CU6051NP - Artificial Intelligence

Face Detection System

Prepared BY: Bidhan Pokhrel

Table of Contents

List of Figures:

1.Introduction:

1.1. Explanation of the topic:

My topic is about Face Detection System using python and it' open library. I have used open source library like open CV and Haar-cascade classifier. During this project, we are using the Face Detector algorithm called a Haar Cascade classifier. (Vision, n.d.) Haar Cascade classifier is an effective object detection approach which was proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001. (Khan, 2019). Also, the Full form of Open CV is Open Source Computer Vision. The Open CV is the Open source library for Computer Vision and machine learning software library. (CV, n.d.) The Open CV already contain many pre-trained classifiers for face, eyes, etc. We also can train the classifier on our own. The image can be obtained from the file or from the live video, which we will provide to the system for learning purpose and later on the system will detect the photo based on the given data. There are various applications of Open CV. Some of them are as follows:

a. Autonomous Vehicles.
b. Facial Recognition
c. Image Search and Object Recognition
d. Robotics

1.2. Introduction of the chosen problem topic:

Facial recognition is a computational technology that detects human faces in digital image . In a visual scene, face recognition also refers to the psychological mechanism by which humans identify and attend to faces. This is also the first stage of many face related technologies. Face detection system can be useful for any kind of people or organization as well. It can be used for security purpose. It can be used for detecting faces of the person in the mass by just using camera. The only thing that should be taken care of is the use of the software. It should be developed in the right way.

There are few points among that should be taken seriously while using face detection technology.

a. It should not be used for monitoring purposes that breach internationally agreed principles.

b. It needs to be equitable, because it does not exacerbate or intensify current prejudices, particularly when this could influence under-represented communities.

c. And it must preserve the privacy of citizens, giving the right level of transparency and control.

(AI, n.d.)

2. Background:

2.1. Research work done:

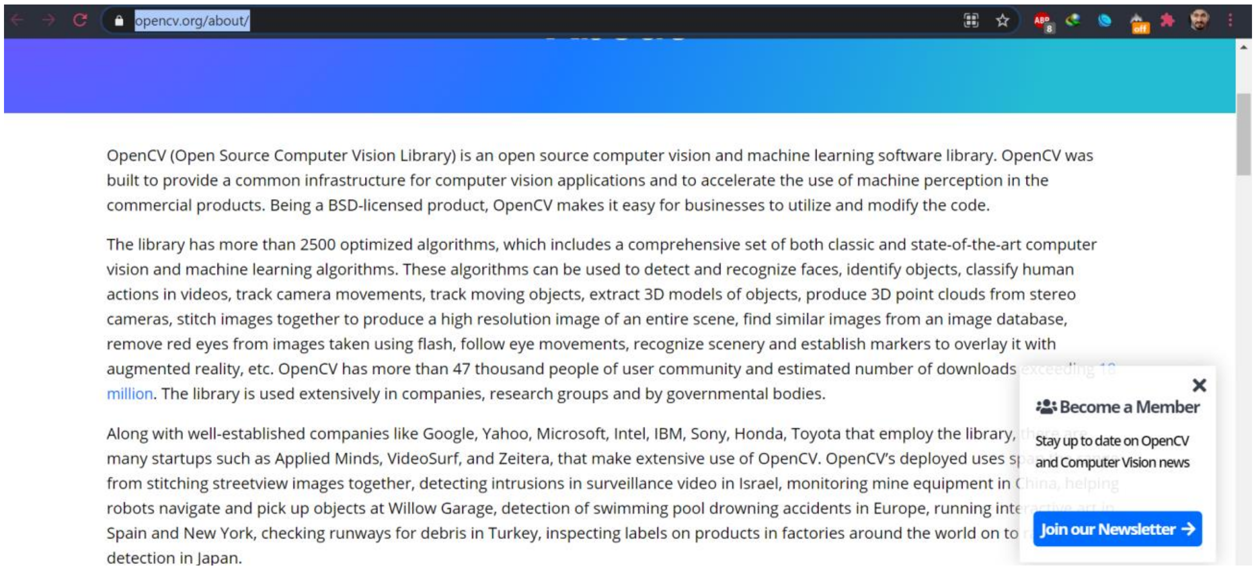
Open CV is the fast and convenient way that can be used in Face Recognition/detection System. One of the advantages of Open CV is its Open source library. The cascades which are in XML format can be used which can be downloaded from external source or one can make a new cascade with the help of open cv. More than 2500 optimized algorithms can be found in Open Cv. It has also multiple interfaces like C++, Python, Java and MATLAB. Which also supports various platforms like Windows, Linux, Android and Mac OS. The Open CV is extensively used by different well-known companies like Google, Yahoo, Microsoft, Intel, Sony, Honda, Toyota etc.. (CV, n.d.)

Sidra Mehta introduces new approach on face detection using Haar-Cascading. The purposed system is named as “Face Detection Using OpenCV and Haar Cascades Classifiers”

There is various analysis done on approach for obtaining Face detection system. Some of them are listed below.

a. Classical Face Recognition Algorithms.
b. Gabor Wavelets.
c. Face descriptor-based methods.
d. 3D based recognition.

Furthermore, talking about the classical face detection system, these techniques can fail to do so. Where faces are accurately portrayed when Wide differences in facial lighting Expressions and other variables emerge. Whereas, Gabor Wavelets is also an approach for face detection system. In this method, the main drawback is Substantially high dimensionality of Since the Gabor function room. The picture of the face is convoluted with a The Gabor Filters Bank. This is impractical for real-time application and computationally intensive. Another approach is face descriptor-based methods but this method is computationally intensive during descriptor extraction stage. The 3D-based face recognition system is also the new approach in this field, but it requires all the 3D element to be well calibrated and synchronized to existing 3D data. (inechopen, n.d.)

