Nigerian Research Journal of Engineering and Environmental Sciences 7(1) 2022 pp. 281-287
p ISSN: 2635-3342; e ISSN: 2635-3350

Nigerian Research Journal of Engineering and Environmental Sciences
Journal homepage: www.rjees.com

Original Research Article

Design and Implementation of a Face Recognition System Using ZEDboard

***Bello, N. and Uwuigbe, E.O.**

Department of Electrical/Electronic Engineering, Faculty of Engineering, University of Benin, PMB 1154, Benin City, Nigeria.
*nosabello@uniben.edu; emmanuel.uwuigbe@uniben.edu

http://doi.org/10.5281/zenodo.6725774

ARTICLE INFORMATION

Article history:
Received 23 Sep 2021
Revised 08 Dec 2021
Accepted 12 Dec 2021
Available online 30 Jun, 2022

Keywords:
OpenCV
Face recognition
Xillinuz
Haarcascade algorithm
Local binary pattern
ZEDboard

ABSTRACT

This paper presents the design and implementation of a face recognition system on a ZEDboard embedded system. The study utilized a Xilinx-zynq-z7020 system-on-a-chip (SoC) in the ZEDboard which ran the Xillinus operating system (OS) to design a facial recognition system using OpenCV libraries that will recognize images in real time from a video stream fed in through a Logitech USB webcam. It was observed that the accuracy of the identification improved when the pose variations used in the database were moderate and the environment was appropriately illuminated

.

1. INTRODUCTION

The expanding use of computer vision in replacing human surveillance motivated the research in the field of face detection (Barnouti et al., 2016). The performance of different faces-based applications, from standard face recognition and verification to the latest face clustering, retrieval and tagging, depends on efficient and accurate face detection. Face detection is among the important advanced topics in computer vision and pattern recognition communities (Barnouti et al., 2016) and it is the first important step for facial analysis. Face recognition system can be used in two modes: verification and identification. Face verification system (one-to-one matching) involves confirming or denying the identity claimed by a specific individual. Face identification system (one-to-many matching) attempts to find the identity of a given individual against all image templates in face individual database (Jafri and Arabnia, 2009).

Face recognition methods can be divided into appearance-based or model-based methods. Appearance-based (Holistic) face recognition legally attempt to identify faces using global representations based on the entire image rather than local facial features. Appearance-based methods can be classified as either linear or non-linear. Linear appearance-based methods perform a linear dimension reduction (Fernandes and Bala, 2013). An image is considered as a high dimensional vector and in this method, the face vectors are projected to the basis vectors and the projection coefficient is used as the feature representation for each face image. Linear methods include principal component analysis (PCA), linear discriminant analysis (LDA), independent component analysis (ICA). Non-linear appearance-based methods are usually more complicated than linear methods. Direct non-linear manifold schemes are explored to learn this non-linear manifold. On the other hand, model-based face recognition schemes aim to construct a model of the human face that can capture facial variations. It can be either 2- or 3-Dimensional. These models are frequently morphable. Morphable model make it possible to classify faces even when pose changes are present. Model-Based methods include elastic bunch graph matching (EBGM) or 3D morphable models (Gayathri and Valarmathy, 2014). Also, there is the hybrid method which is a combination of appearance-based and model-based methods. Face recognition involves recognizing individuals with their intrinsic facial characteristic. Compared to other biometrics, face recognition is more natural, non-intrusive and can be used without the cooperation of the individual which thus makes it valuable towards security.

