AUTOMATIC ATTENDANCE SYSTEM

USING

FACE RECOGNITION

BY

T. SRIRAM **2020504587**

AUTOMATIC ATTENDANCE SYSTEM

USING

FACE RECOGNITION

BY

Nanda kishore.M G-2020504548
Prathipraj.R-2020504560
Vinodh kumar.V-2020504599

Abstract

Creating Automatic Attendance System
using Face recognition which is developed with the help of opencv and record the attendance in an organized format

Image Processing

- Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image.

Resources Used

software used:

- Pycharm Editor

Libraries Used:

- OpenCV-python
- Dlib
- Face-recognition
- Numpy
- Cmake

OpenCV-python:

It stands for Open Source Computer Vision Library. This library consists of around 2000+ optimised algorithms that are useful for computer vision and machine learning. There are several ways you can use opencv in image processing, a few are listed below:

- Converting images from one color space to another i.e. like between BGR and HSV, BGR and gray etc.
- Performing thresholding on images, like, simple thresholding, adaptive thresholding etc.
- Smoothing of images, like, applying custom filters to images and blurring of images.
- Performing morphological operations on images.

Dlib:

- Dlib is one of the most powerful and easy-to-go open-source library consisting of machine learning library/algorithms and various tools for creating software.

Face-recognition:

- Python Module. Automatically find all the faces in an image. Automatically locate the facial features of a person in an image. Recognize faces in images.

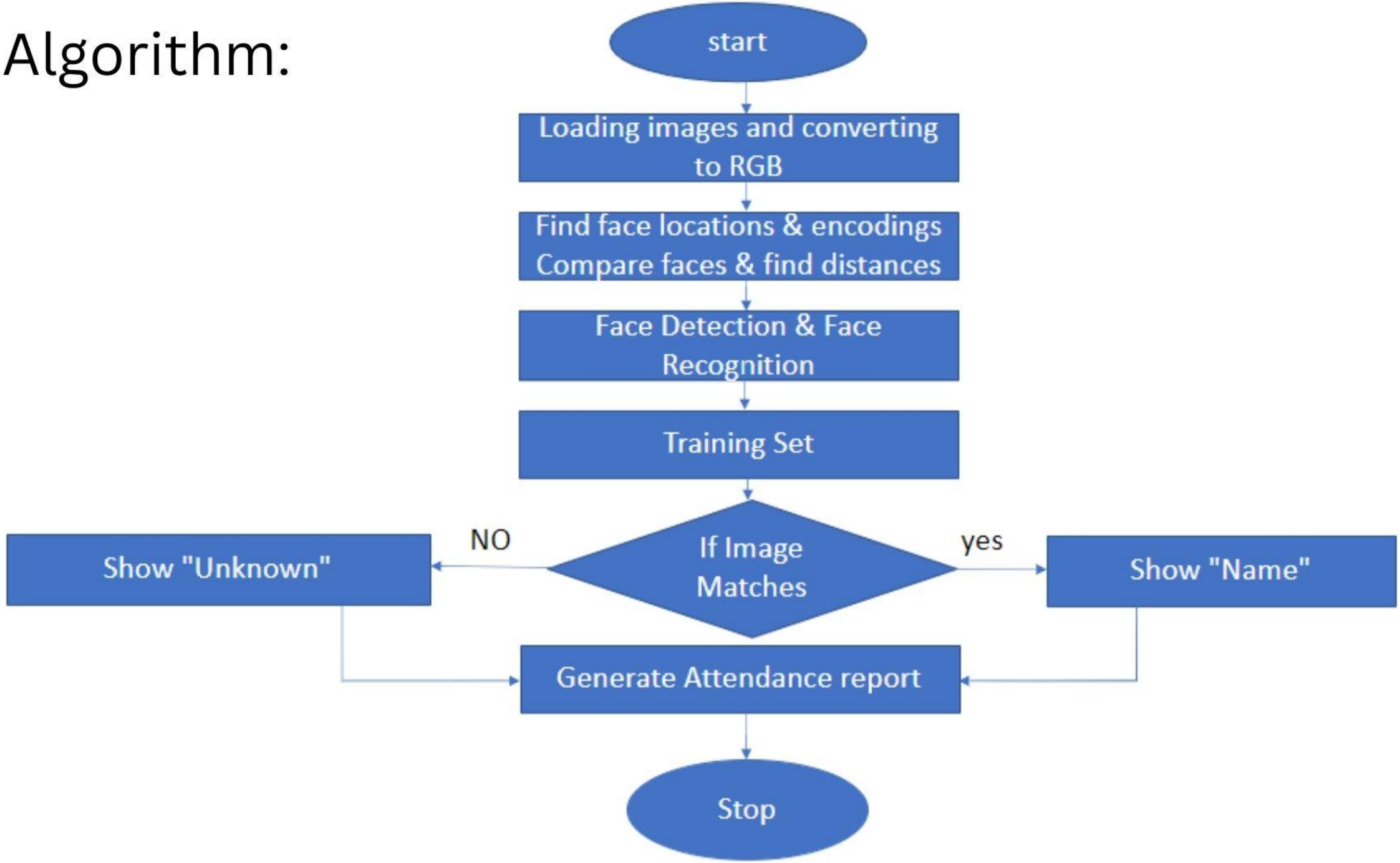
Numpy:

- NumPy can be used to perform a wide variety of mathematical operations on arrays. It adds powerful data structures to Python that guarantee efficient calculations with arrays and matrices and it supplies an enormous library of high-level mathematical functions

Cmake:

- CMake is cross-platform free and open-source software for build automation, testing, packaging and installation of software by using a compiler-independent method.

Algorithm:


start
Loading images and converting to RGB
Find face locations & encodings
Compare faces & find distances
Face Detection & Face Recognition
Training Set
If Image Matches
NO
yes
Show "Unknown"
Show "Name"
Generate Attendance report
Stop

Source Code:

main.py | Attendance.csv | face-recognition.py

```python
import cv2
import numpy as np
import face_recognition
import os
from datetime import datetime


# from PIL import ImageGrab

path = 'photos'
images = []
classNames = []
myList = os.listdir(path)
print(myList)
for cl in myList:
    curImg = cv2.imread(f'{path}/{cl}')
    images.append(curImg)
    classNames.append(os.path.splitext(cl)[0])
print(classNames)
```

```python
def findEncodings(images):
    encodeList = []
    for img in images:
        img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB)
        encode = face_recognition.face_encodings(img)[0]
        encodeList.append(encode)
    return encodeList


def markAttendance(name):
    with open('Attendance.csv','r+') as f:
        myDataList = f.readlines()
        nameList = []
        for line in myDataList:
            entry = line.split(',')
            nameList.append(entry[0])
        if name not in nameList:
            now = datetime.now()
            dtString = now.strftime('%H:%M:%S')
            f.writelines(f'\n{name},{dtString}')
```

```
encodeListKnown = findEncodings(images)
print('Encoding Complete')
cap = cv2.VideoCapture(0)

while True:
    success, img = cap.read()
    #img = captureScreen()
    imgS = cv2.resize(img, (0,0),None,0.25,0.25)
    imgS = cv2.cvtColor(imgS, cv2.COLOR_BGR2RGB)
    facesCurFrame = face_recognition.face_locations(imgS)
    encodesCurFrame = face_recognition.face_encodings(imgS, facesCurFrame)
```

