Multimed Tools Appl (2018) 77:24269–24288
https://doi.org/10.1007/s11042-018-5647-8

Multiple histogram-based face recognition with high speed FPGA implementation

Talal Bonny[1] · **Tamer Rabie**[1] · **A. H. Abdul Hafez**[2]

Received: 30 May 2017 / Revised: 16 November 2017 / Accepted: 11 January 2018 /
Published online: 12 February 2018

Abstract Face recognition is an algorithm that is capable of identifying or verifying a query face from multiple faces in the enrollment database. It poses a challenging problem in the field of image analysis and computer vision, especially for applications that deal with video sequences, face re-identification, or operate on intensity images and require fast processing. In this work, we introduce a high speed face recognition technique along with a high speed FPGA implementation. It uses a new similarity measure to estimate the distance between the query face and each of the database face images. The distance metric is the sum of the standard deviations between multiple histograms, which are calculated from each row of the query and database images. The lowest distance score refers to the database face that matches the query. The proposed technique is independent from the ambient illumination and outperforms the well-known face recognition algorithm "Eigenfaces" (it performs the face recognition 16× faster when both algorithms run on the same platform). Furthermore, we exploit data parallelism in our proposed algorithm to design a hardware accelerator and to implement it on an FPGA prototyping board. The results show 10x execution time improvement in comparison to the software version.

Keywords Face recognition · Gamma correction · Performance · Hardware accelerator · FPGA

✉ Talal Bonny
tbonny@sharjah.ac.ae

Tamer Rabie
trabie@sharjah.ac.ae

A. H. Abdul Hafez
abdul.hafez@hku.edu.tr

[1] Department of Electrical and Computer Engineering, University of Sharjah, Sharjah, United Arab Emirates

[2] Hasan Kalyoncu University, Gaziantep, Turkey

1 Introduction

Recent advances in real-time computer processing have resulted in an increased interest to study face recognition in a dynamic environment. Because of its non-invasive nature and being the primary method of personal identification by humans, face recognition is one of the major biometric technologies [9]. It attracts major interest due to the rapid advances in image capture devices (surveillance cameras, camera in mobile phones), availability of huge amounts of face images on the web, and increased demands for higher security.

Various challenges such as illumination, facial pose, expression, age group, hair color, facial wear, and motion affect the performance of face recognition applications [14, 33]. However, in most of the cases, ambient illumination is the foremost challenge for most recognition applications [30], especially if images/videos are taken from a different camera for re-identification [3, 17]. In this work, we cover this type of category by introducing a new face recognition technique that is robust to ambient illumination.

We solve the problem of performance deteriorating when lighting changes by developing a new face matching technique that is based on standard deviation as a distance function between multiple histograms of a couple of face images. The technique computes the histogram of each row of the query image and then it finds the standard deviation of the histogram of that row. The standard deviations are then compared to find the similarity between images. Our technique may be used for any object recognition among a database of objects such as road traffic signs [16] or comparing information about a scene from several images in stereo vision [23], Our technique may be applied in conjunction with any previous state-of-the-art method to further improve their time performance.

To evaluate our technique, we apply it on the Yale Face Database [8], which includes different facial expressions for different faces, and we measure the performance with different levels of ambient illuminations. We compare the recognition speed of our technique with the most well-known face recognition technique "Eigenfaces" and we show that our technique is 16× faster than the "Eigenfaces" when both techniques run on the same platform. We also implement our technique on an FPGA-based platform by proposing new hardware accelerator to increase the speed of face recognition.

Our software/hardware contributions in this work can be described as follows:

1. A new technique for face recognition is proposed which is based on a new similarity criteria of measurement, i.e. the standard deviation of rows' histograms. The new technique provides accurate results regardless of the level of ambient illumination.
2. The time required to recognize a query face among a different database of faces is explicitly reduced using our technique in comparison to the well-known face recognition algorithm "Eigenfaces".
3. The data parallelism of the proposed technique is exploited and the technique is implemented on FPGA prototyping board by proposing new hardware accelerator architecture which has a great potential to dramatically increase the processing speed.

The rest of the paper is organized as follows: In Section 2, we review the most recent related work to highlight the novelty of our proposer. Section 3 presents the theory behind our new multiple histograms based face recognition algorithm, in addition to an overview of the proposed algorithm. Section 4 presents experimental evaluation of the robustness of the proposed face recognition technique to illumination. Gamma correction method is used to add levels of ambient illumination. In Section 5, we show how to exploit the data parallelism of the proposed technique and present the implementation of the FPGA hardware

accelerator. Experimental results and performance evaluation are presented in Section 6. We conclude this work in Section 7.

2 Related work

In this section, we review the some of the state-of-the-art face recognition works. We start with applications that were implemented on accelerated FPGA platform. After that we present the related face recognition works with respect generic implementation.

Many attempts have been made to accelerate the process of recognition in different areas using FPGAs [22, 24]. The authors in [24] accelerated detecting the facial emotion of children who have autism spectrum disorder. Those children have difficulty in understanding the emotional and mental states from the facial. The authors in this work used the principal component analysis (PCA) algorithm to extract the features and implemented it on the Virtex 7 FPGA and achieved 82.3 % detection accuracy. The authors in [13] proposed design which uses the feature-based method to recognize a face implemented using a combination of FPGA and personal computer (PC). The FPGA captures a video frame and extracts the locations of facial feature points. The computer calculates dissimilarity scores for each face template stored in the database and finds the best match. Another attempt was in [22] where the authors proposed a fast-prototyping of a fall detection system in a home environment that can be useful for helping elderly people in daily life. Their implementation uses a single camera and an optimized descriptor adapted to real-time tasks using system-on-chip Zynq FPGA.

In [27] the authors proposed a simple Spherical Hashing based Binary Codes (SHBC) feature learning method to learn binary face descriptors. They clustered and pooled learned binary codes into a histogram-based feature that describes the co-occurrence of binary codes, and considers the histogram-based feature as the final feature representation for each face image. They investigated the performance of their SHBC method on various face databases and concluded that their SHBC descriptor outperforms other state-of-the-art face descriptors. In [12] the authors proposed a local center of mass face (LCMF) method to extract illumination insensitive features for face recognition. LCMF extracts the gradient angle between the center of mass and center pixel of a selected neighborhood. The angle of the slope between the center of mass and the center pixel of the neighborhood is used as a feature vector. They used the L1 norm distance measure of these feature vectors to classify the images. Their method does not involve training of images.

