Recent Developments in Electronics and Communication Systems
KVS Ramachandra Murthy et al. (Eds.)

© 2023 The authors and IOS Press.
This article is published online with Open Access by IOS Press and distributed under the terms of the Creative Commons Attribution Non-Commercial License 4.0 (CC BY-NC 4.0).
doi:10.3233/ATDE221325

Efficient Face Detection for Distinct Multimedia Using Object-Oriented Programming

Pranshu MISHRA[a.1] and Gaurav Kumar BHARTI[b.1]
[a]*Computer Science and Engineering, Chandigarh University, Mohali, India, 140413*
[b]*Electrical Engineering Department, Chandigarh University, Mohali, India, 140413*

Abstract. In this paper, face detection for multiple surveillance, and tracking has been proposed. This object-oriented programming has been used for data collection and other research fields. Software is designed to work with any computational device with a webcam and with python. With further development, it can also be used to fix recording as the camera can be automatically triggered to turn on and record the video until the person is visible. The proposed scheme has been verified using the Haar Cascade classifier and OpenCV platform. The different face detection platform for videos and cameras has been used to perform multiple tasks.

Keywords. Face Detection, Python, Multimedia, Object-Oriented Programming, Tkinter.

1. Introduction

With an amazing growth of media databases, the basic need for intuitive understanding and testing of data about wise systems as it becomes more obvious. Face plays an important role in social interaction to express specification and feelings. A living being cannot detect a face but a machine can. Therefore, face detection is important for face identification, and the position of the head. Face detection uses a set of four different templates to define unique features. Templates are used because they can be easily processed faster than other methods [1].

The template is mounted on a section of the image, and the weight is calculated based on the pixels below the template. Detecting the face is an important area of investigation in the field of science. Face detection is an applied science, which identifies human faces [2]. The position of a person's face is the first step in facial detection. For example, it can be used for attendance, surveillance purposes, etc. As MATLAB and OpenCV can be used to create such systems. In this paper, a distinct simulation platform has been discussed.

[1]Electrical Engineering Department, Chandigarh University, India, 140413
E-mail: gauravkumarbharti7@gmail.com

2. Theory- OPENCV VS MATLAB

As such, MATLAB is made-up of Java. Therefore, the MATLAB system starts working by translating the program and interpreting it into Java. Although, an OpenCV uses the functions of C / C++. Consequently, the machine language code is directly provided to the computer, which gives a faster performance. The rate ratio reaches more than 80 in the case of OpenCV [3]. Any system needs to compile an image, manage, and record key features for determining the position of the face. To recognize the seized image, the data of numerous properties have to be saved i.e., color, tone, etc.
A similar type of image can be collected for various purposes using facial detection/object detection [4]. Integrating a camera with any computational device, which is compatible to run the program can be used to detect faces. With further development, the program can be used to detect and calculate race, gender, and probable age. The proposed program works by integrating of Haar cascade classifier and OpenCV. Face recognition uses a photo from a webcam or video as embedded and extracts the title of the target image. Facial features can include facial contours and facial cuts. Face detection includes capturing features on the camera [5]. Face detection includes background removal and front focus removing any other features other than the face area [6].

3. Methodology

The OpenCV (Open-Source Computer Vision Library) is a virtual reality computer library, developed by Intel and free to use. The library is cross-platform, which focuses on real-time image processing. An OpenCV uses the functions of C or C++. Therefore, machine language code is directly provided to the computer, which gives faster performance. The advanced version of OpenCV (2.0) includes both C and C++ interfaces. Tkinter is a Python interface for Tk the most commonly used scripting language for testing, prototyping, and GUI development. Tk is an open-source widget kit, used in many programming languages to build GUI programs. Python uses Tkinter as a module. Tkinter collects C extensions that use Tcl / Tk libraries. Tkinter lets you upgrade desktop applications. It is an excellent GUI editing tool for Python. It is compatible with all Windows, macOS, and Linux applications. Pre-installed with standard Python library.

This conference paper has been produced with the clear objective of face detection using object detection techniques. The proposed face detection model is a computer vision integrated with object detection methods. To begin with, in this model, we need to input multimedia (image, video) [7]. Tkinter will input the image or video. Now read the input image. Now the image will change to Grayscale. The process of conversion in grayscale is done as earlier the images had 3 channels (R, G, B). After converting it to grayscale only 1 channel is left, which is again easier to computerize and process [8]. We also have to add the path of the Haar Cascade classifier of frontal face to detect faces. With the support of the Haar Cascade classifier, we detect the faces [9]. At last, we will detect the face and a rectangle will be drawn around the face. The proposed output has been displayed, as highlighted by a rectangle around the face.

Figure 1 shows the Observation of an object using OpenCV. To detect objects via OpenCV we will load the video file, we will segregate the video frame by frame, and execute object detection.

