camera also did not yield accurate results because of poor resolution. Hence facial features were not recognized accurately.

Webcam

Sinca IP camera had poor resolution, so Logitech sphere AF webcam was chosen. It is equipped with pan, tilt and digital zoom facility. The camera offers 189-degree field of view and 102-degree tilt. It is easy to interface it with visual C++. In poorly lit conditions, the camera is able to adjust intelligently thus producing best possible image.

Due to non-availability of optical zoom it can recognize faces correctly. It can be connected to laptop or PC directly with USB interface.

Microcontroller Module

The Hardware module of Automated Face Recognition system is controlled via central microcontroller module that is interfaced with the host system. The module does the job of controlling the gate. The microcontroller is interfaced with the host laptop or PC via USB interface. Commands are generated from there and the gate is controlled i-e opened or closed by the microcontroller circuit. A relay is used after the microcontroller for switching purposes.

USB Interface

Project's hardware module communicates with the host system to which the Camera is connected through the USB interface. The microcontroller has been programmed to allow USB interfacing and with minimal hardware components.A stable USB interface is accomplished that controls the entrance system module from a host computer system.

Selection of Gate

Barrier

A barrier is a physical structure that stops entrance of unwanted persons when it is controlled by some electronic means. It is most commonly used for security purposes.

Despite being most commonly used security gate, the barrier is not adopted for this project. The opening / closing time of barrier is very large. The barrier opens slowly and closes slowly. Secondly, more than two persons can pass at a time when the barrier is open. This is highly unfavorable for this project.

Figure: Barrier

Tripod gate

Tripod gate is second option to keep unwanted person from entering while using electric means. It allows persons to enter one by one. An RFID card reader is also attatched with the tripod gate. Its efficiency is better than electronic barrier and it occupies less space. Since it is costly, so this was option was also not chosen.

Figure:Tripod Gate

Gate

A Gate is specially designed for the project. This gate allows only one person to pass at a time. The structure of the gate is made of wood and aluminum. The frame of the door is wooden whereas aluminum and glass is used in door. The height of the door is 6 feet.

Mechanism for opening and closing of gate

The door of the gate has an electronic lock. A retractable mechanism has been installed in it that automatically brings the door back to its initial stage after it is opened. This causes the door of gate to close automatically after the person has passed.

If the person is recognized, a command from the host PC is generated and through the microcontroller module to unlock the gate and hence, the gate opens. The gate automatically closes after the person passes. If the unknown person is not recognized by the system, the gate remains closed.

Alarm system

The gate opens if the person is recognized by the integrated security system. If an unknown person tries to enter the gate, it does not open and an alarm system is activated.

Software

Python

python is one of the best Softwares used for simulations as it has lot of image processing tools but it is extremely slow when it comes to processing video streams from a large number of cameras.

OpenCV Library

In Microsoft Visual Studio, OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision, developed by Intel.

It focuses mainly on real-time image processing. If the library finds Intel's Integrated Performance Primitives on the system, it will use these proprietary optimized routines to accelerate it. For the project, the researchers have used OpenCV library in Microsoft Visual Studio.

Conclusion

The project Integrated Security System provides fool proof security at the entrance. We have implemented this project at the entrance of the gate. It encompasses proficient use of bio-metric and non-biometric sources. A customized gate is designed for the project that has automatically opening and closing system.

The project effectively uses the modern computer vision techniques for Gait recognition which provides 90% accuracy. The time for complete authentication of person can be changed as per requirements. It provides a user friendly interface for the person entering at the gate.

ANNEX

Extracting features from Face

First, you need to get a dataset or even create one of you own. Just make sure to arrange all images in folders with each folder containing images of just one person.