The last decades have seen an increased interest by researchers to address the problem of accuracy and complexity of face recognition algorithms. Starting from the pioneering work of "Eigenfaces" by Pentland et al. [29] to the latest work of DeepFace by Wolf et al. presented in [26] and DeepID by Wang et al. presented in [25], researchers are able to improve the recognition rate to near perfection. "Eigenfaces" is a well-known technique used widely for face detection and recognition [21, 28, 29]. It's popularity comes from its simplicity and its real time support comparing to other techniques. The discriminating features are simply the eigenvectors of the set of faces from which the name "Eigenfaces" comes.

Histogram of intensity, as a statistical image description [5], has been attracted by the research community in the past and recent years [2, 4, 6, 7]. Face recognition was proposed in [6, 7] by matching the gray-scale intensity histogram of the test image with those of the training images. However, direct matching as in [6] does not account for changes in lightning conditions.

Face recognition evaluation reports [18, 19] as well as the conclusions from the reviewed related work above, indicate that the performance of many state-of-the-art face recognition methods deteriorates with changes in lighting, pose, and other factors [20]. In addition, linear methods like those presented in [1, 15, 29] are unable to preserve face manifolds necessary to differentiate among individuals. We propose in this article to match multiple sub-histograms of the rows of the face images and find the difference between their standard deviation which is found to be invariant to changes in lightning conditions. In addition, it facilitates the parallel implementation on an accelerated hardware using FPGA platform.

3 Face recognition based on multiple histograms

The proposed multiple histogram face recognition method is presented in this section. The multiple histograms here are derived each from one row of the image, but the concept is presented in a general formula to allow using multiple histograms from different portions of the image. In the sequel, we present the concept of the image histogram and then histograms of multiple portions of the image, particularly these portions are the rows of it. Then, we present the proposed distance measure between multiple histograms.

3.1 Theoretical definition of image histogram

Consider that we have an image $I(x, y)$ that contains N rows and M columns, the total number of pixels is given as $N \times M$. Assuming that each pixel value $I(x, y)$ is represented using 8-bits and the image is a monochrome image, then $I(x, y) = r \in [0, 255]$. Then for a pixel with index $i \in [1, NM]$ and coordinates (x, y), the color intensity value is denoted as $I(i) = I(x, y) = r$.

An image histogram is a vector $H(r)$ that represents the number of occurrences n_r of all possible values of the discrete variable r. Given the histogram $H_I(r)$ of the image $I(x, y)$, it can be represented using the vector

$$H_I(r) = [\, H_I(0) \ \cdots \ H_I(r) \ \cdots \ H_I(255) \,]. \tag{1}$$

Each element of this vector is given as

$$H_I(r) = \sum_{i=1}^{NM} n_{ri}, \tag{2}$$

where $n_{ri} = 1$ if $I(i) = r$, and $n_{ri} = 0$ otherwise.

One of our main contributions in this work is the recognition using histograms of individual rows $s \in [1, N]$ from the face image. The histogram for a certain row s can be written as

$$H_I(r, s) = [\, H_I(0, s) \ \cdots \ H_I(r, s) \ \cdots \ H_I(255, s) \,]. \tag{3}$$

Each element of this vector is given as

$$H_I(r, s) = \sum_{i=1}^{M} n_{ri,s}, \tag{4}$$

where $n_{ri,s} = 1$ if $I(i) = r$, and $n_{ri,s} = 0$ otherwise. Note that here $i \in [1, M]$ where M is the number of pixels in each row.

3.2 Face recognition using distance between multiple histograms

We propose a new distance metric to compare two face images, which is based on the standard deviation. It is highly discriminative and easy to be implemented on an accelerated hardware platform. The proposed method is empirically proven in Section 4 to be robust to changes in lighting conditions. The standard deviation $STD_I(s)$ of the histogram $H_I(s)$ of the row s in the image I gives an idea of how close the entire pixels' intensities occurrences in the row s to the average occurrences value. A histogram with with large standard deviation has occurrences which are spread out over a wide range of values. The standard deviation $STD_I(s)$ of the histogram $H_I(s)$ of the row s in the image I is defined mathematically as follows:

$$STD_I(s) = \sqrt{\frac{\sum_{r=0}^{255}(H_I(s) - \bar{H}_I(s))^2}{255}} \tag{5}$$

Where $\bar{H}_I(s)$ is the average value of the row histogram $H_I(r, s)$. Finally, a vector STD_I is calculated, it contains N components each of which represents $STD_I(s)$ of the sth row of the image I.

Given a query face image Q, the task is to compute the distance metric of this image to every image D from the database. Our proposed distance metric technique is L_1 norm of the difference vector between STD_Q and STD_D. This distance is called the city block distance metric as well. These are the standard deviation vector of the query image and an image from the database respectively. The vector STD_D corresponding to each image in the database is calculated in an off-line step while STD_Q is calculated online.

Mathematically, the distance score (DS) between the query face 'Q' and the database face 'D' is defined as following:

$$DS = \sum_{s=1}^{N} |STD_Q(s) - STD_D(s)|, \tag{6}$$

where N is the number of rows in each of both images, $STD_Q(s)$ is the standard deviation of the histogram of row s for the query face image, and $STD_D(s)$ is the standard deviation of the histogram of row s for the database face image.

The next section shows an overview of the proposed algorithm. In addition, we are going to show in the next section that this proposed distance metric is robust to changing the lighting conditions with no considerable effect on the estimate of the standard deviation of the histogram of the image. The well known Gamma correction method has been used to add illumination effects to the images.

3.3 Our face recognition technique

As discussed in Section 1, our face recognition technique involves one-to-many matching that compares a query face against multiple faces in the enrollment database. In each comparison, a distance score (described later in this section) is calculated. The lowest score

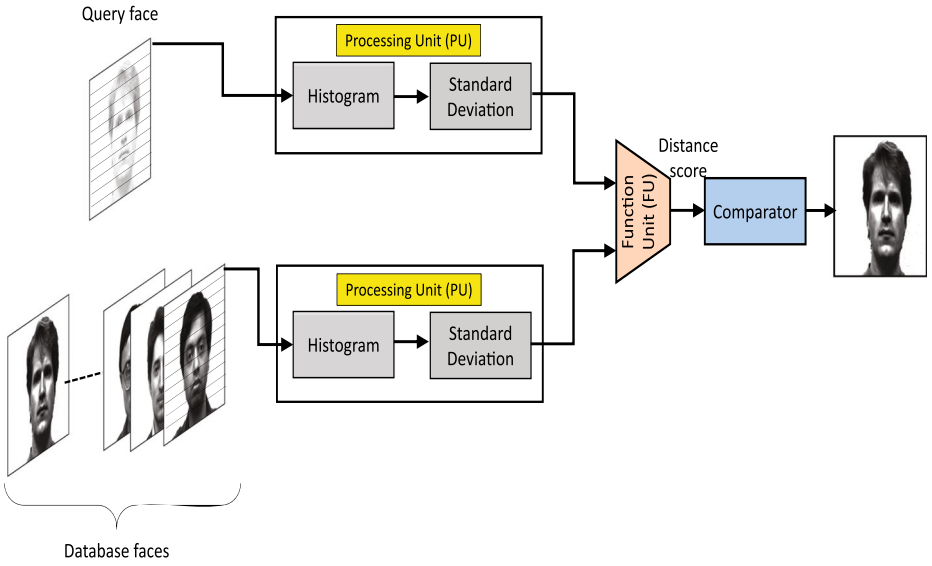

Fig. 1 Block diagram of our face recognition algorithm to compare query face against multiple faces in a database and to compute the distance score for each one. The database face which has the lowest distance score matches the query face

refers to the face in the database that is most similar to the query one. Our technique assumes the following:

