Raspberry Pi and Computers-Based Face Detection and Recognition System

Ayman A. Wazwaz
Electrical Engineering Department
Palestine Polytechnic University PPU
Hebron, Palestine
e-mail: aymanw@ppu.edu

Amir O. Herbawi, Mohammad J. Teeti, Sajed Y. Hmeed
Electronics and telecommunication engineering
Hebron, Palestine
e-mail: Df.w@hotmail.com, mohammadteeti@gmail.com, 125055@ppu.edu.ps

Abstract—**This paper aims to deploy a network that consists a group of computers connected with a microcomputer with a camera. The system takes images of people, analyze, detect and recognize human faces using image processing algorithms. The system can serve as a security system in public places like Malls, Universities, and airports. It can detect and recognize a human face in different situations and scenarios.**
This system implements "Boosted Cascade of Simple Features algorithm" to detect human faces. "Local Binary Pattern algorithm" to recognize these faces. Raspberry Pi is the main component connected to a camera for image capturing. All needed programs were written in python. Tests and performance analysis were done to verify the efficiency of this system.

Keywords-face detection, face recognition, raspberry Pi, Python, OpenCv

I. Introduction

Computer-based face detection and recognition systems are rapidly spreading is various sectors such as malls, universities, and ministries.

The goal of this research is to build a system that can detect and recognize faces of people using image-processing techniques. Practically, this idea can be implemented in large places to provide security.

The benefits of this system are:

1) Use available computers (servers) in corporation with microcomputers.

2) Expand the desired microcomputer capabilities.

3) Implement the face detection and recognition algorithms to run over the microcomputer to notice the effect of combining multiple computers with a microcontroller.

II. Related Work

In [1], a face detection system using Raspberry Pi was developed. Authors did not approach face recognition implementation due to complexity of the recognition process, since recognition process would need more powerful resources to accomplish better results. In [2], the authors were able to switch from the closed-circuit television CCTV graphical processor to a computer graphical processing unit GPU, and embed the security cameras into the computer GPU. In [3], authors attempted to detect faces in a digital image using various techniques such as skin color segmentation, morphological processing, template matching, Fisher linear discriminant (FLD), and Eigen face decomposition [7].

III. Research Methodology and Solution

The purpose of this paper is mainly to optimize the face detection and recognition speed. The optimization is conducted using multiple servers approach to do the recognition process, which could take a long time due to the complexity of the recognition process, the database size and the quality of the processed faces.

Using multiple servers – three in this scenario- will allow a faster recognition for a stream of faces, instead of working among the same microcomputer to detect and recognize the stream of faces, which implies to divide the process in-between the different electronics available to reduce the processing time for both detection and recognition.

IV. Hardware Implementation

The block diagram in Figure 1 consists of a camera that will capture and forward frames to the Raspberry Pi, the Raspberry Pi then will detect and crop the faces in this frame and transmit the resulting face image over the network to one of the available computers to recognize the received faces and display the results.

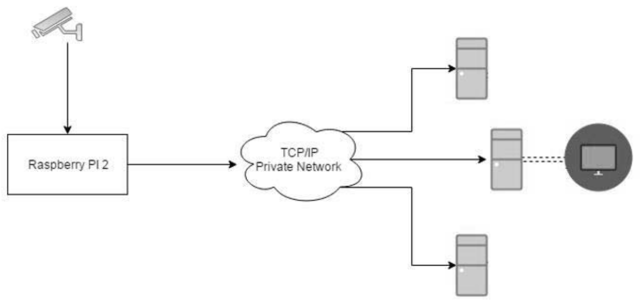

Figure 1. General System Block Diagram.

The system works as follows:

- A camera is connected to the Raspberry Pi will stream live video.
- Faces of selected people should be detected, and cropped to transmit them to one of the connected personal computer over the networks. This is done by implementing detection algorithm on the Raspberry Pi.

