FACIAL RECOGNITION USING OPENCV

[1]Akshay Pratap Singh, [2]Raj Vikram
[1,2]B.tech Student
[1,2]CSE Department, Lovely Professional University

Abstract: The strong demand in the last decade's image recognition. Driven by the gradually doubling numerical pace every 13 months, visual recognition has transcended an enigmatic region to a common machine vision science field and one of the best and most effective implementations in image processing and algorithm-based comprehension. Thanks to its fundamental existence, image processing is not just a study area in computer engineering, but also a focus of the neuroscience and characteristics based, primarily because of the popular belief, that advancement in digital picture analysis and cognitive learning brings us perspectives into how our brain operates and vise - versa.
The authors propose making a request for entry by users to a specific computer focused on a detailed examination of an individual's face characteristics, considering the general interest and concern in this subject. Intel Open Source Vision Initiative, OpenCV and Microsoft .NET Framework are used with this framework as well.

Key-Words: *face, detection, recognition, system, OpenCV, Eigenface*

1. INTRODUCTION

The purpose of this article is to confront between the person and the computer simpler when identification by face detection and recognition of the individual is required. A computer will identify and recognize a person's face using a standard web camera. A personalized login display must be created with the potential to restrict customer access using users' facial characteristics.
The goal of this study is to include a series of detection methods which can afterwards be packed inside simple to understand context between the different hardware platforms that are currently used in devices (computers). Such algorithms will have a good detection score of at least 95%, fewer than 3% of the faces identified being misleading.

2. PROBLEM DEFINITION

Over most of the past decade the identification of faces and recognition pervaded theoretical to mainstream areas in image processing science as one of the biggest and fastest implementations in the study of photographs and algorithms . Thanks to the inherent existence of the issue, image processing is often the subject of neuro-scientific, psychological and computer engineering study, primarily because of the common belief that advancement in digital image processing and study in perception provides visibility into how our brain functions and vise - versa.

The following is a general description of a face-recognition issue (in computer vision): provided silent or video representations of the environment, one or more figures in the environment may be recognized or checked by utilizing a recorded facet database. Two steps are normally needed for facial recognition
Face identification when a picture was looked for to identify a object, the image is then manipulated and the object of the individual is removed to make it easy to see. Face Recognition where the face identified and analyzed is linked to a recognized facet database in order to determine who the person is.

The OpenCV[1] framework of Intel has been able to do face recognition fairly quickly and accurately since 2002. The system features a built-in facial scanner, which operates in about 90-95 percent of a person's direct images. Detecting a faces in an image from an angle is typically challenging, so often 3D head location calculations are needed. In addition, the failure to detect a face or increase the contrast in shadows on your face can boost a great deal or perhaps the image is blurred or the user is wearing sunglasses, etc.

Yet facial identification is far less accurate than image recognition, with an average performance of 30-70%. Facial recognition has been a popular research area since the 1990s, but is still a long way from a secure app authentication tool. Each year, even more

strategies are being created. The Significance approach is known as the best way to correctly identify the face, although many other (much more complex) approaches or variations of multiple methods are much easier.

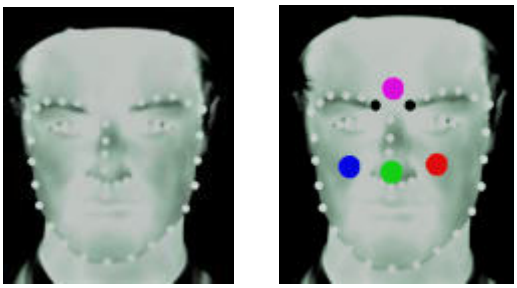

Fig 1. Retrained and Original Thermal Images

Gary Bradski launched OpenCV on Intel in 1999 to speed up the world's scientific and industrial implementations in computer vision and to create competition for Intel's increasingly stronger computers. Vadim Pisarevsky joined Gary to oversee the OpenCV department of Intel Russian applications. Multiple businesses and scholars were transferred over period to the OpenCV team. Half of the original team actually operated in robotics and moved to Willow Workshop. In 2008, Willow Garage see a need to quickly advance its robotic perception capability, exploiting the science community and the business sector and beginning to aggressively promote OpenCV.