Next, save the dataset in a folder the same as you are going to make the file. Now here is the code:

```
from imutils import paths
import face_recognition
import pickle
import cv2
import os
 #get paths of each file in folder named Images
#Images here contains my data(folders of various persons)
imagePaths = list(paths.list_images('Images'))
knownEncodings = []
knownNames = []
# loop over the image paths
for (i, imagePath) in enumerate(imagePaths):
    # extract the person name from the image path
    name = imagePath.split(os.path.sep)[-2]
    # load the input image and convert it from BGR (OpenCV ordering)
    # to dlib ordering (RGB)
    image = cv2.imread(imagePath)
    rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    #Use Face_recognition to locate faces
    boxes = face_recognition.face_locations(rgb,model='hog')
    # compute the facial embedding for the face
    encodings = face_recognition.face_encodings(rgb, boxes)
    # loop over the encodings
    for encoding in encodings:
        knownEncodings.append(encoding)
        knownNames.append(name)
#save emcodings along with their names in dictionary data
data = {"encodings": knownEncodings, "names": knownNames}
#use pickle to save data into a file for later use
f = open("face_enc", "wb")
f.write(pickle.dumps(data))
close()
```

Face Recognition in Images

The script for detecting and recognising faces in images is almost similar to what you saw above. Try it yourself and if you can't take a look at the code below:

```
1  import face_recognition
2  import imutils
3  import pickle
4  import time
5  import cv2
6  import os
7   #find path of xml file containing haarcascade file
8  cascPathface = os.path.dirname(
9   cv2.__file__) + "/data/haarcascade_frontalface_alt2.xml"
10# load the harcaascade in the cascade classifier
11faceCascade = cv2.CascadeClassifier(cascPathface)
12# load the known faces and embeddings saved in last file
13data = pickle.loads(open('face_enc', "rb").read())
14#Find path to the image you want to detect face and pass it here
15image = cv2.imread(Path-to-img)
16rgb = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
17#convert image to Greyscale for haarcascade
18gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
19faces = faceCascade.detectMultiScale(gray,
20  caleFactor=1.1,
21  minNeighbors=5,
22  minSize=(60, 60),
23  flags=cv2.CASCADE_SCALE_IMAGE)
24 # the facial embeddings for face in input
25encodings = face_recognition.face_encodings(rgb)
26names = []
27# loop over the facial embeddings incase
28# we have multiple embeddings for multiple fcaes
29for encoding in encodings:
30    #Compare encodings with encodings in data["encodings"]
31    #Matches contain array with boolean values and True for the embeddings it matches closely
32    #and False for rest
33    matches = face_recognition.compare_faces(data["encodings"],
34    encoding)
35    #set name =inknown if no encoding matches
36    name = "Unknown"
37    # check to see if we have found a match
38    if True in matches:
39    #Find positions at which we get True and store them
40    matchedIdxs = [i for (i, b) in enumerate(matches) if b]
41    counts = {}
```

```
42    # loop over the matched indexes and maintain a count for
43    #each recognized face face
44    for i in matchedIdxs:
45     #Check the names at respective indexes we stored in matchedIdxs
46     name = data["names"][i]
47     #increase count for the name we got
48    counts[name] = counts.get(name, 0) + 1
49    #set name which has highest count
50    name = max(counts, key=counts.get)
51   # update the list of names
52    names.append(name)
53    # loop over the recognized faces
54    for ((x, y, w, h), name) in zip(faces, names):
55    # rescale the face coordinates
56    # draw the predicted face name on the image
57    cv2.rectangle(image, (x, y), (x + w, y + h), (0, 255, 0), 2)
58    cv2.putText(image, name, (x, y), cv2.FONT_HERSHEY_SIMPLEX,
59    0.75, (0, 255, 0), 2)
60    cv2.imshow("Frame", im
```

FINGER PRINT CODE:

Code

Before uploading the code, you need to enroll different fingerprints from different persons. Go to "**Enroll a New Fingerprint"** section above, upload the given code and follow the instructions to enroll two fingerprints.