- The connection between the Raspberry Pi and the computers is TCP/IP based over Ethernet connection.
- The selected computer receives the cropped face image and forward it to the "face recognizer algorithm" to recognize the face stored in the database.
- If the face is unknown, and the system cannot recognize it, a notification message would appear over the computer screen or send elsewhere.

V. Software Implementation

The software implementation is summarized in the following:

- The Raspberry Pi will be programmed to stream video. While each frame is being streamed the Boosted cascade of simple features BCOSF [4] algorithm will try to detect the faces in this frame. If any appears, the program which is Python-based will crop the face, saves it on the Raspberry Pi and immediately forward it to the available server.
- The servers are all Linux-based. Whenever an image arrives, the server will initiate local binary pattern histograms LBPH [5] algorithm on this face, equalize the facial image to reduce the variations and compare the resulted LBPH from this face to the pre-saved LBPH in the database.
- The servers are familiar with the known faces stored in the database. A stage prior to the system initialization will run each known face into the LBPH algorithm to generate the LBP values, used in the previous step.
- The detection algorithm stated earlier, will take care of framing the live video, detect the face, crop it and forward it to the available computers.
- The recognition algorithm which will be implemented on the computers (servers) will receive the cropped picture from the detection phase and process it using the LBPH algorithm which will compare each detected face with the faces in the database.
- The external database contains the known faces and names. If this database contains many images for the same person, it will recognize it with a better correlation percentage, but will consume more time to process and recognize.

VI. System Implementation

To measure the performance, multiple tests were conducted to test the system capabilities in order to improve its detection/recognition rates.

The first test addressed the first component of the system; the Camera. The test determined several parameters related to the camera, these parameters were used in the face detection algorithm to help detect people faces with the least percentage of error.

The test investigates three main parameters of the captured frame, these parameters are:

1. **Windows Size** :the size of the rectangle drawn around the detected area and its divided into :
   - Minimum Size
   - Maximum Size

   Window size will decide the minimum detection distance after which, no detection is valid.
2. **Scale Factor (SF)**: Parameter specifies how much the image size is reduced at each image scale.
3. **Minimum Neighbors (MN)**: The minimum number of applied windows that each candidate rectangle should contain. The test will run with the following conditions:
   - The camera will take a **100 image** (faces and wrong faces are counted) on each run.
   - Random movement will be in front of the camera, some people are closer and some are further away.
   - On each camera run, one of the previous parameters (Window size, SF, MN) was under test to observe the number of wrongly detected faces.
   - Window size (min | max) will decide the distance, the **First** window size: (20*20 | 300*300) is capable of 4 meters distance, **Second** window (50*50 | 250*250) can detect 3.7 meters while the **Third** window (80*80 | 200*200) detects 3 meters away.

The following results were then obtained for the three window sizes:

TABLE I. Detection Parameters Test (SF 1.3, MN 6)

Window	percentage of false (1)	percentage of false (2)	percentage of false (3)	Average %
First	21	36	27	28%
Second	24	40	35	33%
Third	52	38	43	44.33%

TABLE II. Detection Parameters Test (SF 1.2, MN 6)

Window	percentage of false (1)	percentage of false (2)	percentage of false (3)	Average %
First	13	20	23	18.66%
Second	26	19	19	21.33%
Third	16	14	23	17.66%

TABLE III. Detection Parameters Test (SF 1.1, MN 6)

Window	percentage of false (1)	percentage of false (2)	percentage of false (3)	Average %
First	31	33	20	28%
Second	12	15	12	13%
Third	31	39	39	36.33%

Tables I, II, and III show that when SF is 1.3 the percentage of error is relatively high, when the SF is reduced to be 1.2 the percentage of error decreases. However, when SF is reduced to 1.1 there was an increment in the error at (20*20 | 300*300) window size as well as an increment of error at (50*50 | 250*250), but it's clear that the error decreased when the window size (80*80 | 200*200) is used.

In the following Tables IV to VI the same conditions applies, but a different MN was chosen.