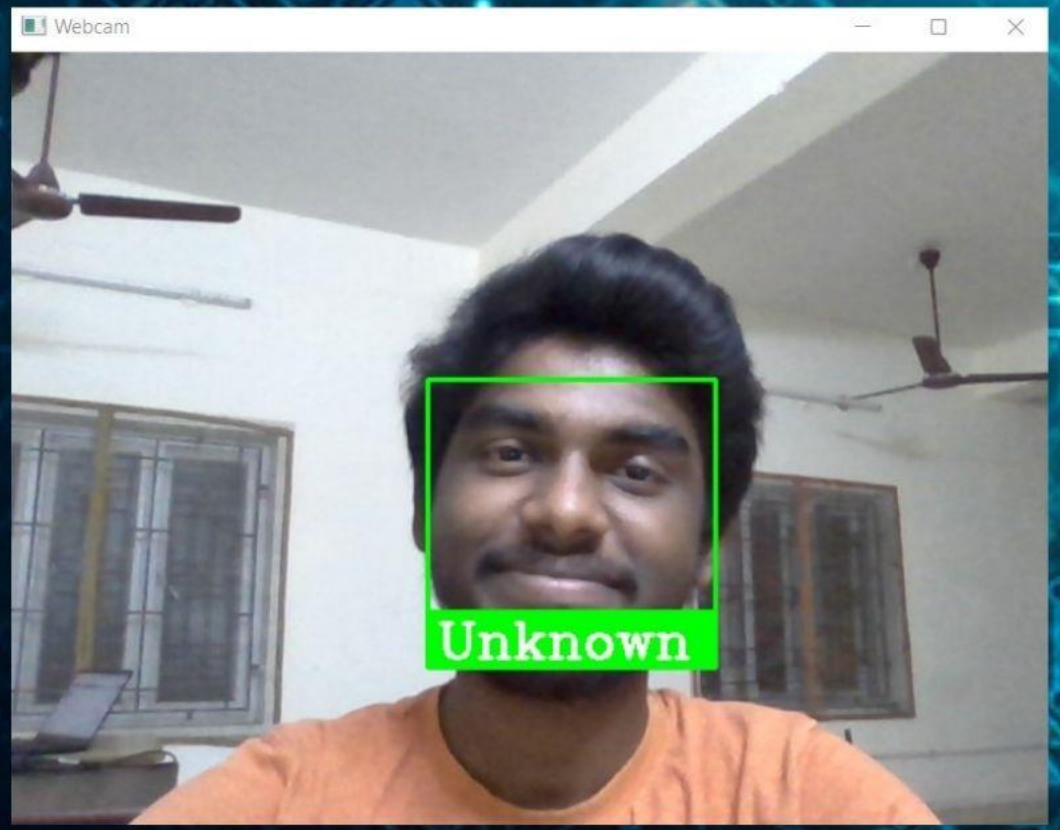
```python
    for encodeFace, faceLoc in zip(encodesCurFrame, facesCurFrame):
        matches = face_recognition.compare_faces(encodeListKnown, encodeFace)
        faceDis = face_recognition.face_distance(encodeListKnown, encodeFace)
        print(faceDis)
        matchIndex = np.argmin(faceDis)



        if faceDis[matchIndex] < 0.50:
            name = classNames[matchIndex].upper()
            markAttendance(name)
            print(name)
        else:
            name = 'Unknown'
            print(name)



        y1, x2, y2, x1 = faceLoc
        y1, x2, y2, x1 = y1 * 4, x2 * 4, y2 * 4, x1 * 4
        cv2.rectangle(img, (x1, y1), (x2, y2), (0, 255, 0), 2)
        cv2.rectangle(img, (x1, y2 - 35), (x2, y2), (0, 255, 0), cv2.FILLED)
        cv2.putText(img, name, (x1 + 6, y2 - 6), cv2.FONT_HERSHEY_COMPLEX, 1, (255, 255, 255), 2)

    cv2.imshow('Webcam', img)
    cv2.waitKey(1)
```