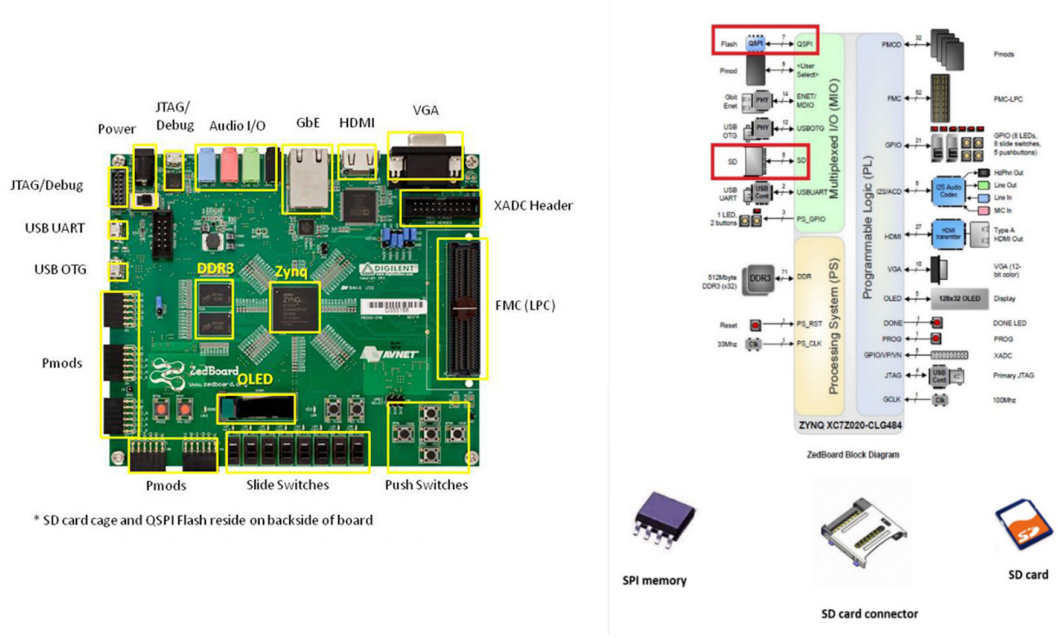
Studies have investigated ways to improve the approach for face detection by using OpenCV, Viola–Jones algorithm based on coding eyes which removes the falsely detected faces (Hossen et al., 2017). OpenCV has an implementation of the Haar Feature-based cascade classifiers proposed by Viola and Jones (2001). It is a machine learning based approach in which a cascade function is trained from a lot of positive and negative images. A training group of these positive and negative images forms the input to the classifier which needs to extract the best features out of a large set of possible features (6000+ features) using AdaBoost algorithm (OpenCV team, 2017; OpenCV team, 2021). The detection is done using a cascade of classifiers with the best features stored as XML files. Barnouti et al. (2016) proposed an automatic face recognition system using appearance-based features that focus on the entire face image rather than local facial features. The feature extraction and dimension reduction method applied were the principal component analysis method combined with square Euclidean distance (SED) to obtain image similarity. The implementation was simulated with MATLAB. Also, Ismael and Irina (2020) proposed a face recognition system that was implemented in a smart building for security. The system used a stream of pictures or video feed to recognizes the person according to Viola–Jones object detection framework developed using Python. The proposed software system was developed using OpenCV and proposed an efficient algorithm for detecting and recognizing a human face from live video by applying the Viola–Jones algorithm depending on Python. However, no information was discussed about identifying more than one human face in a video. Ouanan et al. (2015) described the implementation and optimization of Viola Jones face detection framework using OpenCV on Devkit8500, which is a low power, open-source single-board computer. The study presented the development of a face recognition system using OpenCV in a low-cost development board, ZEDboard for the Xilinx Zynq-7000 all programmable system-in-a-chip (SoC).

In this study, a face recognition system was designed and implemented using the Xilinx Zynq-7000 SoC which is a new kind of device which combines an FPGA fabric with a capable application processor (dual core ARM cortex-A9) which will carry out image matching processing between the images already stored on the security database and the real time images taken by the surveillance cameras or the video cameras.