TABLE IV. DETECTION PARAMETERS TEST (SF 1.3, MN 5)

Window	percentage of false (1)	percentage of false (2)	percentage of false (3)	Average %
First	29	22	17	22.67%
Second	24	27	22	24.33%
Third	67	58	70	65%

TABLE V. DETECTION PARAMETERS TEST (SF 1.2, MN 5)

Window	percentage of false (1)	percentage of false (2)	percentage of false (3)	Average %
First	63	19	25	35.67%
Second	24	17	21	20.67%
Third	36	40	46	40.67%

TABLE VI. DETECTION PARAMETERS TEST (SF 1.1, MN 5)

Window	percentage of false (1)	percentage of false (2)	percentage of false (3)	Average %
First	15	17	14	15.33%
Second	21	8	17	15.33%
Third	49	31	40	40%

After decreasing MN to 5, tables show the results for each SF value. The error is fluctuates between 20% to 65% in the Tables IV, and V which is considered a relatively high error percentage.

When the SF decreased to 1.1, the error dramatically decreases to about 15% for window size ranges (20*20 | 300*300) and (50*50 | 250*250). At the same time, high percentage (about 40%) still exists for window size range (80*80 | 200*200).

It is worthy mentioning that the above results were taken in daylight in a normal sunny day. Different light conditions would cause different results especially for the far faces.

Previous results were conducted with three reading each time, the following tests works with window size of (20*20 | 300*300), but this time the SF were varied from 1.14 to 1.3 with a 0.02 step each time.

Table VII provides the SF readings versus number of false detections for MN 4.

TABLE VII. DETECTION PARAMETERS TEST VARYING SF (MN 4).

Scale Factor	percentage of false (1)	percentage of false (2)	percentage of false (3)	Average %
1.14	10	8	7	8.33
1.16	7	10	8	8.33
1.18	9	6	10	8.33
1.2	8	12	8	9.33
1.23	10	11	10	10.33
1.25	10	12	12	11.33
1.27	11	13	16	13.33
1.3	13	16	16	15

It is noted that when the scale factor increases, the average error rate increase, because increasing scale factor will decrease the quality of the image and consequently increase the error rate.

To measure latency, a test was applied to determine the speed of the file transfer between Raspberry Pi and the servers using socket programming. The number of transferred files is determined and then a relationship is obtained between number of receiving servers and number of files transferred.

Tables VIII and IX clarify the results obtained, note that the time is a conventional term that corresponds to speed, in the tables below.

Table VIII shows the delay when one server is connected to the Raspberry Pi for processing, with a fast Ethernet connection between them.

TABLE VIII. TRANSMISSION TIME TOWARDS ONE SERVER

Number of images sent	Transmission delay (seconds)
10	0.042
50	0.225
100	0.4
200	0.868
500	2.4
750	3.43
1000	4.56

Increasing the number of servers to three produced the following transmission delay shown in Table IX.

TABLE IX. TRANSMISSION TIME TOWARDS THREE SERVERS

Number of images sent	Transmission time (seconds)
10	0.0723
50	0.2029
100	0.3266
200	0.8699
500	1.434
750	1.94
1000	2.21

It is noted that an increase in the number of images will not produce a significant increase in transmission time since images are small in size and the speed of the network is high (100 Mbps). And increasing number of receiving nodes will speed up the transmission at large number of images compared with one receiver, since one receiver will receive incoming images using one network card and cause congestion and delay.

The system efficiency increase when the number of servers' increases, we can investigate that by testing the required time to recognize varying number of images for one server, comparing these results with three servers to compare the effects of multi-servers approach.

Processing Time (Tp) is the time required to process all images as a function of images number versus the number of servers. Processing time includes face recognition for all images sent to the servers wither one or three. The following points were taken into consideration while conducting the test:

1) The recognition database contains one hundred names.
2) Each person has varying number of faces, some users have three images, and others have twenty-five images.
3) The test images group consist of 1000 images.

Tables X and XI show the results of the recognition processing time on one and three servers.