Then, modify the code so that the fingerprint IDs match the name of the persons enrolled – scroll down to page for an explanation of the code. Finally, you can upload the code provided.

```
/*********

  Rui Santos

  Complete project details at https://randomnerdtutorials.com

*********/



#include <Wire.h>
```

```
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define OLED_RESET 4
Adafruit_SSD1306 display(OLED_RESET);

#include <Adafruit_Fingerprint.h>
#include <SoftwareSerial.h>
SoftwareSerial mySerial(2, 3);

Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial);
int fingerprintID = 0;
String IDname;

void setup(){
  //Fingerprint sensor module setup
  Serial.begin(9600);
  // set the data rate for the sensor serial port
  finger.begin(57600);

  if (finger.verifyPassword()) {
    Serial.println("Found fingerprint sensor!");
  }
```

```
  else {
    Serial.println("Did not find fingerprint sensor :(");
    while (1) { delay(1); }
  }

  //OLED display setup
  Wire.begin();
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  //displays main screen
  displayMainScreen();
}

void loop(){
  displayMainScreen();
  fingerprintID = getFingerprintIDez();
  delay(50);
  if(fingerprintID == 1 || fingerprintID == 3 || fingerprintID == 4 || fingerprintID == 5){
    IDname = "Sara";
    displayUserGreeting(IDname);
  }
  else if(fingerprintID == 2){
    IDname = "Rui";
```

```
    displayUserGreeting(IDname);
  }
}

// returns -1 if failed, otherwise returns ID #
int getFingerprintIDez() {
  uint8_t p = finger.getImage();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.image2Tz();
  if (p != FINGERPRINT_OK)  return -1;

  p = finger.fingerFastSearch();
  if (p != FINGERPRINT_OK)  return -1;

  // found a match!
  Serial.print("Found ID #");
  Serial.print(finger.fingerID);
  Serial.print(" with confidence of ");
  Serial.println(finger.confidence);
  return finger.fingerID;
}
```

```
void displayMainScreen(){
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(7,5);
  display.println("Waiting fingerprint");
  display.setTextSize(1);
  display.setTextColor(WHITE);
  display.setCursor(52,20);
  display.println("...");
  display.display();
  delay(2000);
}

void displayUserGreeting(String Name){
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(2);
  display.setCursor(0,0);
  display.print("Hello");
  display.setCursor(0,15);
```

```
  display.print(Name);

  display.display();

  delay(5000);

  fingerprintID = 0;

}
```

Setup

In the setup(), both the fingerprint sensor and the OLED display are initialized. We also print a message on the serial monitor so that we know if the fingerprint sensor was found successfully.

```
void setup(){
 //Fingerprint sensor module setup
 Serial.begin(9600);
 // set the data rate for the sensor serial port
 finger.begin(57600);

 if (finger.verifyPassword()) {
   Serial.println("Found fingerprint sensor!");
 }
 else {
   Serial.println("Did not find fingerprint sensor :(");
   while (1) { delay(1); }
 }

//OLED display setup
 Wire.begin();
 display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
 //displays main screen
 displayMainScreen();
}
```

Loop

In the loop(), the code displays the main screen on the OLED display – this is done in the displayMainScreen() function. Then, the code is continuously checking for incoming fingerprints. If the sensor founds a saved fingerprint, the Arduino saves the corresponding ID in the fingerprintID variable.

Then, the code has an if/else statement to check the ID the fingerprint corresponds to. You should edit the following lines of code with the corresponding IDs and names.

```
if(fingerprintID == 1 || fingerprintID == 3 || fingerprintID == 4 || fingerprintID == 5){
  IDname = "Sara";
  displayUserGreeting(IDname);
}
else if(fingerprintID == 2){
  IDname = "Rui";
```

Sometimes, the sensor will recognize a fingerprint better if it is saved several times in different IDs. After identifying the ID name, the OLED displays a greeting – this is done in the displayUserGreeting() function.

REFERENCES

1. 'Ruff Improved gait recognition by gait dynamics normalization by Z Liu, S Sarkar'

2. 'Automatic gait recognition MS Nixon, JN Carter, D Cunado, PS Huang'

3. 'Gait recognition without subject cooperation K Bashir, T Xiang, S Gong - Pattern Recognition Letters, 2010'

4. 'Gait recognition using a view transformation model in the frequency domain by Y Makihara, R Sagawa, Y Mukaigawa, T Echigo'

5. 'Silhouette analysis-based gait recognition for human identification by L Wang, T Tan, H Ning, W Hu'

6. 'Human gait recognition by R Zhang, C Vogler, D Metaxas'

7. 'Liang Wang, Tieniu Tan, Huazhong Ning, and Weiming Hu, "Silhouette Analysis-Based Gait Recognition for Human Identification", IEEE transactions on pattern analysis and machine intelligence, Vol. 25, No. 12, December 2003'

8. X. Yang, Y. Zhou, T. Zhang, G. Shu, J. Yang, "Gait Recognition Based on Dynamic Region Analysis," Signal Processing, Vol. 88, No. 9, pp. 2350–2356, September 2008.