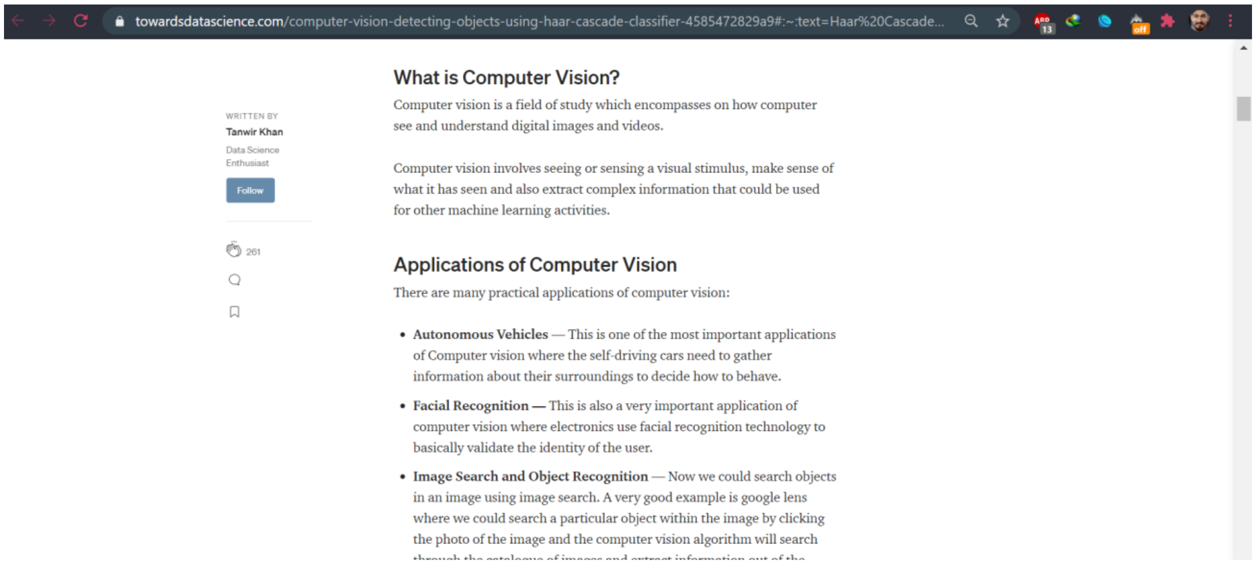

opencv.org/about/

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code.

The library has more than 2500 optimized algorithms, which includes a comprehensive set of both classic and state-of-the-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc. OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library is used extensively in companies, research groups and by governmental bodies.

Along with well-established companies like Google, Yahoo, Microsoft, Intel, IBM, Sony, Honda, Toyota that employ the library, th many startups such as Applied Minds, VideoSurf, and Zeitera, that make extensive use of OpenCV. OpenCV's deployed uses sp from stitching streetview images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects at Willow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to r detection in Japan.

Figure 1:Open CV


towardsdatascience.com/computer-vision-detecting-objects-using-haar-cascade-classifier-4585472829a9#:~:text=Haar%20Cascade...

WRITTEN BY
Tanwir Khan
Data Science Enthusiast

What is Computer Vision?

Computer vision is a field of study which encompasses on how computer see and understand digital images and videos.

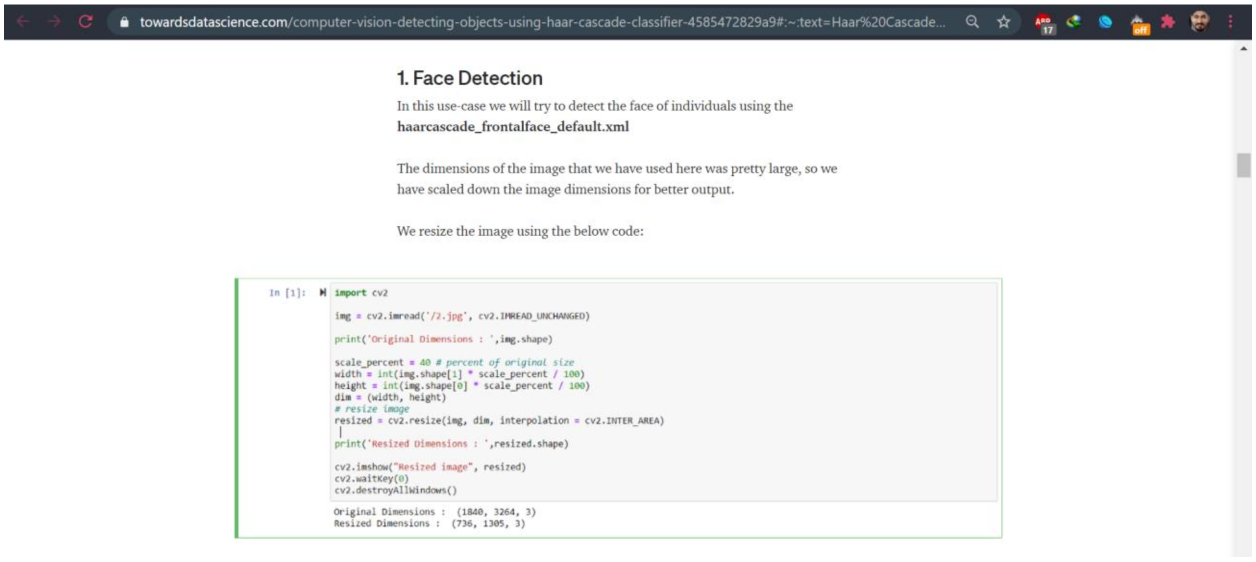
Computer vision involves seeing or sensing a visual stimulus, make sense of what it has seen and also extract complex information that could be used for other machine learning activities.

Applications of Computer Vision

There are many practical applications of computer vision:

- **Autonomous Vehicles** — This is one of the most important applications of Computer vision where the self-driving cars need to gather information about their surroundings to decide how to behave.
- **Facial Recognition —** This is also a very important application of computer vision where electronics use facial recognition technology to basically validate the identity of the user.
- **Image Search and Object Recognition** — Now we could search objects in an image using image search. A very good example is google lens where we could search a particular object within the image by clicking the photo of the image and the computer vision algorithm will search through the catalogue of images and extract information out of the

Figure 2: Uses of Open CV


towardsdatascience.com/computer-vision-detecting-objects-using-haar-cascade-classifier-4585472829a9#:~:text=Haar%20Cascade...