1. The user is willing to be cooperative but he is unable to present the face in a favorable condition with respect to the camera. In other words, our technique assumes that facial pose, expression, age span, hair, facial wear, and motion are not considered in query face.
2. As ambient illumination is the foremost challenge for most face recognition applications, our technique considers this factor when it computes the distance score. So, changing the lighting level will not affect the performance of our technique. The method that is used to change the ambient illumination of images is Gamma Correction.
3. All faces are in gray-scale, i.e., each pixel is represented as 8-bit value (0 to 255).

As depicted in Fig. 1, our proposed face recognition algorithm passes through the following steps:

The first step, to compare a new query face against multiple faces in the enrollment database, is to calculate the histogram for each row of the query face individually according to (4) and (2). To do that, each row is considered as an array of pixels equal to a number of image columns. Each pixel might be any number between 0 and 255 (pixels values). The row array is scanned starting from the first pixel till the last one using 256 counters. One counter for each pixel value. Each counter is incremented by one when it meets new pixel of the same value. At the end of scanning, each counter has a number of repeated pixel values (bin count) stored in the array "hist". The hardware implementation of the histogram calculation process is presented in Section 5. Moreover, the Algorithm 1 presents pseudo-code for calculating the histogram of an image row. After calculating the histograms of all rows, the standard deviation for the histogram of each row is calculated according to (5).

For any new database of faces, objects, etc., calculating the histogram and the standard deviation have to be done off-line (in design time). Therefore, it does not matter how long time it takes.

Algorithm 1 Calculate histogram for image row

```
procedure ROW_HISTOGRAM(row, hist, n)
    // n is number of image columns
    // row is array of n pixels
    // hist is array of 256 elements initialized with zeros
    for each pixel i in row do
        hist[row[i]] = hist[row[i]] + 1;
    end for
    Return hist
end procedure
```

In the next step, The distance score between the query and database faces is calculated as shown in (6). It reflects how close is the query face from database face. A lower score means better matching. This step is repeated for each face image registered in the database and to compute its distance score with the query face. In the last step of our technique, the comparator compares the distance scores and finds the minimum one. The minimum score refers to the database face which is much closer to the query face. The full hardware implementation is presented in Section 5.

4 Robustness to illumination and gamma method

4.1 Overview on gamma intensity transform

Gamma is one of the important factors that can control the intensity level of an image. Thus, gamma correction is a non-linear operation that is used widely to adjust the luminance channel and produce an image with different brightness levels [10]. Gamma correlation technique can be considered as a histogram modification technique that can be obtained by:

$$f(x) = x^{1/\mu} \tag{7}$$

where μ is the gamma value, x represents the input gray-level value and $f(x)$ is the output gamma-corrected gray-level value of the image. This method boosts dark pixels much more than bright ones. Figure 2 shows the output gamma-corrected gray-level values when the input values of the images change from 0 to 1, for different μ values: 0.5, 1, 2, and 3.

From this figure, we can see that when $\mu = 1$, then there is no changes in the output values. To obtain a darker image, μ should be below 1. In this case, the dark pixels values are decreased (becoming darker) more than the bright ones (see Fig. 2, $\mu = 0.5$). To obtain a brighter image, μ should be above 1. In this case, the dark pixels values are increased (becoming brighter) more than the bright ones (see Fig. 2, $\mu = 2$, and $\mu = 3$).

4.2 Robustness evaluation of the proposed method against ambient illumination

To evaluate our face recognition technique, we change the ambient illumination of the query image by using Gamma correction method with different $\mu = \{1, 0.5, 2, 3\}$ values. These values respectively correspond to "No ambient illumination is added", "Dark ambient

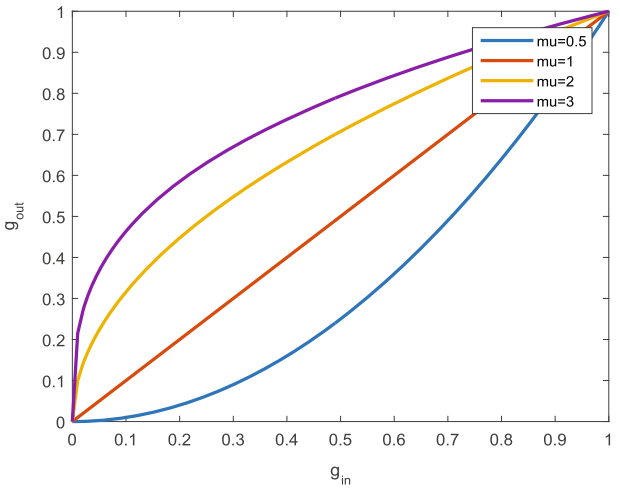

Fig. 2 Gamma correction method with different values of Gamma

illumination is added", "Bright ambient illumination is added", and "Brighter ambient illumination is added".

If the query face has a lighting level which is not the same as in the database face, then the histogram differences will not refer to correct comparison. This is because the pixel values of the image are increased/decreased based on μ value. For example, Fig. 3 shows the histograms of the first row of a query face for different μ values: In Fig. 3a, the query face has no added ambient illumination ($\mu = 1$). One of the highest repeated pixels (bins) in this figure is "128" (bin count = 43). When dark ambient illumination is added by $\mu = 0.5$ (Fig. 3b), the histogram is shifted to the left (toward the low pixel values) and the highly repeated pixel "128" (bin count = 43) becomes "64" (with the same bin count). Adding bright ambient illumination by $\mu = 2$ (Fig. 3c) shifts the histogram to the right (toward the high pixel values). Now, the high repeated pixel becomes "181" and its frequency is increased (bin count = 44). Increasing the μ value to 3 (Fig. 3d) shifts the histogram further toward the high pixel values. Therefore, the high repeated pixel becomes "202" and its frequency increases (bin count = 49).