2. MATERIALS AND METHODS

The setup of this study made use of primarily, a ZEDboard, a monitor, keyboard, mouse and Logitech USB webcam, of which the computer had Xilinux, OpenCV-2.7.9, python and other dependencies installed. The ZEDboard as shown in Figure 1 is a developmental board used for working with the Xilinx zynq7020, an all programmable SOC (system on chip). This product integrates a feature-rich dual-core ARMCortex-A9 MPCore based processing system (PS) and Xilinx programmable logic (PL) which contains 85,000 series-7

programmable logic cells in a single device. Unlike the several other development boards, the ZEDboard combines on-board peripherals and expansion capabilities which makes it an ideal and widely used platform.



(a) (b)

Figure 1: (a) The ZEDboard (b) system representation of the ZEDboard

The setup required interfaces of the ZEDboard through the Ethernet port, an RJ-45 cable and SD card slot. These peripherals will be highlighted to present relevant information. The ZEDboard implements a 10/100/1000 Ethernet port for network connection using a Marvell 88E1518 PHY. This part operates at 1.8 V. The RJ-45 connector is a TE Connectivity 1840750-7 featuring integrated magnetics. There is an SD slot fixed under the ZEDboard. It controls communication with the ZEDboard. A SD card of size, 8 GB is included in the ZEDboard kit. Just like the QUAD SPI flash it can be used for non-volatile storage of external files and also for booting of the zynq-7020 AP SoC after its programming or modification by the files stored on the SD card thus eliminating the need for wired connections for programming. The board is powered via a barrel jack and has a voltage specification of 12 V at 1.3 A which is suitable for powering the peripheral components and the ZEDboard itself. In this study, the peripheral components utilized are outlined as follows:

i) Power consumed by the mouse at 100mA and 5V was obtained as follows:

$$Power = Current \times Voltage = 100\,mA \times 5\,V = 0.5\,W$$

ii) Power consumed by the keyboard is also 0.5 W maximum

The power requirement of the complete setup is completely satisfied by the rated 12 V power supply adapter but, however in the case where more peripherals are to be connected that could overload the USB port on the ZEDboard, an externally-powered USB hub is used.

The Logitech USB webcam used features HD video capture up to 1280×720 pixels, uses Logitech fluid crystal ™ technology, with autofocus, and up to 8 megapixels picture resolution (software enhanced). It also has a built-in mic with Logitech right sound ™ technology and a Hi-Speed USB 2.0 certified. Xillinux is an embedded operating system for use in embedded system applications (Xillybus, 2017). It is a complete, graphical, Ubuntu 12.04 LTS-based Linux distribution for the Zynq-7000 EPP device, intended as a platform for rapid development of mixed software/logic projects. The currently supported boards are ZEDboard,

MicroZed and Zybo. In the setup of this study, a guided procedure was followed in the development setup for the successful implementation of the aim of the research which are summarized as follows:

i. Setup development of the ZEDboard with Xillinux which comprises the generation of the boot files, copying the files to an SD card, and the configuration of ZEDboard to boot from SD card.

ii. Setup of an active connection between the Logitech HD720 webcam and the ZEDboard

iii. Set up of a file transfer protocol (FTP) connection over the RJ45 cable for connection between ZEDboard and workstation (computer).

iv. Installation of the OpenCV library and the dependencies on ZEDboard

v. Project development using setup for a facial recognition system

vi. Running of the program and tests

3. RESULTS AND DISCUSSION

In this section, the results obtained from the study will be presented. The results of the proposed system can be presented as a two-phase process; the first phase is the recognition and detection of the face from the live video and the second phase handles matching the face to the database. However, the steps for the development setup of OpenCV in a Linux environment are discussed first.