TABLE X. PROCESSING TIME ON ONE SERVER

Images number	**Processing time (seconds)**
10	3.73
100	42.29
500	202
1000	413.11

TABLE XI. PROCESSING TIME ON THREE SERVERS

Images number	**Processing time (seconds)**
10	0.81
100	14.41
500	96.97
1000	126.6

It is noted that adding two more servers would speedup processing, and reduce the delay especially with large number of images. Three servers will give faster face recognition as shown in Figure 2.

Figure 2 was generated using MATLAB to visualize the effect of adding two extra servers.

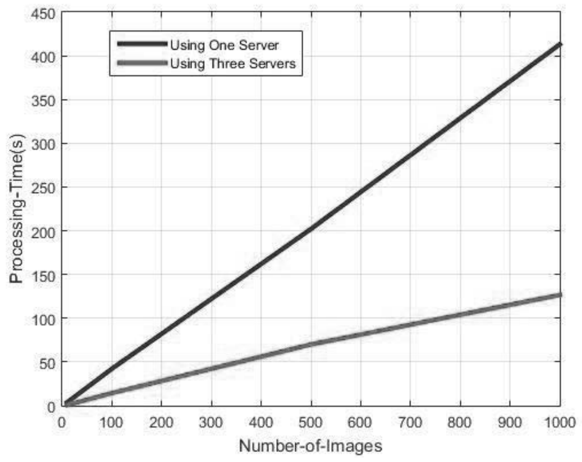

Figure 2. Recognition Processing Time (one server vs. three servers)

The curves were fitted with MATLAB fitting tool; the following equations were derived for the required time to recognize faces. Here, TP is Time of Processing and N is the number of images sent from the Raspberry Pi to the servers in the network.

<u>For one server</u>

- $\boldsymbol{Tp = 0.4122 \times N - 0.611}$

<u>For three servers</u>

- $\boldsymbol{Tp = 0.1271 \times N + 1.794}$

VII. CONCLUISONS

A distributed system was designed to process images from capturing, until face recognition using one microcomputer and number of supporting computers.

The possibility of utilization computers to support the microcomputer in face recognition would speed up processing, but the cost is to distribute the database on different machines, and the security between these machines should be considered.

Increasing window size increased the error rate. When the scale factor increases, the average error rate increase, because image size reduction will decrease the quality of the image and consequently increase the error rate. Also, more images for the same person will enhance the correlation rate. Results vary depending on lighting, distance, and the resolution of the camera. Adding more computer as recognition servers will speed up processing especially for large number of images compared with one server.

REFERENCES

[1] Swathi. V, Steven Fernandes," Raspberry Pi Based Human Face Detection", published in International of advanced research in computer and communication engineering Vol. 4, Issue 9, September 2015

[2] Víctor Bautista Saiz; Fernan Gallego, GPU: Application for CCTV systems, Published in: Security Technology (ICCST), 2014 International Carnahan Conference.

[3] Ping Hsin Lee, Vivek Srinivasan, and Arvind Sundararajan. Face Detection, Final Year Project, Stanford University, 2014

[4] Face Detection using Haar Feature-based Cascade Classifiers in OpenCV. Online technical documentation. Available at https://docs.opencv.org/3.3.0/d7/d8b/tutorial_py_face_detection.html

[5] Local Binary Pattern Histograms in OpenCV (Open Source Computer Vision Library) Online technical documentation. Available at https://docs.opencv.org/2.4/modules/contrib/doc/facerec/facerec_tutorial.html#local-binary-patterns-histograms-in-opencv.

[6] Detect Multi Scale Function in OpenCV (Open Source Computer Vision Library). Online technical documentation. Available at https://docs.opencv.org/trunk/d1/de5/classcv_1_1CascadeClassifier.html

[7] T. Shakunaga , and K. Shigenari," Decomposed Eigen-Face for face recognition under various lighting conditions" . IEEE *Computer Society Conference on Computer Vision and Pattern Recognition*, 2001.