1. Face Detection

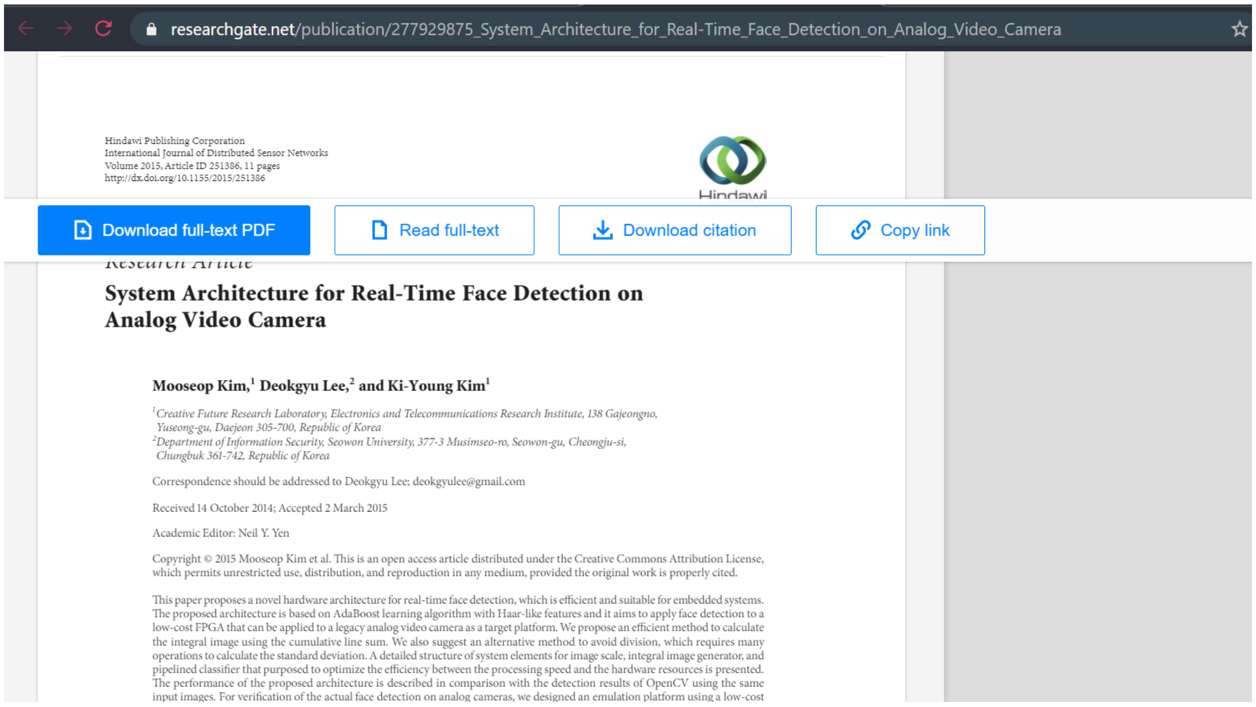
In this use-case we will try to detect the face of individuals using the **haarcascade_frontalface_default.xml**

The dimensions of the image that we have used here was pretty large, so we have scaled down the image dimensions for better output.

We resize the image using the below code:

```
In [1]:  import cv2

         img = cv2.imread('/2.jpg', cv2.IMREAD_UNCHANGED)

         print('Original Dimensions : ',img.shape)

         scale_percent = 40 # percent of original size
         width = int(img.shape[1] * scale_percent / 100)
         height = int(img.shape[0] * scale_percent / 100)
         dim = (width, height)
         # resize image
         resized = cv2.resize(img, dim, interpolation = cv2.INTER_AREA)

         print('Resized Dimensions : ',resized.shape)

         cv2.imshow("Resized image", resized)
         cv2.waitKey(0)
         cv2.destroyAllWindows()

         Original Dimensions :  (1840, 3264, 3)
         Resized Dimensions :  (736, 1305, 3)
```

Figure 3:Open CV and Haar-cascade integration


researchgate.net/publication/277929875_System_Architecture_for_Real-Time_Face_Detection_on_Analog_Video_Camera
Hindawi
Download full-text PDF
Read full-text
Download citation
Copy link

Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2015, Article ID 251386, 11 pages
http://dx.doi.org/10.1155/2015/251386

Research Article

System Architecture for Real-Time Face Detection on Analog Video Camera

Mooseop Kim,[1] Deokgyu Lee,[2] and Ki-Young Kim[1]

[1]*Creative Future Research Laboratory, Electronics and Telecommunications Research Institute, 138 Gajeongno, Yuseong-gu, Daejeon 305-700, Republic of Korea*
[2]*Department of Information Security, Seowon University, 377-3 Musimseo-ro, Seowon-gu, Cheongju-si, Chungbuk 361-742, Republic of Korea*

Correspondence should be addressed to Deokgyu Lee; deokgyulee@gmail.com

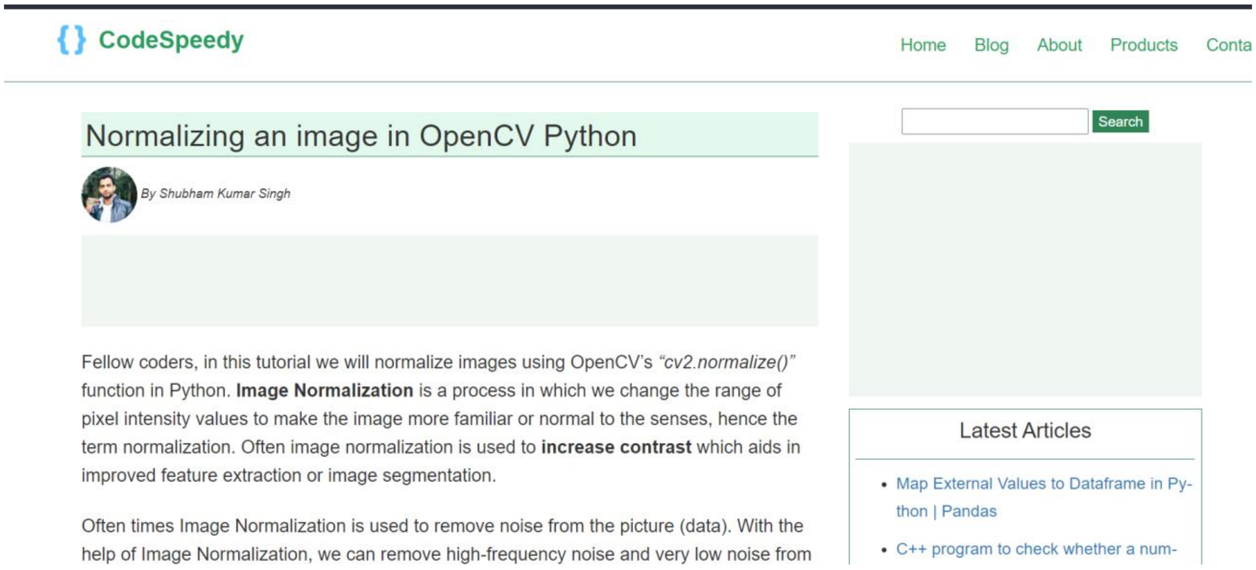
Received 14 October 2014; Accepted 2 March 2015

Academic Editor: Neil Y. Yen

Copyright © 2015 Mooseop Kim et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