3.1. Development Setup

The command required to download and install the dependencies is presented as follows and it summarizes the entire steps taken in the development setup (Figure 2). The prerequisite dependencies required for the proper build of OpenCV were first installed with the package manager for Ubuntu. Thereafter, the compressed files were downloaded and built with Cmake. Finally, the path to the program was configured.

```
sudo apt-get install build-essential libgtk2.0-dev libjpeg-dev libtiff4-
dev libjasper-dev libopenexr-dev cmake python-dev python-numpy python-tk
libtbb-dev libeigen2-dev yasm libfaac-dev libopencore-amrnb-dev
libopencore-amrwb-dev libtheora-dev libvorbis-dev libxvidcore-dev
libx264-dev libqt4-dev libqt4-opengl-dev sphinx-common texlive-latex-
extra libv4l-dev libdc1394-22-dev libavcodec-dev libavformat-dev
libswscale-dev
```

```
cd
wget http://sourceforge.net/projects/opencvlibrary/files/opencv-
unix/2.4.9/opencv-2.4.9.zip
```

```
cd opencv-2.4.9
mkdir build
cd build
cmake-DWITH_TBB=ON -D BUILD_NEW_PYTHON_SUPPORT=ON -D WITH_V4L=ON -D -
INSTALL_C_EXAMPLES=ON INSTALL_PYTHON_EXAMPLES=ON -D BUILD_EXAMPLES=ON -
D WITH_QT=ON -D WITH_OPENGL=ON ../
make
sudo make install
```

```
sudo gedit /etc/ld.so.conf.d/opencv.conf (copy and paste the path to
OpenCV /usr/local/lib)
sudo ldconfig
sudo gedit /etc/bash.bashr (copy and paste
PKG_CONFIG_PATH=$PKG_CONFIG_PATH:/usr/local/lib/pkgconfig and export
PKG_CONFIG_PATH)
```

Figure 2: Development setup for OpenCV

3.2. Face Detection

Figures 3 and 4 show the results of running a simple python script given in the appendix that uses the OpenCV library. The script constructs rectangles on the detected face region using the OpenCV library and thus all detected faces have these rectangles constructed on the face regions in the live video (OpenCV team, 2021). However, the faces of the subjects wearing eye glasses in the video could not be detected by the algorithm. It can be seen from Figure 3 that it fails to detect faces with eye glasses on them. This could possibly be caused by the environmental lighting, facial aging and expressions, similarity in faces and other systematic problems like noise, image-acquisition, video camera distortion etc (Ohlyan et al., 2013).

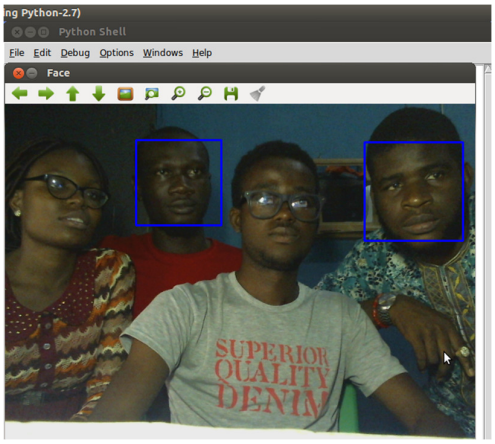


Figure 3: Output of the python script that run OpenCV library to test the face detection technique with some subjects using glasses

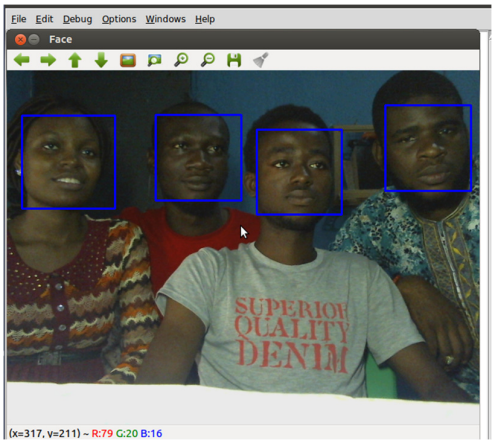


Figure 4: Output of the python script that run OpenCV library to test the face detection technique when subjects have no facial obstructions

3.3. Dataset Creation for Face Recognition

A dataset generator python script was created which was used to capture eight face samples of one person from the live video frame and assign an identification (ID) to them, then saved in a folder called "Dataset". The collection of faces had four subjects having eight samples of different gestures as shown in Figure 5. This is done to build the database and aid the learning of the classifier models.