Recognition of known Faces

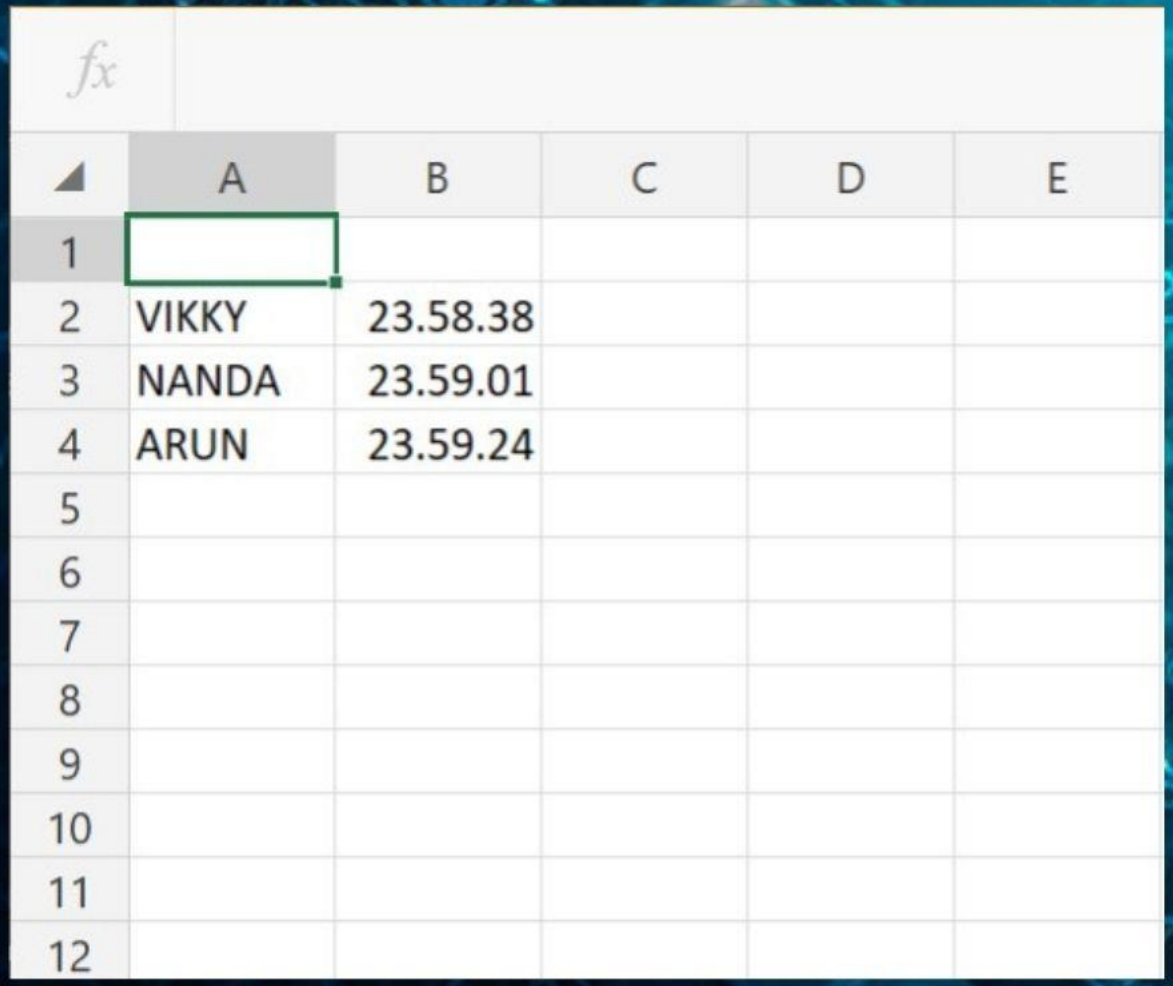
Uploaded Image

Detected using WebCam

Recognition of Unknown Faces



Entry of the Attendance in MS Excel



	A	B	C	D	E
1					
2	VIKKY	23.58.38			
3	NANDA	23.59.01			
4	ARUN	23.59.24			
5					
6					
7					
8					
9					
10					
11					
12					

Applications:

- It is used in Educational sectors to record the attendance of the students and the staff.
- It is used for security purpose.

Advantages:

- Facial recognition is a quick and efficient verification system.
- No Fuss In Workflow Management.
- Paperless Work Environment.
- Real-Time Tracking.
- Cost-Effective.
- Easy Access To Reports.

Conclusion

Thus Automatic Attendance System has been executed using Face Recognition.The system has been verified with the help of real time data samples.The system developed gives 85% accurate results, with some set conditions.


Thank You !