This paper proposes a novel hardware architecture for real-time face detection, which is efficient and suitable for embedded systems. The proposed architecture is based on AdaBoost learning algorithm with Haar-like features and it aims to apply face detection to a low-cost FPGA that can be applied to a legacy analog video camera as a target platform. We propose an efficient method to calculate the integral image using the cumulative line sum. We also suggest an alternative method to avoid division, which requires many operations to calculate the standard deviation. A detailed structure of system elements for image scale, integral image generator, and pipelined classifier that purposed to optimize the efficiency between the processing speed and the hardware resources is presented. The performance of the proposed architecture is described in comparison with the detection results of OpenCV using the same input images. For verification of the actual face detection on analog cameras, we designed an emulation platform using a low-cost

Figure 4:Article about Face Detection using Haar-cascade algorithm



Normalizing an image in OpenCV Python

By Shubham Kumar Singh

Fellow coders, in this tutorial we will normalize images using OpenCV's *"cv2.normalize()"* function in Python. **Image Normalization** is a process in which we change the range of pixel intensity values to make the image more familiar or normal to the senses, hence the term normalization. Often image normalization is used to **increase contrast** which aids in improved feature extraction or image segmentation.

Often times Image Normalization is used to remove noise from the picture (data). With the help of Image Normalization, we can remove high-frequency noise and very low noise from

Latest Articles

- Map External Values to Dataframe in Python | Pandas
- C++ program to check whether a num-

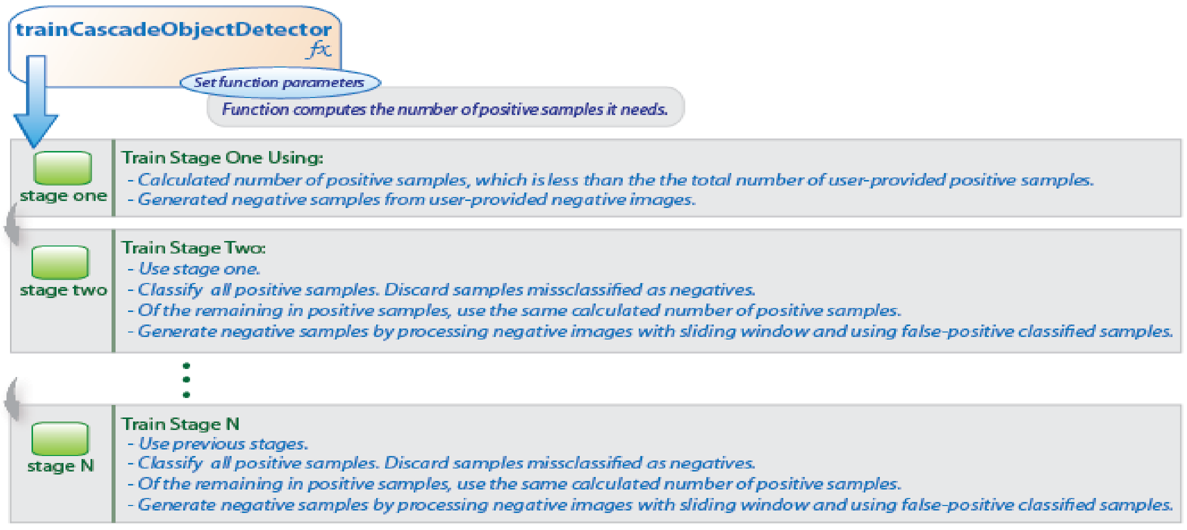
Figure 5:Normalization of image in open cv and python

3.Solution:

3.1. Explanation of the proposed solution/Approach to solving the problem:

The best effective way of solving the problem of face recognition system is using open cv. It has many features like it can read and write images, capture and save videos, Image processing such as filtering and transforming. Face Detection etc.

In my project, the program is made using open cv and Haar-cascade classifier. First, Open-CV and Haar-cascade file is imported in my program. Then the image is imported form a file or from video camera of laptop is turned on to take the input form the camera. Then the image/video is converted into grayscale image and by the help of Haar-cascade classifier the image is detected if it is a face or not. The program is terminated if the esc key is pressed in the keyboard.



Function computes the number of positive samples it needs.

Train Stage One Using:
- Calculated number of positive samples, which is less than the the total number of user-provided positive samples.
- Generated negative samples from user-provided negative images.

Train Stage Two:
- Use stage one.
- Classify all positive samples. Discard samples missclassified as negatives.
- Of the remaining in positive samples, use the same calculated number of positive samples.
- Generate negative samples by processing negative images with sliding window and using false-positive classified samples.

Train Stage N
- Use previous stages.
- Classify all positive samples. Discard samples missclassified as negatives.
- Of the remaining in positive samples, use the same calculated number of positive samples.
- Generate negative samples by processing negative images with sliding window and using false-positive classified samples.

Figure 6: Haar Cascade Trainning Algoithm

(Electronics, 2019)

3.2. Pseudocode of the solution:

For face detection in video:

```
Import open cv as cv2

Define face_cascade=cv2.cascaeClassifier('classifier location and cascade name.xml')

Cap=cv2.videoCapture(0) for importing video from live cam

While true:

Image=read capture image/video()

Grey=cv2.cvtcolor(image,cv2.color_Bgr2gray)

Faces=face cascade.detectmultiscale(gray,1.1,4)

For (X,Y,W,H) in faces:

Cv2.ractangle (img,(X,Y),(x+w,y+h),(255,0,0),2)

Cv2.imshow('image',image)

K=cv2.waitforkey[ress(30)&0xff

If key esc(0xff) is pressed:

Break operation.

Capture.realease()
```

For Face Detection in photo:

Import open cv as cv2

Define face_cascade=cv2.cascaeClassifier(‘classifier location and cascade name.xml’)

Image=cv2.imread(‘test.jpg’) for importing image file.