Figure 3 also shows the standard deviation of the histogram of a query face for different μ values: In Fig. 3a, the standard deviation of the histogram of the first row is 25.92. In Fig. 3b, STD is almost the same as in Fig. 3a (actually, the two values differ in the third digit in the fractional part) because the change in μ was very small (from 1 to 0.5). STD increases slightly to 25.951 when $\mu = 2$ (Fig. 3c) and to 26.285 when $\mu = 3$ (Fig. 3d).

From the previous discussion, we conclude that changing the lighting of an image using Gamma correction method will change the histogram but will not affect (or affect slightly) the standard deviation of the histogram of the image. Hence, the standard deviation is a good option to compare the query and database faces if the query face has different lighting level.

5 FPGA hardware accelerator implementation

In this section, we show how to improve the time performance of our face recognition algorithm by implementing it using FPGA prototyping board. Before we implement it, we

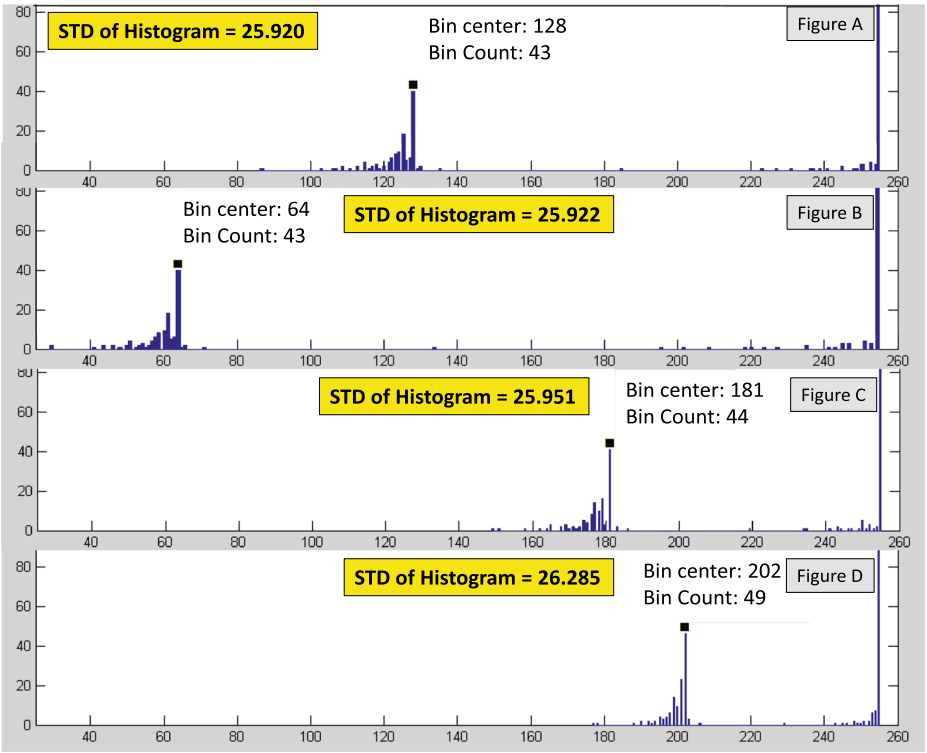

Fig. 3 The histograms and standard deviations of the first row of a query face for different μ values: A: $\mu = 1$, B: $\mu = 0.5$, C: $\mu = 2$, D: $\mu = 3$

need first to exploit the data parallelism in our algorithm and based on it we design the hardware accelerator.

High level of data parallelism in our algorithm can be exploited in two ways: firstly, all rows of query face are independent. This allows each row to be processed in parallel with other rows. Secondly, calculating histogram, standard deviation, and distance score are independent. So, executing them can be adopted in different pipeline stages.

The prototyping FPGA Zed-Board from Xilinx[1] is one of the good choices to implement our algorithm. The board contains ZYNQ-7000 All Programmable SoC FPGA [11] which has a capacity of 13,300 logic slices, 220 DSP48E1s, and 140 BlockRAMs. The Zynq chip combines ARM dual-core Cortex – A9 MPCore Processing System (PS) with Xilinx 28nm Programmable Logic (PL), where the user can create his own IP cores, on the same chip. The two parts communicate with each other through the industry standard Advanced eXtensible Interface (AXI) interconnects which provide high bandwidth, low latency connections between the two parts of the device [31]. The board contains also 512 MB DDR3 Memory and some other peripherals to enable users to experiment with a various aspect of their embedded designs.

Figure 4 shows the block diagram of our hardware implementation on the FPGA board. It consists of the following IP blocks:

[1] www.digilentinc.com/

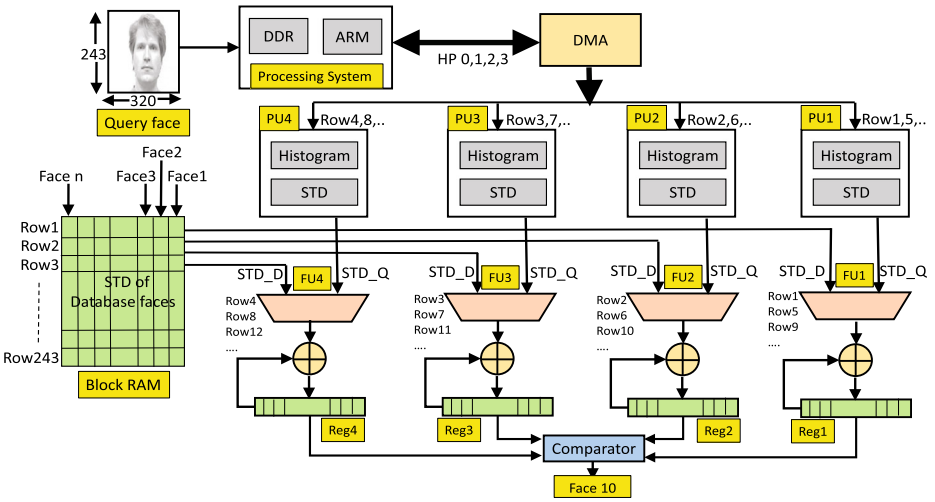

Fig. 4 Hardware accelerator of our face recognition algorithm on FPGA Zed-Board

- The Zynq Processing System (PS) which includes the ARM processor and DDR memory controller.
- The FPGA Block RAM.
- The AXI Direct Memory Access (DMA).
- The Processing Units (PU).
- The Function Units (FU).
- The Comparator IP.