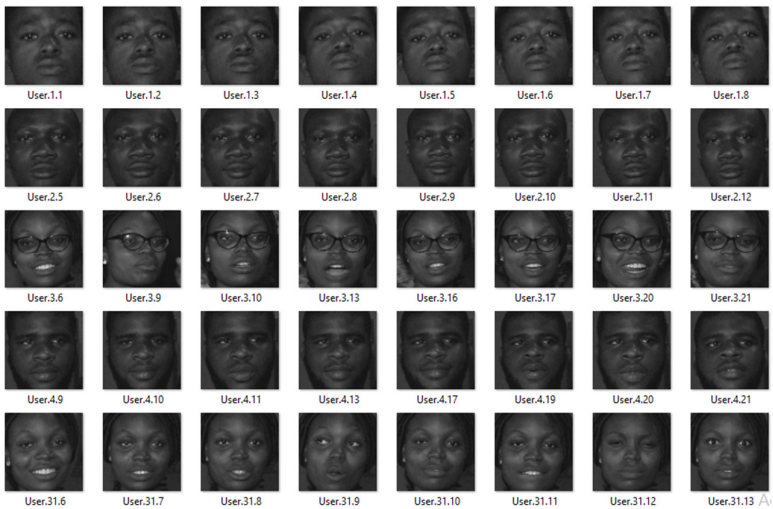


Figure 5: Dataset of faces used for training the face recognition model

3.4. Face Recognition

The final phase would detect faces from a live feed and comparing them to the images stored in the database to get a match. With the set database, the proposed system trains models to recognize images from a live feed (Vaidya et al., 2017). To accomplish that, recognition models available within OpenCV; Eigen Faces, Fisher Faces, and LBPH Faces were employed. The proposed methodology recognized the human face and matched it with the stored database and also displayed the stored name associated with the face of the subject as shown in Figure 6. The recognition system was designed such that the names matching the face associated with its ID in the database appeared on the detected face of the subject. During the test, it was observed that there was huge latency performance in the video streaming and hence, the frame rate was relatively slow. The achievable frame rate correlated with the speed of the ARM cortex-A9 processor (666 MHz). The CPUs load are shown in Figure 7 which hit as high as 100% and 80% for both CPUs. As expected, the challenges associated with face detection were also seen during the face recognition test. Therefore, to improve the accuracy of the results, when capturing the reference subject that will be stored in the database, moderate lighting, straight pose and a normal expression should be used. Furthermore, it was observed that the distances of the subject from the camera mattered.

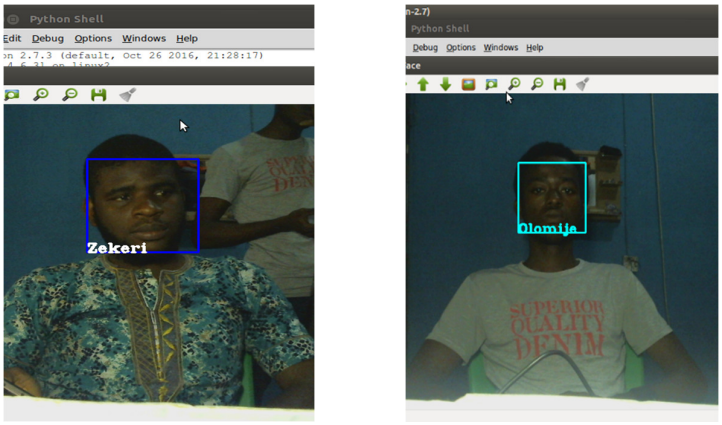


Figure 6: Face recognition results

Figure 7: Latency performance

In this research, there was a relatively high detection rate with OpenCV program similar to the equivalent implementation in MATLAB by Barnouti et al. (2016). The Viola-Jones method offered the highest face detection results without facial obstructions like glasses and gestures. Also, there were appreciable high levels of recognition rates from the test carried out particularly, with two out of the three databases used in the research. In comparison, provided the lighting was moderate and the subject had a straight face with

normal expression, the face recognition rate was also relatively high. Also, the implementation can detect multiple subjects' faces in a video stream as four subjects were tested with the system as opposed to one subject in Barnouti et al. (2016) and Ismael and Irina (2020).