Grey=cv2.cvtcolor(image,cv2.color_Bgr2gray)

Faces=face cascade.detectmultiscale(gray,1.1,4)

For (X,Y,W,H) in faces:

Cv2.ractangle (img,(X,Y),(x+w,y+h),(255,0,0),2)

Cv2.imshow(‘image’,image)

Cv2.imshow(‘image’,img)

Cv2.waitkey()

3.3. Diagrammatical representations of the solution

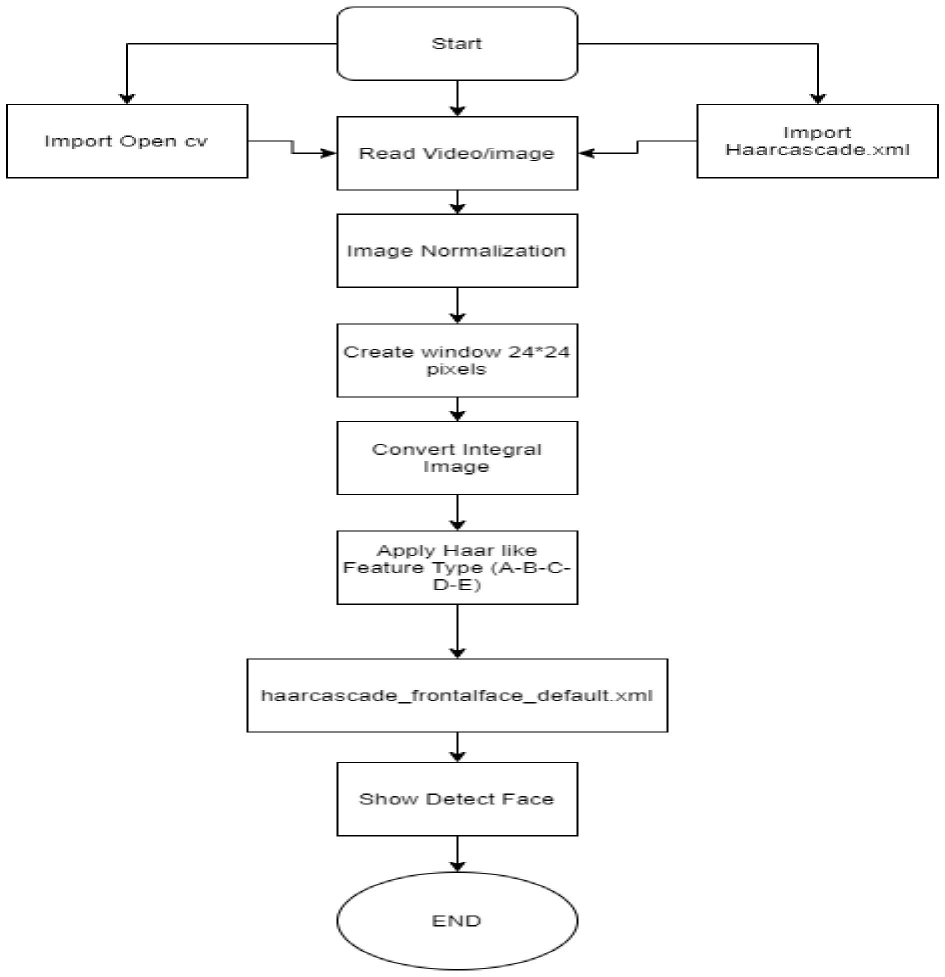


Figure 7:Flowchart

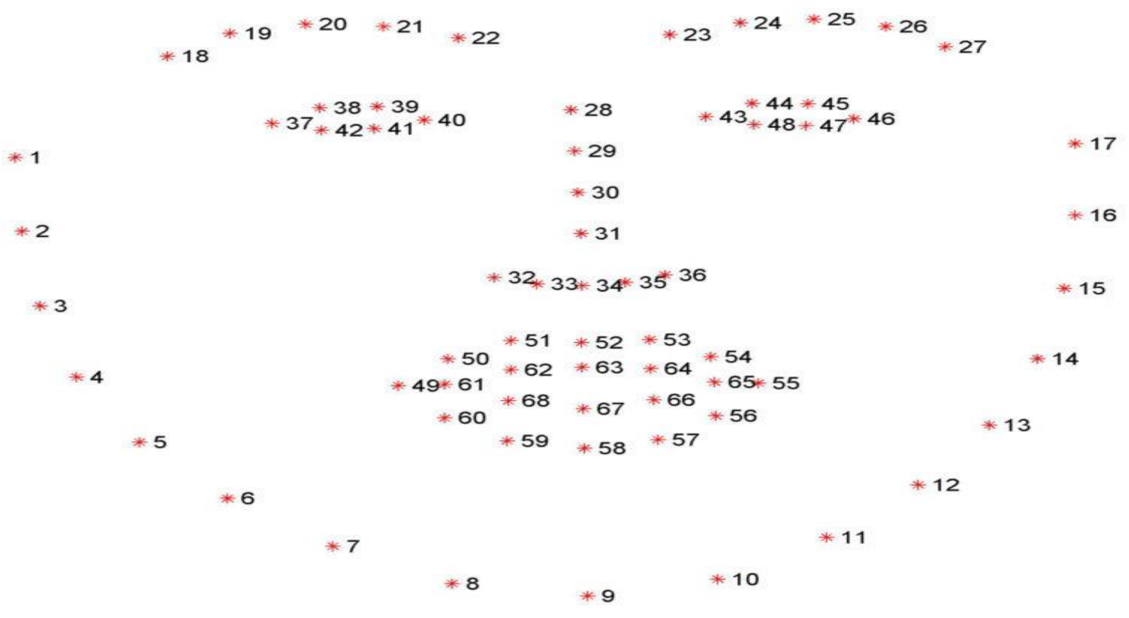


Figure 8: visualization of facial Coordinate

(Rosebrock, 2017)

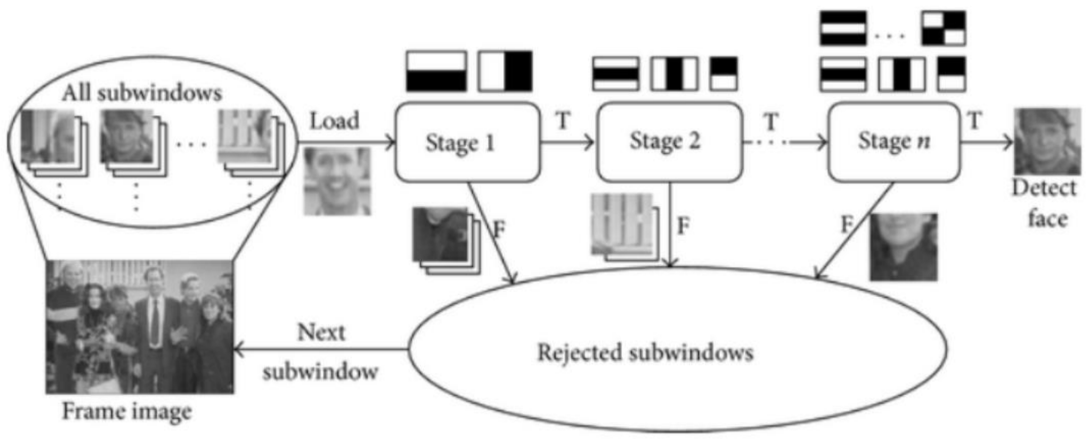


Figure 9: Cascade Structure for Haar Classifier

(Kim, et al., 2015)

3.4 Explanation of the development process

First, I imported the Open CV in my program. Then I imported the Haar-cascade file which is in the .xml format. Then the camera is turned on using the code. The input image/video is then converted into the grayscale format. Then the face is detected using the open cv and Haar-cascade. After that a rectangular box is created to show the facial part of the image. Then the image image/video is shown by keeping the rectangular shape around the face. The program is designed in such a way that the program is closed after pressing the esc key on the keyboard.