The machine-view library of Intel, open access, will render computer programming considerably easier. It contains sophisticated technologies, such as facial identification, facial monitoring, image recognition and Kalman screening, and a range of ready-touse technologies of artificial intelligence. Additionally, via its lower-level API, it offers some simple computer vision algorithms.

OpenCV This serves that both Windows and Macos, and most recently, Mac OS X as a multiple platforms system**.** OpenCV has so many apps that it might at first feel intimidating. A strong knowledge of the way certain systems function is important if OpenCV wants to produce successful outcomes. Luckily, only a few must be learned in advance to continue.

Within many units, OpenCV software is included, which would be used for facial recognition. The following is a concise overview of main namespace:

the **CXCORE** namespace includes simple descriptions of the data form, linear algebra and methods of analysis, perseverance and error handling. Only the software features for painting pictures are oddly positioned here.

CV Name Space requires sensor configuration and image processing processes. There are also machine geometry features.

CVAUX Name space shall be defined as having outdated and innovative code in the OpenCV documents.Nevertheless, this module provides the shortest interfaces for facial recognition. The coding between them is for facial recognition focusing and is commonly used for this reason.
ML The name space includes frameworks for machine learning.
High GUI Namespace includes the standard I / O interfaces and window functionality for the different platforms.
CVCAM Namspace includes 32-bit Windows device video control interfaces via DirectX.

Eigenfaces are viewed to be the easiest technique for exact face recognition but several other (more difficult) methods or multi-method configurations are a little more precise.
Most facial recognition tools are for simple neural networks that typically don't function as do the characteristics. And sadly certain simple explantations are only possible on the side of the face recognition and the Successful Apparition Models for improved typology of facial recognition. But for other methods, you can read recent CVPR and other computer vision articles. Many computer vision and machine vision workshops provide recent developments in facial recognition and facial performance.

3. Proposed Solution

There are numerous factors that affect the accuracy of the model when picture quality is taken into account.
Different image pre-processing methods are highly essential to standardize the pictures you send to a facial recognition program. Many facial recognition algorithms are especially susceptible to lighting environments, and it's definitely not identifiable in a bright environment because they are qualified for identifying a human while they are in a dark space, etc.

This concern is related to as "lumination based" and several other problems often emerge, such as face, which should always be in a very clear picture location (e.g. eyes are in one pixel coordine), clear height, angle of rotation, hair and make-up, expression (smiling, furious, etc). This is why the implementation of facial recognition is so necessary to use a strong image preprocessing filters. You can also do stuff like remove some pixels across the face that are not needed to display the inner image, not the hair and photo context, for example for the elliptical mask, since they alter rather than the image.

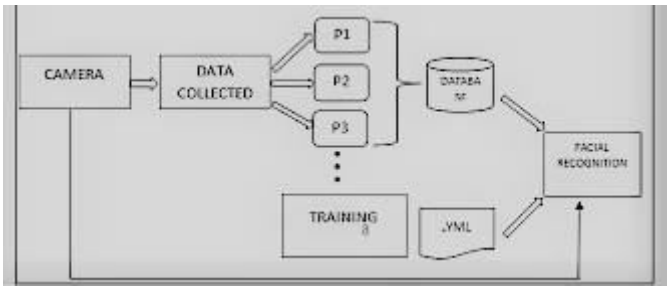

Fig 2. Block Diagram

For simplification, Eigenfaces utilizing gray images are thefacial recognition device used in this paper. The paper illustrates how simple it is to transform color graphics to grayscale (also known as "white scale"), and then use Histogram Equalization [8] to optimize the face photos' luminosity and contrast. You may either utilize image face recognition (ideally with HSV appropriate color histogram or a different color space rather than RGB), or implement additional steps, e.g. edge enhancement and contour identification, motion detection, etc. For optimal performance. This technology often resizes pictures to a normal scale, but the aspect ratio of the face may be changed. A method of resizing an picture with the same appearance ratio is defined .

OpenCV uses the Hair Cascade classifier for a form of face detector. Provided a picture that may originate from a filename or from live footage, the face detector analyses either photo position and categorizes it as "Face" or "Not Face".When the faces in an picture become smaller or bigger, the classificator is constantly running through the picture to look for faces across a number of stages.

It might sound like an enormous processing number, but the classification is quite simple, albeit performed on multiple scales due to algorithmic stuff discussed in the sidebar. The classifier utilizes data contained in an XML file to decide whether each image position is categorized. The installation of OpenCV contains four XML styles for facial frontal and profile images. There are also three non-face XML scripts, one for body recognition (pedestrian) and one for upper body, and one for lower body.