Figure 2 shows the conversion of RGB to grayscale. RGB images have 3 channels which creates a problem for the computer so we change it to grayscale which has 1 channel, which is easier to computerize and process.

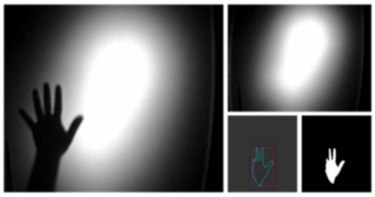


Figure 1. Object Observation using OpenCV

Figure 2. RGB image converted to grayscale

3.1 Methodology for face detection of the proposed work

Faces can be detected via different multimedia input. Here is a detailed flowchart for face detection via a pre-recorded video or live input using the webcam. When the video feed is received, it will be converted into grayscale. Then the Cascade comes into action and detects all the facial features. At last, the output is displayed on the screen with a rectangular box around the face. We all know videos are made up of frames, which are still images. Therefore, it will perform face detection for each frame in a video [11]. The following is a description of the methodology used to detect faces (Figure 3).

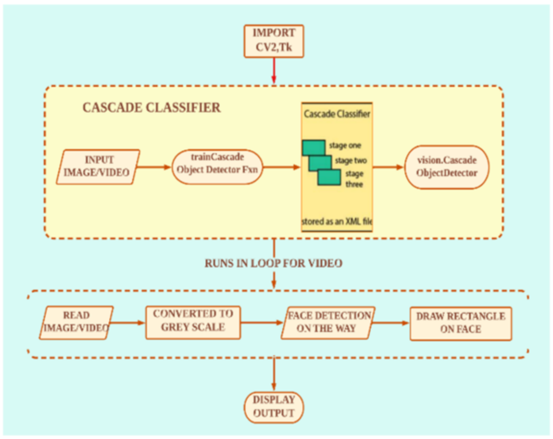
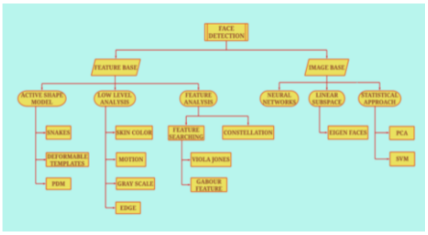


Figure 3. Working on Proposed System

4. Software and Hardware used

Operating system - windows 10
Processor – AMD Ryzen 7 4800H
GPU – NVidia RTX 3050
Memory - 8 GB
Compilers - Anaconda, VS Code
Scripting Language - Python 3.10
Camera – ASUS USB 2.0 Webcam
Libraries: OpenCV, Tkinter

Figure 4 is the classification of different types of face detection. Face detection can be performed with different techniques i.e., Feature base and image base [12].



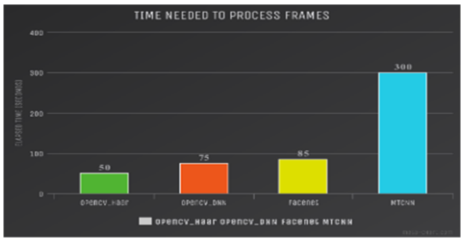


Figure 4. Classification of face detection **Figure 5.** The time needed to process frames by different algorithms

Figure 5 depicts the OpenCV_Haar algorithm that needed the shortest time to process video. On the other hand, MTCNN took the longest [13].
The main use of object detection is to estimate and detect a target from either a video or an image. It is used in image detection, counting, and face recognition. Object detection includes, Image processing, ML, AI, and Pattern Recognition [14-15].

5. Application of the software

Take a video input from the user using the webcam. Input can be live or recorded and use OpenCV and Haar Cascade to process the input. When a face is found in a video, the OpenCV library will provide a drawing process for face links.

- A Haar-cascade section is used to find the face.
- The frames of the input video are captured by the camera and are stored in temporary memory.
- To show the output a window is created using the OpenCV output function, which shows the live or recorded frames in motion while analyzing them in a live format.

6. Result and Analysis

Image or face caught by a webcam with the help of OpenCV has produced a rectangular classroom that keeps track of facial links. The OpenCV library example has been created and converts the image to grayscale. Load the cascade, at last, it will detect the face and, the output will be shown with a rectangle around the faces. The algorithm has been compared in Table 1, with the existing state-of-the-art.

Table 1. Different algorithm comparisons

Ref	Algorithm	Application of the work	Performance
6.	Haar Cascades	At first, it took time, but after the translation, it did not take long.	It is very well organized and used widely.

12.	Finding via motion	It takes time to detect a face.	It is less reliable than any other algorithm.
13.	Finding via color images	It takes a lot of time and is hard to execute.	Finds skin circle over the entire image
Proposed work	OpenCV_Haar with Tkinter	It is the fastest among all as we can input numerous multimedia with the help of Tkinter.	It is very well organized and used widely.