Various types of tools are used in this development.

PyCharm:

PyCharm is an integrated development environment(IDE) used for computer programing, especially for python language. PyCharm is cross-platform supporting various OS like Windows, Linux and macOS platforms.

Python: The program is written in the python language. Python is worlds mostly used program language. It contains a lot of libraries. Here in this program open-cv and harassed is imported to get the desired output.

Command Prompt:

Command Prompt also known as CMD is used to compile and run the Python code in the laptop. Python detect_face_video.py is the code that is used in the cmd to run the program. CMD is an program that is used to run various commands.

Open CV:

OpenCV is a cross-platform library using which we can develop real-time computer vision applications. It mainly focuses on image processing, video capture and analysis including features like face detection and object detection.

Haar-cascade:

Haar Cascade is a machine learning-based approach where a lot of positive and negative images are used to train the classifier.

Positive images- These images contain the images which we want our classifier to identify.

Negative Images – Images of everything else, which do not contain the object we want to detect.

In this project, Pre-trained Haar-cascade file is used to detect the face of the object/video.

3.5. Achieved Results:

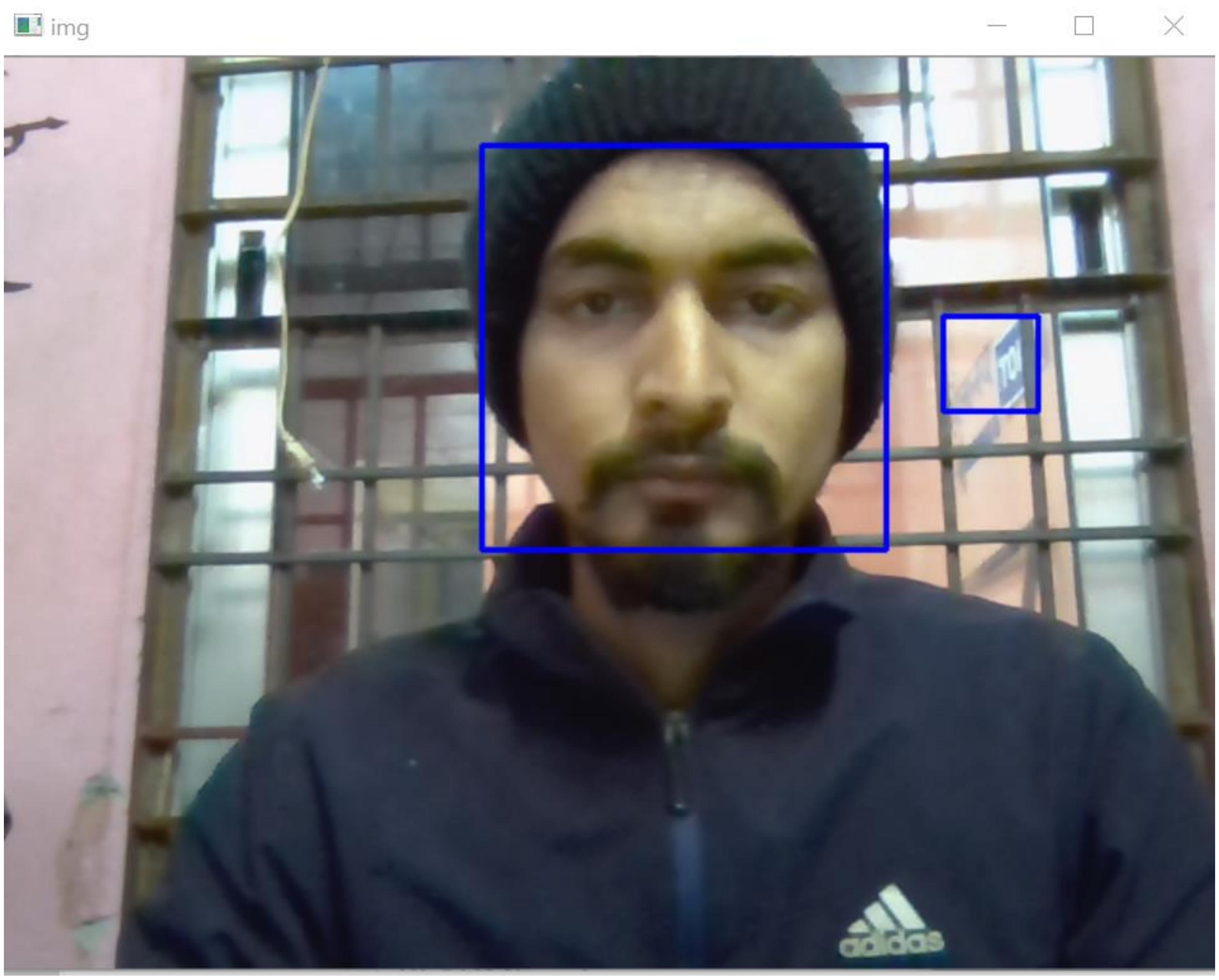


Figure 10:Single Face detection

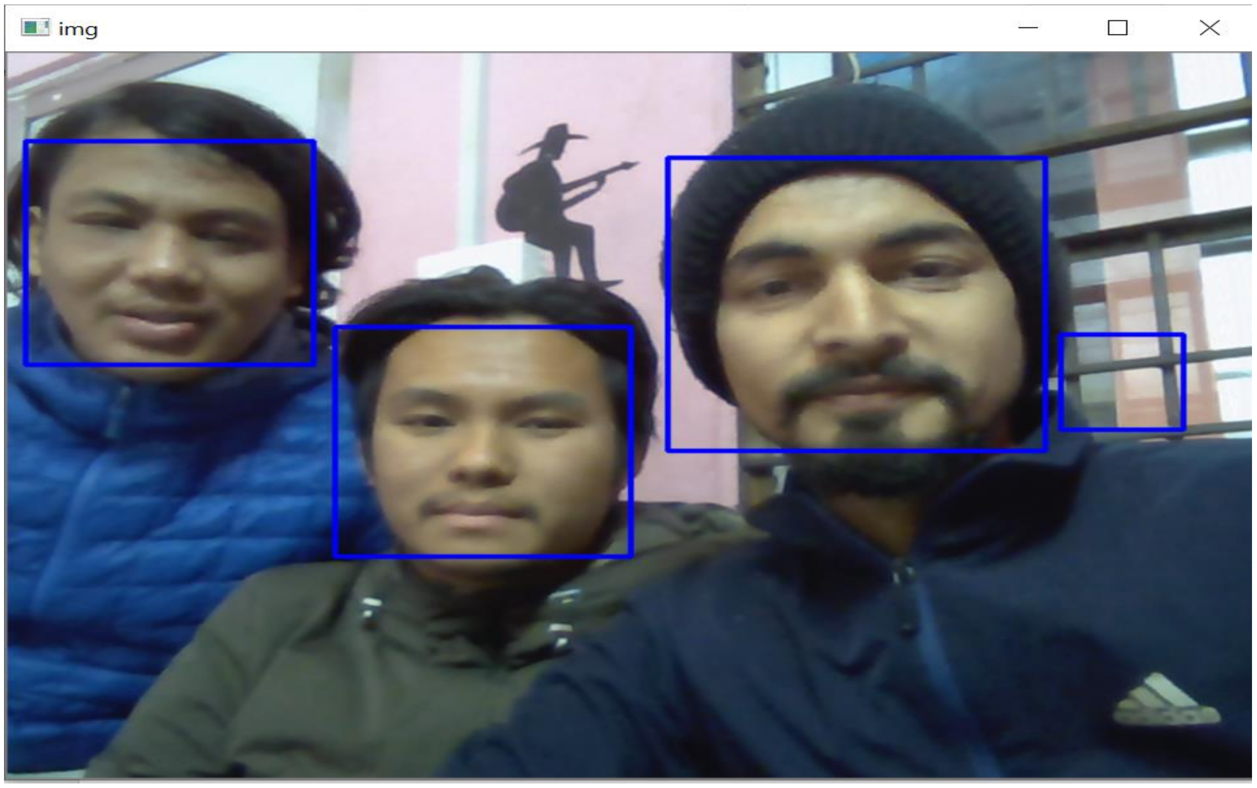
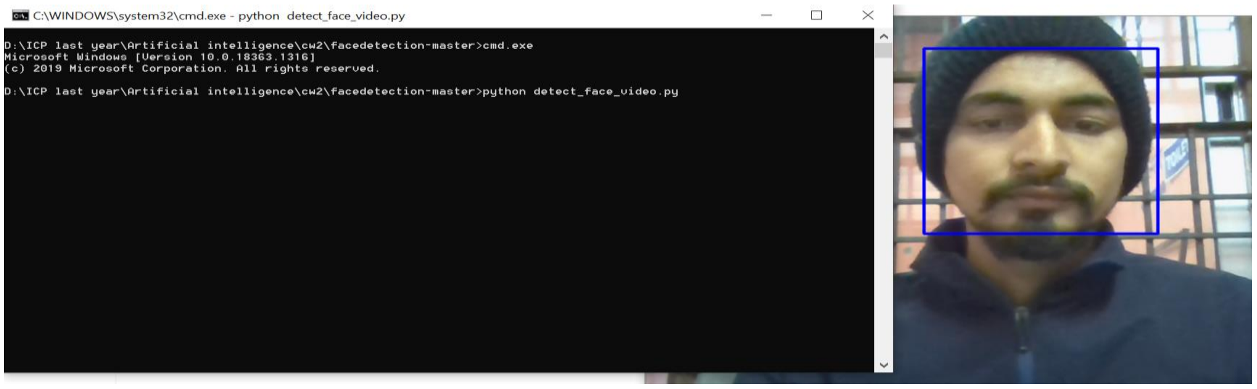


Figure 11:Multiple Face Detection

`



```
D:\ICP last year\Artificial intelligence\cw2\facedetection-master>cmd.exe
Microsoft Windows [Version 10.0.18363.1316]
(c) 2019 Microsoft Corporation. All rights reserved.

D:\ICP last year\Artificial intelligence\cw2\facedetection-master>python detect_face_video.py
```