The rows of query face (320 × 243 pixels) are stored in the DDR memory, while the Block RAMs are used to store the STD of each row for each database face. These values were calculated off-line (design time) for only one time. For any new database face, its STD for each row has to be calculated and stored in the Block RAM. In Fig. 4, if our database has n database faces, each row of the Block RAM will have n STD values. Each value refers to the STD of the row of the corresponding database face.

The DMA is used to transfer data from the DDR memory to other system parts through High Performance ports (HP) without interfering with the CPU. Using the DMA will increase the data throughput and will offload the processor from tasks that involve memory transfers. The processor is required just in the beginning to initiate transfers by programming the DMA controllers registers. The processing unit (PU) consists of two IP blocks, the histogram and the standard deviation (STD). It receives the query face row from the DDR through the DMA port and calculates the histogram for the row and then it computes the standard deviation for it. The function unit (FU) gets the STD of query face from the PU and the STD of database face from the Block RAM and then it computes the distance score based on (6). It also sums up the distance scores of all rows to find the final one between query face and database face, and then stores the results in a register. For example, Reg1 will have the sum of STDs for each database face and for only the rows 1, 5, 9, etc. Reg2 will have the results for rows 2, 6, 10, etc. and so on. The comparator stores all distance scores and finds the minimum one which refers to the database face who matches the query one. Algorithm 2 shows pseudo-code of hardware accelerator for matching the query with database faces on FPGA Zed-Board.

Algorithm 2 Pseudo-code of hardware accelerator on FPGA Zed-Board

```
procedure DATABASE_QUERY_MATCHING(Q, STD_D)
    // Q is query face
    // STD_D is standard deviation of database face
    // PU is processing unit
    // FU is function unit
    for each row S in Query Q do
        PU finds histogram hist[S]
        PU finds standard deviation STD_Q[S]
        for each database face i of all database faces D do
            FU gets STD of row S of database face i STD_D[s][i] from Block RAM
            FU finds distance score DS[i] = DS[i] + |STD_Q[s] − STD_D[s][i]|
        end for
    end for
    for each database face i of all database faces do
        Comparator finds minimum distance Min through all distances DS[i]
    end for
    Return database face i which has the minimum distance Min
end procedure
```

In the FPGA Zed-Board, the High Performance ports (HP) are used to access the external memory (DDR3). Those ports are designed for maximum bandwidth of 4 ports × 64 bits × 150 MHz = 4.8 GByte/sec. To exploit the full bandwidth, we use the four 32-bit DMA ports (HP0, HP1, HP2, and HP3) of the processing System to receive data from the DDR, and consider each one as an individual thread. Therefore, our hardware accelerator contains four parallel threads. Each thread consists of the processing unit (PU) to calculate the histogram and standard deviation, and function unit (FU) to calculate the distance score. Hence, our accelerator can process four rows in parallel.

The execution of each thread passes through different pipeline stages (see Fig. 5): In the beginning, the histogram of row 1 is calculated through the first pipeline stage. The other two pipeline stages are empty. Once the histogram is calculated, row 1 is shifted to the second stage and the standard deviation is calculated. In the same time another row enters the histogram pipeline stage, and so on. The three pipeline stages will be busy in calculations as soon as row 3 is delivered by the DMA and enters the pipeline stage. This will ensure high level of data parallelism and will achieve high execution performance.

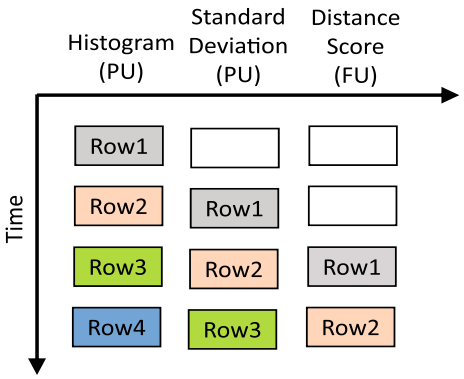

Fig. 5 The pipeline stages for executing our algorithm in the hardware accelerator

Our Design is implemented on the prototyping FPGA Zed-Board, and the execution time and FPGA resources are reported in the experimental results section.

6 Experimental results

In this section, we present the experimental results of our face recognition technique by applying it on a tested database from the Yale Face Database [8]. The tested database contains total of 480 faces (320 × 243 pixels) including eight different facial expressions and four different ambient illumination levels: original level ($\mu = 1$), dark level ($\mu = 0.5$), bright level ($\mu = 2$), and brighter level ($\mu = 3$). Figure 6 shows the eight different facial expressions of an original sample face without ambient illumination ($\mu = 1$) (top part). Figure 6 also shows the same sample face but with three ambient illumination levels, dark, bright, and brighter levels (bottom part). The proposed technique is evaluated for the recognition performance and efficiency.

Our technique is independent of the image size. It computes the histogram of each row through all columns, then it finds the standard deviation of the histogram for each image row. At the end, our technique uses one value (standard deviation) for each image row to compute the distance score independently from a number of columns or rows.

6.1 Recognition performance evaluation

The experimental performance evaluation of our technique is depicted in Fig. 7, 8, 9, 10, 11, 12, 13 and 14.

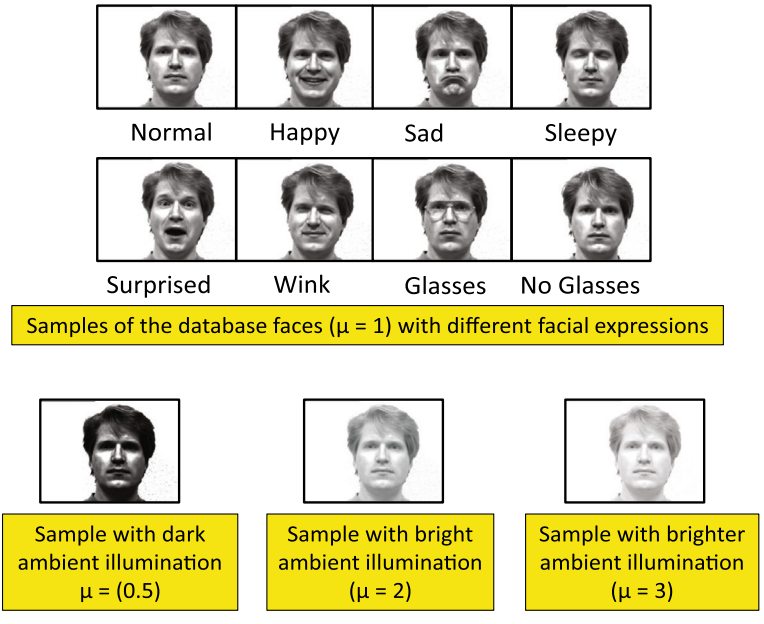

Fig. 6 Samples of the tested database with different facial expressions (top part), and with different ambient illumination levels (bottom part)