To define the information file you need to use, you would have to say the classifier. The version that I would use is *haarcascade_frontalface_default.xml*. This is [OPENCV ROOT]/data / haarcascades / haa rascade frontalface default.xml in OpenCV Version 1.0, where [OPENCV ROOT] is the route to install OpenCV. You will use [OPENCV ROOT] = "C:/Program Files / OpenCV," if you operated on a 16-bit older Windows Version you will use the directory separator "\" For eg, if on Windows XP you chose your application default place, Rather than '/'.)

It is a smart thing to find your XML file before you cod the rest of your face-sensing software and to insure your direction to that is right. It's very convenient to use a camera stream instead of a file list as an entry into the facial recognition program. Essentially, only frames are to be taken from a camera and not a script, so you can run indefinitely before the user finishes instead of merely running until the script list is ended. Tap sets can be conveniently deployed with this feature from a Webcam:
/ Take its next frame. Wait before you're ready for the next frame and / offers clear access to it, but don't change or free the picture returned! / The image on the first frame must auto-initialize.

```
IplImage* getCameraFrame(CvCapture*
    &camera)
    {
     IplImage *frame;
     int w, h;

//If you have not started your phone, open it.
if (!camera) {
printf("Acessing the camera ...\n");
camera = cvCreateCameraCapture( 0 );
if (!camera) {
printf("Couldn't access the camera.\n");
exit(1);
}
    // TRY SETTING THE CAMERA RESOLUTION320*240.
        cvSetCaptureProperty(camera,
       CV_CAP_PROP_FRAME_WIDTH,320);
        cvSetCaptureProperty(camera,
      CV_CAP_PROP_FRAME_HEIGHT,240);
```

```
// For the first frame for the initialization of
the camera.
frame = cvQueryFrame( camera );
if (frame) {
w = frame->width;
h = frame->height;
printf("Got the camera at %dx%d
resolution.\n", w, h);
}
// Wait for it to automatically change the
luminosity.
Sleep(1000); // (in milliseconds)
}

// Wait for the next frame of the image, then
pick up.
frame = cvQueryFrame( camera );

if (!frame) {
printf("Couldn't     grab     a      camera
frame.\n");
exit(1); } return
frame;
}
```

This feature may be also use like this :

```
CvCapture* camera = 0;      // The
camera device.

// Quit on "Escape" key.
while ( cvWaitKey(10) != 27 ) {
IplImage            *frame            =
getCameraFrame(camera);

…

}
// Free the camera.
cvReleaseCapture( &camera );
```

4.Conclusion

There are many aspects which can be changed in order to boost identification efficiency, some of which are relatively simple to enforce. You can add color care, edge detection, etc, for example.
Normally, you can increase the accuracy of face recognition by taking nearly 50 more photographs each user, specifically from different angles and light conditions, with more pictures. When you can not take further portraits, there are many easy methods to produce further training pictures, making fresh photos from the old ones: you can make clones of the face pictures of double as many of them and do not have a left or right orientation. You should subtly convert or rotat the face photos to generate other different training pictures, such that they become less sensitive to particular conditions.
You may attach image noise for more training pictures to improve noise tolerance. In order to enable the classifier to identify the person in various lighting conditions and roles, instead of searching for similar circumstances, a lot of different circumstances would be needed for any entity.But it is also important to insure that a variety of images are not so different for a individual , for example if certain images are rotated by 90 degrees. This renders the classifier too general and often provides incredibly poor performance, so you may generate different sets of training pictures for each individual if you know that you would have a collection of photos with a very wide variance (say, rotation more than 20 degrees).