Figure 12:Program execution using CMD

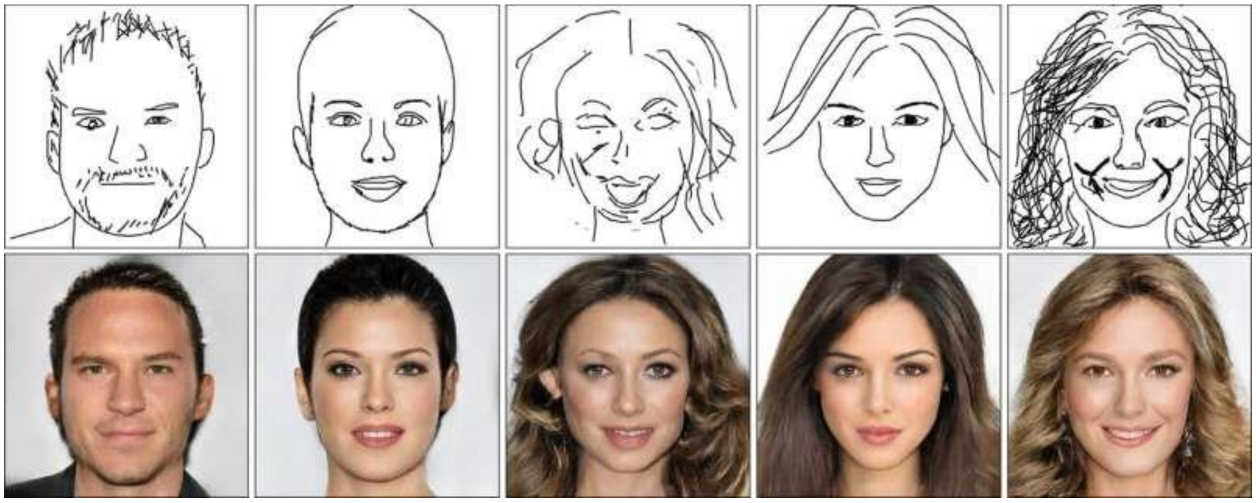


Figure 13:Sample image for Face Detection

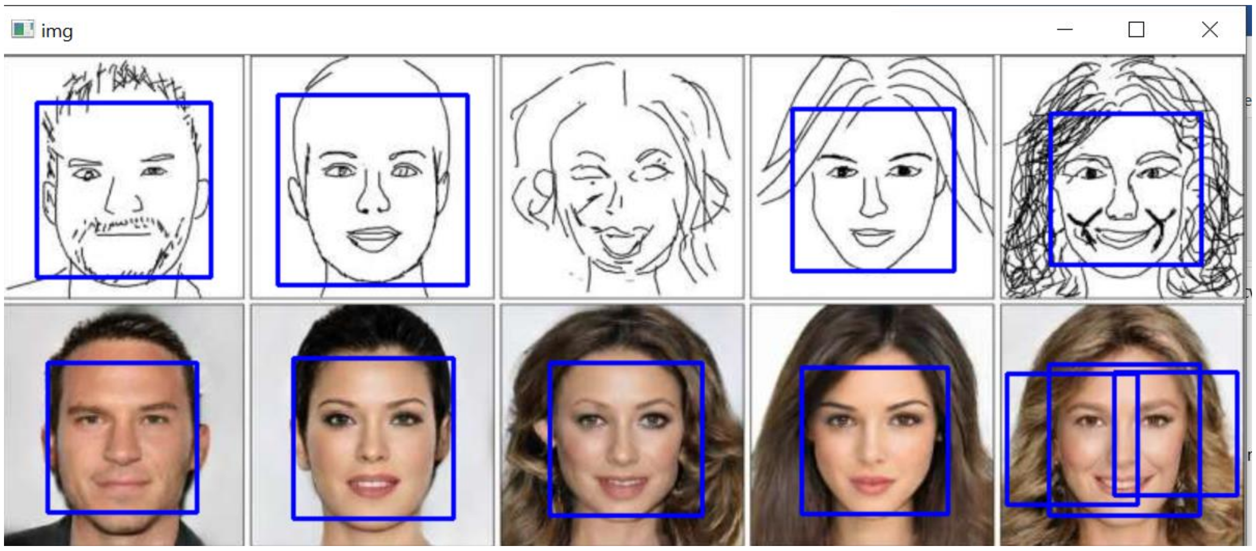


Figure 14: Image After face detection

4.Conclusion

4.1. Analysis of the work done:

A face detection system would certainly help in the surveillance whether the captured image is of human not. It will help in security purpose. With the help of this face detection system one can develop another system which will only capture the photo if any human face is detected. Now a day’s different types of software are in use. But the inbuilt library of Open-CV and the algorithm of Haar-Cascading will certainly boost up the speed of detection also will increase the accuracy of the detection.

4.2. How the solution addresses real world problems:

There are multiple face detection systems developed till now, but most of them are either not much accurate or they consume much more time to detect the faces. But this system is more accurate and takes less time to detect the faces. The algorithm used by other systems are quite slow in function because they consume more time to detect the non-face areas and when they detect it in first stage they keep processing further in those areas. But in case of Haar-cascade algorithm, it quickly detects the non-face areas and ignores it and directly jumps to the next part for the detection. And it further processes the facial part only for detection. Hence it can be the appropriate solution to overcome the real-world solution regarding the speed of detection and accuracy of detection.

4.3. Further work:

According to the given coursework, I did lots of research in the first coursework and I planned to develop the face detection system. As per the coursework, face detection system is developed, and the function of this system is shown in the above achieved result part. The program can detect human faces with high accuracy. Now I will be preparing for my presentation.

5. References

AI, G., n.d. *Our approach to facial recognition.* [Online]
Available at: https://ai.google/responsibilities/facial-recognition/

Brownlee, J., 2016. *Machine Learning Mastery.* [Online]
Available at: https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/
[Accessed 2021].

CV, O., n.d. *About.* [Online]
Available at: https://opencv.org/about/

Cv, O., n.d. *Face Detection using Open Cv.* [Online]
Available at: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html
[Accessed 2021].

Electronics, 2019. [Online]
[Accessed 2020].

inechopen, n.d. *Face Recognition: Issues, Methods and Alternative Applications.* [Online]
Available at: https://www.intechopen.com/books/face-recognition-semisupervised-classification-subspace-projection-and-evaluation-methods/face-recognition-issues-methods-and-alternative-applications

Khan, N., 2019. *Towards Data Science.* [Online]
Available at: https://towardsdatascience.com/computer-vision-detecting-objects-using-haar-cascade-classifier-4585472829a9#:~:text=Haar%20Cascade%20classifier%20is%20an,of%20Simple%20Features%E2%80%9D%20in%202001.&text=Based%20on%20the%20training%20it,objects%20in%20the
[Accessed 2020].

Kim, M., Lee, D. G. & Kim, K.-Y., 2015. *System Architecture for Real-Time Face Detection on Analog Video Camera.* [Online]

Available at: https://www.researchgate.net/publication/277929875_System_Architecture_for_Real-Time_Face_Detection_on_Analog_Video_Camera
[Accessed 2020].

OGla, R., 2017. *Research Gate.* [Online]
Available at: https://www.researchgate.net/figure/Flow-chart-the-open-CV-classifier_fig3_330171076
[Accessed Januarary 2021].

Rosebrock, A., 2017. *pyimage search.* [Online]
Available at: https://www.pyimagesearch.com/2017/04/10/detect-eyes-nose-lips-jaw-dlib-opencv-python/

Singh, S. K., n.d. *CodeSpeedy.* [Online]
Available at: https://www.codespeedy.com/normalizing-an-image-in-opencv-python/
[Accessed 2021].

Vision, O. S. C., n.d. *Open CV.* [Online]
Available at: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html

6.Code of the program:

Code for face detection from video.

```
import cv2

# Load the cascade
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# To capture video from webcam.
cap = cv2.VideoCapture(0)
# To use a video file as input
# cap = cv2.VideoCapture('filename.mp4')

while True:
    # Read the frame
    _, img = cap.read()

    # Convert to grayscale
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

    # Detect the faces
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)

    # Draw the rectangle around each face
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x+w, y+h), (255, 0, 0), 2)

    # Display
    cv2.imshow('img', img)

    # Stop if escape key is pressed
    k = cv2.waitKey(30) & 0xff
    if k==27:
        break
```

```
# Release the VideoCapture object
cap.release()
```

Code for face detection from image.

```
import cv2

# Load the cascade
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default.xml')

# Read the input image
img = cv2.imread('test.jpg')

# Convert into grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Detect faces
faces = face_cascade.detectMultiScale(gray, 1.1, 4)

# Draw rectangle around the faces
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x + w, y + h), (255, 0, 0), 2)

# Display the output
cv2.imshow('img', img)
cv2.waitKey()
```