4. CONCLUSION

The proposed system has presented a face recognition system using an embedded platform on a cortex-A9 processor. Primarily due to ease of access without human cooperation, face recognition systems have the possibility of being employed in several implementations such as access control, surveillance, security system, etc. Therefore, the proposed system is a suitable choice because it is a very low-cost implementation of face detection and recognition. The system provides high detection and recognition rates and is less time and memory complex as it is built on a light weight program, OpenCV. However, it should be noted that factors such as the lighting, facial gestures, camera distortion and noise could affect the results.

5. ACKNOWLEDGEMENT

The authors appreciate the contribution and support from the staff of the Department of Electrical/Electronic Engineering, University of Benin, Benin City, towards the success of this research.

6. CONFLICT OF INTEREST

There is no conflict of interest associated with this work.

REFERENCES

Barnouti, N.H., Al-Dabbagh, S.S.M., Matti, W.E. and Naser, M.A.S. (2016). Face detection and recognition using Viola-Jones with PCA-LDA and square euclidean distance. *International Journal of Advanced Computer Science and Applications* (IJACSA), 7, pp. 371–377.

Fernandes, S. and Bala, J. (2013). Performance Analysis of PCA-based and LDA-based Algorithms for Face Recognition. *International Journal of Signal Processing Systems*, 1, pp. 1–6.

Gayathri, M.J.P.R. and Valarmathy, S. (2014). Face Recognition by Using Distance Classifier Based On PCA and LDA. *International Journal of Innovative Research in Science, Engineering and Technology*, 3, pp. 1121–1126.

Hossen, A.M.A., Ogla, R.A.A. and Ali, M.M. (2017). Face Detection by Using OpenCV's Viola-Jones Algorithm based on coding eyes. *Iraqi Journal of Science*, 58, pp. 735–745.

Ismael, K.D. and Irina, S. (2020). Face recognition using Viola-Jones depending on Python. *Indonesian Journal of Electrical Engineering and Computer Science*, 20, pp. 1513–1521.

Jafri, R. and Arabnia, H.R. (2009). A survey of face recognition techniques. *Journal of information processing systems*, 5, pp. 41–68.

Ohlyan, S., Sangwan, S. and Ahuja, T. (2013). A survey on various problems & challenges in face recognition. *International Journal of Engineering Research & Technology* (IJERT), 2, pp. 2533–2538.

OpenCV team (2017) About-OpenCV, [Online], Available: http://www.opencv.org/about.html [23 Aug 2017].

OpenCV team (2021) Cascade Classifier, [Online], Available: https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html [13 Nov 2021].

Ouanan, H., Ouanan, M. and Aksasse, B. (2015). Implementation and optimization of face detection frameworkbased on OpenCV library on mobile platforms using Davinci's technology. *International Journal of Imaging and Robotics*, 15, pp. 81–88.

Vaidya, B., Patel, A., Panchal, A., Mehta, R., Mehta, K. and Vaghasiya, P. (2017). Smart home automation with a unique door monitoring system for old age people using Python, OpenCV, Android and Raspberry pi. International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 82–86.

Viola, P. and Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001, p. 1.

Xillybus (2017) Xillinux: A linux distribution for Z-Turn Lite, ZEDboard, ZyBo and MicroZED, [Online], Available: http://xillybus.com/xillinux [12 Oct 2017].