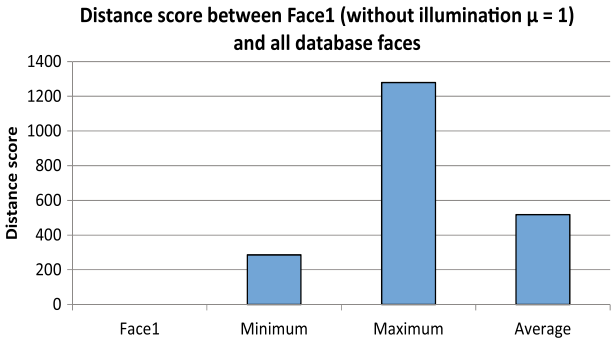

Fig. 7 Distance between the first query face (Face 1) and the database Face 1 (no ambient illumination, $\mu = 1$). In addition to the minimum, maximum, and average scores with all other faces. The query face matches Face 1 in the database (distance = 0) and does not match other faces (minimum, maximum, and average distances > 0)

The performance is evaluated for different ambient illumination levels: dark level ($\mu = 0.5$), bright level ($\mu = 2$), and brighter level ($\mu = 3$), in addition to the case where there is no ambient illumination ($\mu = 1$).

As it is difficult to present all figures, and for brevity, we show samples of the results for selected faces from the tested database. The faces are Face 1, Face 2, Face, 5, and Face 10. In each figure, the first bar presents the distance score between the distorted query face and its original one in the database. The second bar presents the minimum distance score between the query face and all other database faces. The third and fourth bars presents the maximum and the average distance scores, respectively.

Figure 7 shows the distance score between query and database faces. In this figure, Face 1 is used as a sample case query. This face has no ambient illumination ($\mu = 1$) as well as all database faces. Our technique recognizes Face 1 in the database by showing that the distance score is zero (first bar in Fig. 7), but all other distance scores (minimum, maximum, and average) are greater than zero (the remaining three bars in Fig. 7). This implies that Face 1 is identified in the database correctly.

Adding bright ambient illumination level ($\mu = 2$) to Face 1 as a query face does not affect our face recognition algorithm. Figure 8 shows another sample for evaluation. The

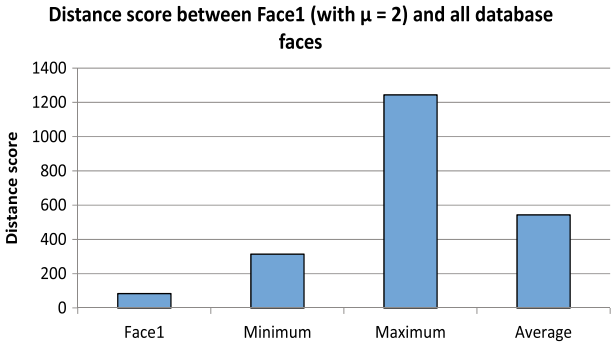

Fig. 8 Distance between the query Face 1 and the database Face 1 (with bright ambient illumination, $\mu = 2$). In addition to the minimum, maximum, and average scores between query Face 1 and all other database faces. The distance score of Face 1 in the database is lower than other faces

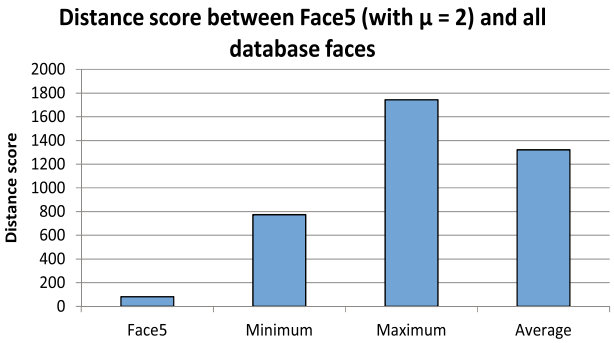

Fig. 9 Distance between the query Face 5 and the database Face 5 (with bright ambient illumination, $\mu = 2$). In addition to the minimum, maximum, and average scores with all other database faces. The distance score between query Face 5 and Face 5 in the database is lower than other faces

query in this figure is still Face 1 but with added illumination. The distance score between Face 1 with (bright illumination) and the original Face 1 in the database faces remains less than the distance scores (minimum, maximum, and average) with other database faces. This identifies the face in the database even if the query face has an ambient illumination level.

Another result sample is shown in Fig. 9 where the query is Face 5 with a bright ambient illumination level ($\mu = 2$). The distance score between Face 5 (query) and Face 5 (database) is the lowest among other database faces. This shows that the query face matches Face 5 of the database faces. Figure 10 shows another sample when the query face is Face 10 (and also with illumination level). The figure shows that the lowest distance score appears for Face 10 of the database faces. From previous results, we can conclude that for any query face, if its distance score with a database face is the smallest, the query matches this database face.

Our proposed technique will not be affected when the ambient illumination level changes. In Fig. 11, we use Face 2 as a query with three ambient illumination levels: $\mu = 0.5$, $\mu = 2$, $\mu = 3$. For any illumination level, the distance score between Face 2 (as a query) and Face 2 in the database is the lowest among other database faces which means that Face 2 is identified simply using our proposed method for any illumination level.

Figure 12 shows how the ambient illumination level affects the distance score between the query face and database faces by showing the minimum distance score between our four

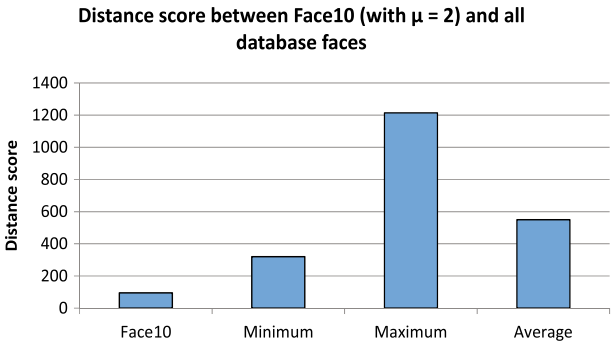

Fig. 10 Distance between the query Face 10 and the database Face 10 (with bright ambient illumination level, $\mu = 2$). The distance score between query face and Face 10 in the database is lower than other faces