OpenCV_Haar with Tkinter, our model will be more efficient as we have introduced Tkinter, which will increase the rate of performance as we can input multimedia (image/video) easily. After inputting the image, Haar Cascade will analyze the facial features. The Haar Cascade is well organized and does not take time. This system is efficient because we can get results easily as multimedia input will be very fast.
Figure 6 is the result; the rectangle drawn around the face is desired output. With the help of OpenCV, we will get desired output with the help of different parameters such as imread, imshow, and cvtcolor.

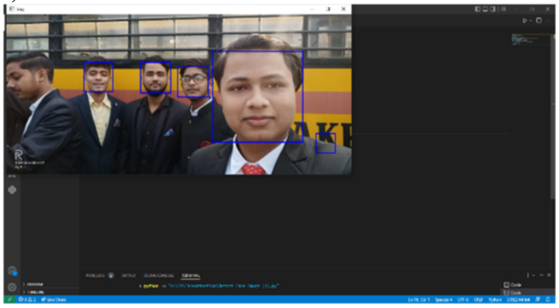


Figure 6. Result of face detection via input image

In this paper, we have discussed and compared various techniques for using face detection. It identifies Haar Cascades as the most effective way to get a face detected.

- The functions of Haar cascades provide better accuracy in facial expressions.
- Edge and drawbacks of OpenCV and MATLAB are discussed.
- Several algorithms are compared. and we found Haar cascade as the best method to get a face detected.
- Face detection working with future goals and references is discussed.

7. Conclusion

A face detection system has been proposed. Simulation results verified the proposed scheme for the detection of a person's face. The paper shows image processing, using OpenCV. The proposed scheme shows potential for the application in face recognition and mask detection. The proposed work can be used in numerous places whether it is a municipal station or any public place. By introducing a face-to-face program at every station and comparing the refugee ranks around the world, police departments may be able to control crime cases.

References

[1] Theocharides T, Link G, Vijay Krishnan N, Irwin MJ, Wolf W. Embedded hardware face detection. In17th International Conference on VLSI Design. Proceedings 2004 Jan 9 (pp. 133-138). IEEE.

[2] Zhou Y, Liu D, Huang T. Survey of face detection on low-quality images. In2018 13th IEEE international conference on automatic face & gesture recognition (FG 2018) 2018 May 15 (pp. 769-773). IEEE

[3] Zhang F, Fan X, Ai G, Song J, Qin Y, Wu J. Accurate face detection for high performance. ArXiv preprint arXiv: 1905.01585. 2019 May 5.

[4] Burghardt T, Calic J. Real-time face detection and tracking of animals. In2006 8th seminar on neural network applications in electrical engineering 2006 Sep 25 (pp. 27-32). IEEE.

[5] Zhu Y, Cai H, Zhang S, Wang C, Xiong Y. Tinaface: Strong but simple baseline for face detection. arXiv preprint arXiv:2011.13183 2020 Nov 26.

[6] Peer P, Solina F. Automatic Detection of Human faces in images.

[7] Lang L, Gu W. Study of face detection algorithm for the real-time face detection system. In2009 Second International Symposium on Electronic Commerce and Security 2009 May 22 (Vol. 2, pp. 129-132). IEEE.

[8] Liu Z, Qi X, Torr PH. Global texture enhancement for fake face detection in the wild. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition 2020 (pp. 8060-8069).

[9] Jones M, Viola P. Fast multi-view face detection. Mitsubishi Electric Research Lab TR-20003-96. 2003 Jul 15; 3 (14):2.

[10] Wu H, Chen Q, Yachida M. Face detection from color images using a fuzzy pattern matching method. IEEE transactions on pattern analysis and machine intelligence. 1999 Jun; 21 (6):557-63.

[11] Yow KC, Cipolla R. Feature-based human face detection. Image and vision computing. 1997 Sep 1 15 (9):713-35.

[12] Kumar A, Kaur A, Kumar M. Face detection techniques: a review. Artificial Intelligence Review 2019 Aug 5 2(2):927-48.

[13] Hsu RL, Abdel-Mottaleb M, Jain AK. Face detection in color images. IEEE transactions on pattern analysis and machine intelligence 2002 Aug 7 24(5):696-706.

[14] Jiang H, Learned-Miller E. Face detection with the faster R-CNN. In2017 12th IEEE international conference on automatic face & gesture recognition (FG 2017) 2017 May 30 (pp. 650-657). IEEE.

[15] Yang S, Luo P, Loy CC, Tang X. Wider face: A face detection benchmark. In Proceedings of the IEEE conference on computer vision and pattern recognition 2016 (pp. 5525-5533).