For starters, you could learn to recognize a classifier "Facing John" and "Facing Forward" and "Elizabeth Facing Forward" and "John Facing Left" "Mary Facing Left" and so on. Every classification algorithm will then be somewhat specific, but not too specific, so you just have to mix each entity with the same person in the various classifiers (i.e.:'Jesus' or 'Mary').
That is why, if you do not use proper pre - processing on your photos. You also get really poor outcomes. Like I said earlier, the contrast enhancement is a basic picture preprocessing tool and in certain scenarios may render matters worse, so you would definitely have to mix many various methods before you obtain satisfactory results.
At the root of the method is that pictures are compared by basically the inverse of the test picture being subtracted to a reference image to see how close they are. This would be very interesting if a person was doing it, but just in pixels and percentages the machine feels. When you can assume that the gray scale value of the pixel will be subtracted by the pixel value within each training picture at the same EXACT spot, and less the discrepancy so the greater the match will be.
And whether you simply transfer an picture across a several pixels, or use a picture that is only a couple of pixels smaller, or that displays a few of pixels of your forehead, then you're going to believe it's completely special! It often happens if the backdrop is different, as the code does not recognize the distinction

between the context and foreground (face), and that it is necessary to eliminate the backdrop as far as possible, for instance by having just a tiny segment of the face that does not include any context.

Since the pictures are nearly precisely matched, in general this also results in greater visibility than big high-resolution photos of tiny images (like shrinking thumbnails to the scale of photographs)! Even, particularly though the pictures have been matched

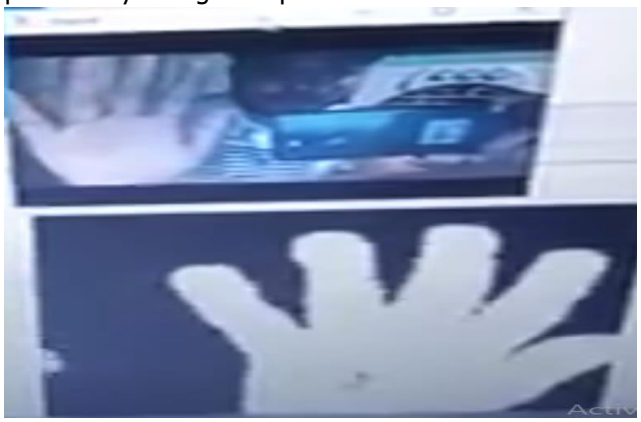

Fig 3. Face Detection and Object Detection

correctly, if the evaluation image is a little higher than the testing imagery, it always looks a little bit strange. Proposed method will in certain situations aid, but it will render it worse in many situations, so lighting variations are a complicated and frequent issue. There are often things with a shadow in the learning picture on the left and the shade on the ground, and it always does not function properly, etc.

Therefore, face recognition is fairly simple to do in live time as you teach someone and then attempt to identify them directly afterwards, because the image is the same because the context is the same, their faces are nearly similar and the setting is the same. And at the moment you would also receive positive outcomes of appreciation. Yet when you seek to remember them 0from another place, whether from another space or outside or at another time of day, it always yields bad results! So it's necessary to conduct a lot more research if you want better outcomes because if you still can't produce successful results after you've done several items, you'll probably need a more complex face recognitions algorithm than PCA (Eigenfaces).

5. References

[1] .Open Source Computer Vision Library Reference Manual-intel [Media]

[2]https://docs.opencv.org/3.4/d7/d8b/tutorial_py_face_d etection.html

[3] M.A.Turk and A.P. Pentaland, "Face Recognition Using Eigenfaces", IEEE conf. on Computer Vision and Pattern Recognition, pp. 586-591, 1991

[4] Face Recognition Homepage, http://www.face-rec.org/algorithms/

[5] Computer Vision Papers, http://www.cvpapers.com/

[6] L.H. Liang, H.ZH. Ai & G.Y. Xu (2002), "A Survey of Human Face Detection," J.Computers. China, Vol. 25, Pp. 1– 10

[7] Hussein Rady (2011), "Face Recognition using Principle .

[8] https://www.semanticscholar.org/paper/One-to-many-face-recognition-with-bilinear-CNNs-Chowdhury-Lin

[9]https://www.pyimagesearch.com/2018/09/24/opencv-facerecognition/

[10] https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b

[11] M.A.Turk and A.P. Pentaland, "Face Recognition Using Eigenfaces", IEEE conf. on Computer Vision and Pattern Recognition, pp. 586-591, 1991

[12] Abdullah, M., Wazzan, M., & Bo-saeed, S. (2012, March). Optimizing face recognition using PCA. *International Journal of Artificial Intelligence & Applications, 3*(2), 23-31.

[13] Barnouti, N. H. (2016, May). Face Recognition using PCA-BPNN with DCT Implemented on Face94 and Grimace Databases. *International Journal of Computer Applications, 142*(6), 8-13.