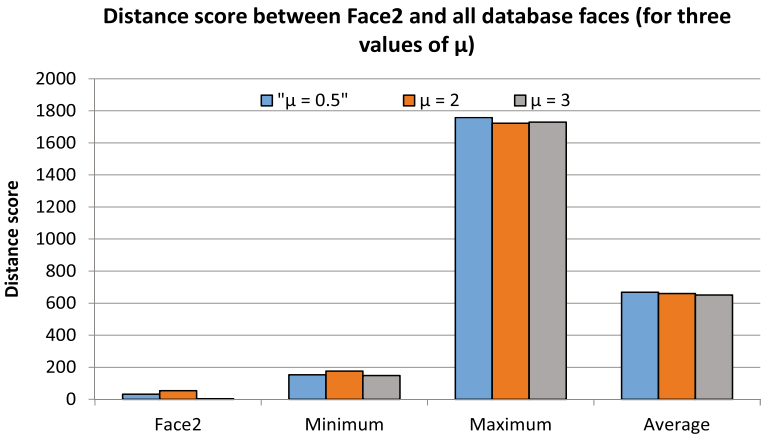

Fig. 11 Distance between the query Face 2 and database Face 2 (with different ambient illumination levels, $\mu = 0.5, 2, 3$). In addition to minimum, maximum, and average distances with all other faces. The distance score between query face and Face 2 in the database is lower than other faces for all illumination levels

query samples and all database faces. It is clear that when the ambient illumination level applied on the query face is high ($\mu = 3$) or low ($\mu = 0.5$), the minimum distance score is low but when the illumination is average ($\mu = 2$), the score is high. The reason for that is, high bright illumination level converges the pixel values in the bright range of pixel scale, and when high dark illumination level coverages the pixel values in the dark range of pixel scale (as shown in Fig. 2). This makes the standard deviation of the illuminated query face more close to the original one in the database. and hence reduces the distance score. From the previous discussion, we conclude that for Gamma correction method, the higher ambient illumination level ($\mu = 3$, 4 or 5, etc.) the less distance score. And the same conclusion when the illumination gets lower.

To show the effect of having an ambient illumination from only one direction on our technique, two more sidelight effects are added to each face of the Yale Face Database. A sample of the sidelight effects from right and left are shown in Fig. 13.

Figure 14 shows samples of results for query faces with different facial expressions shown in Fig. 6 and different sidelight effects shown in Fig. 13. The first bar shows the

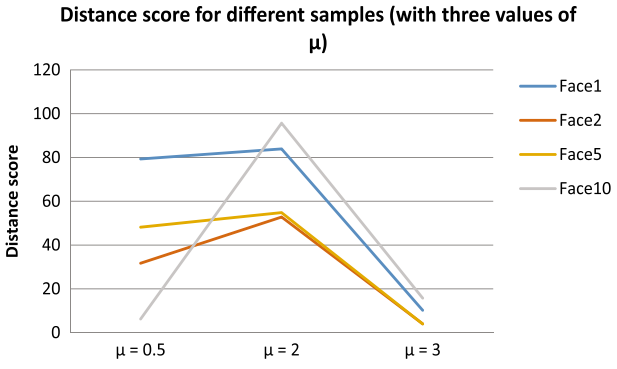

Fig. 12 The effect of different ambient illumination levels on distance score for our four query samples. The score is low when the illumination level is high ($\mu = 3$) or low ($\mu = 0.5$)

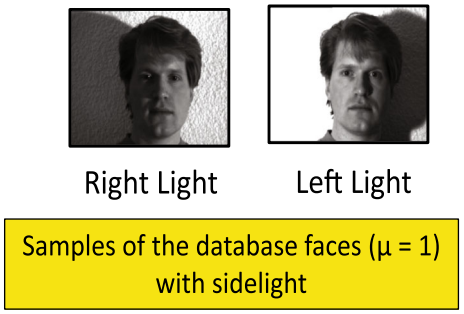

Fig. 13 Samples of database faces with sidelight effects (right and left) without any added ambient illumination ($\mu = 1$)

distance score between the query face (with facial expressions) and the original face in the database (without facial expressions). The second bar shows the distance score between the same query as in the first bar but with the next closest face in the database. From this figure, it is worth noting that the distance score in the first bar is less than the second bar in the queries with facial expressions but not in the queries with sidelight effects (last two bars). This concludes that our approach recognizes all faces with different facial expressions but not the faces with sidelight effects. The reason for that is, when side lighting is added, the histogram of rows will change asymmetrically and our approach will not be able to recognize it. In case of the "Eigenfaces" approach, the reconstruction error is reduced by increasing the basis features of face images which are collected from the training set.

6.2 Efficiency evaluation

This section shows performance improvement by comparing the execution time of our algorithm with the well-known face recognition algorithm "Eigenfaces". As the "Eigenfaces"

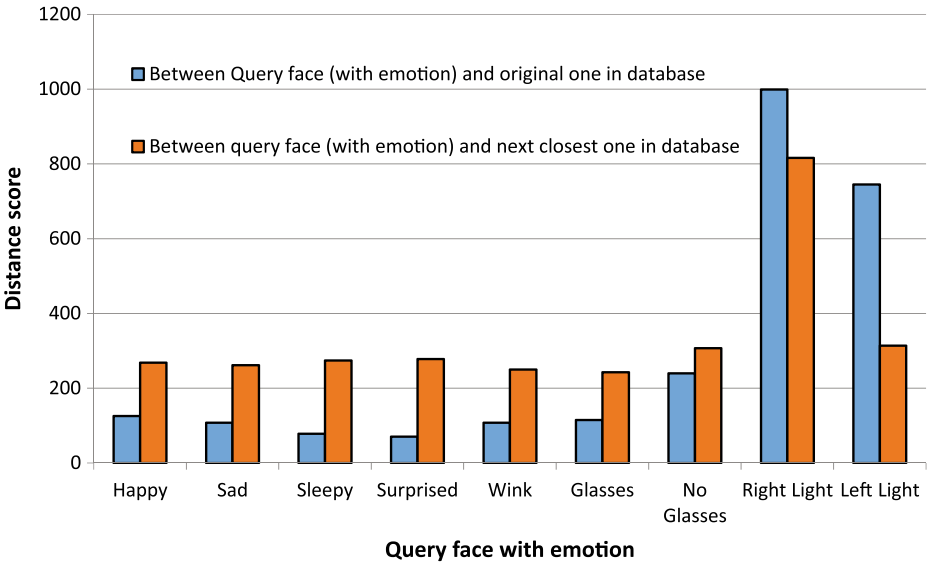

Fig. 14 Distance score between the query face (with facial expressions) and the original face in database (first bar), and the next closest face in the database (second bar). Our approach recognizes all faces with different facial expressions but not the faces with sidelight effects

Table 1 Algorithm Execution Time (in seconds)

Software version (on desktop)		Hardware version (on FPGA)
Eigenfaces	Our algorithm	Our Algorithm
160	10	1

algorithm uses "OpenCV" library, our algorithm is re-written in C++ using this library to assure fair comparison. The platform used to execute both programs is Microsoft Visual Studio 2015. The same database faces and same query face with ambient illumination level are used in both algorithms. Table 1 shows execution time of the "Eigenfaces" algorithm (column 1) and our algorithm (column 2) when both algorithms run on a same platform (Intel core-I5 @ 350 GHz, 64-bit operating system). From this table, it is clear that our algorithm is 16× faster than the "Eigenfaces" algorithm. When a number of faces in the database is larger the time difference between both algorithms will be much bigger because the standard deviation for any new face in the database face in our algorithm would be calculated off-line.

To create our design, synthesize it, implement it, and verify it on the FPGA prototyping board (Zed-Board), the Xilinx Vivado design suite [32] is used. The code is re-written in VHDL and simulated using ModelSim. All IPs are clocked using 100 MHz frequency clock signal generated by the processing system. The STDs of all face rows are stored in FPGA Block RAMs. Each STD has 4 bytes (floating point). Our design uses FPGA prototyping board which includes fabricated configurable logic blocks. The FPGA logic gates capacity and the size of its BLOCK RAM determine the price of the FPGA chip and the prototyping board. The cost of our Zed-board does not exceed $200.

The Zynq Z-7020 FPGA chip of the Zed-Board is suitable for the implementation, as it contains 4.9 Mb size of Block RAMs. This size would be enough to store all STDs of all rows (243 rows) of all database faces.

Our implemented embedded design on the FPGA Zed-Board utilizes very small amount of the FPGA resources. Table 2 summarizes the resources utilization of our hardware accelerator. For example, only 18% of the slice look-up tables and 11% of the Flip-Flops registers are utilized.

To measure the execution time of our algorithm, we add AXI_Timer IP to our design. This IP is clocked using 100 MHz frequency clock signal generated by the processing system and used to measure the time for any code segment. Column 3 in Table 1 shows execution time of our algorithm on the FPGA Zed-board which is 10× faster than the software version and 160× faster than the "Eigenfaces" algorithm.

The high speed execution time of our design would be enough to support the real-time applications. Therefore, our future plans are to test it for real-time applications such as video surveillance and to measure its efficiency.

Table 2 Resources utilization of our hardware accelerator

Resources	Utilization	Available	Utilization (%)
FF	11850	106400	11.14
LUT	9594	53200	18.03
Memory LUT	1222	17400	7.02
BUFG	1	32	6.25

7 Conclusions

We have presented a new technique to accelerate identifying a query face among database faces. The database used for evaluation contains facial images from the Yale Face Database with different facial expressions and ambient noise levels. Our technique has the advantage of fast face identification even if the ambient illumination changes. We compared our technique with the well-known face recognition algorithm "Eigenfaces" and showed that the software version of our algorithm recognizes the query face among database faces 16× faster than the "Eigenfaces" algorithm. We also implemented the proposed design on an FPGA Zed-Board and showed that the implemented hardware design improves the software version by 10x with low FPGA resources utilization (18% of the slice look-up tables and 11% of the Flip-Flops registers).

References

1. Bartlett MS, Movellan JR, Sejnowski TJ (2002) Face recognition by independent component analysis. IEEE Trans Neural Netw 1:1450–1464
2. Bateux Q, Marchand E (2017) Histograms-based visual servoing. IEEE Robot Autom Lett 2(1):80–87
3. Bedagkar-Gala A, Shah SK (2014) A survey of approaches and trends in person re-identification. Image Vision Comput 32(4):270–286. https://doi.org/10.1016/j.imavis.2014.02.001
4. Beheshti I, Maikusa N, Matsuda H, Demirel H, Anbarjafari G (2017) Histogram-based feature extraction from individual gray matter similarity-matrix for alzheimers disease classification. J Alzheimers Dis 55(4):1571–1582
5. Cha SH, Srihari SN (2002) On measuring the distance between histograms. Pattern Recogn 35(6):1355–1370
6. Demirel H, Anbarjafari G (2008) Pose invariant face recognition using probability distribution functions in different color channels. IEEE Signal Process Lett 15:537–540
7. Déniz O, Bueno G, Salido J, De la Torre F (2011) Face recognition using histograms of oriented gradients. Pattern Recogn Lett 32(12):1598–1603
8. Georghiades A, Belhumeur P, Kriegman D (1997) Yale face database. In: Center for computational vision and control at Yale University, vol 2. http://cvc.yale.edu/projects/yalefaces
9. Hietmeyer R (2000) Biometric identification promises fast and secure processing of airline passengers. Int Civil Aviat Org J 55(9):10–11
10. Huang SC, Cheng FC, Chiu YS (2013) Efficient contrast enhancement using adaptive gamma correction with weighting distribution. Trans Image Process 22(3):1032–1041. https://doi.org/10.1109/TIP.2012.2226047
11. Inc X. (2014) 7 Series FPGAs Overview, vol 1. Xilinx
12. Kar A, Sarkar S, Bhattacharjee D. (2017) Local centre of mass face for face recognition under varying illumination. Multimed Tools Appl 76(18):19211–19240
13. Leung HY, Cheng LM, Li XY (2015) A fpga implementation of facial feature extraction. J Real-Time Image Process 10:135–149. https://doi.org/10.1007/s11554-012-0263-8
14. Li C, Yee LY, Maruyama H, Yamaguchi Y (2017) Fpga-based volleyball player tracker. SIGARCH Comput Archit News 44:80–86. https://doi.org/10.1145/3039902.3039917
15. Liu C, Wechsler H (2002) Gabor feature based classification using the enhanced fisher linear discriminant model for face recognition. Trans Image Process 11(4):467–476. https://doi.org/10.1109/TIP.2002.999679
16. Malik A, Salcic Z, Chong C, Javed S (2013) System-level approach to the design of a smart distributed surveillance system using systemj. ACM Trans Embedded Comput Syst (TECS) 11(4):77:1–77:24. https://doi.org/10.1145/2362336.2362344
17. Phama TTT, Lea T, Vua H, Daoa TK, Nguyen VT (2017) Fully-automated person re-identification in multi-camera surveillance system with a robust kernel descriptor and effective shadow removal method. Elsevier Image Vis Comput 59:44–62
18. Phillips PJ, Moon H, Rizvi SA, Rauss PJ (2000) The feret evaluation methodology for face-recognition algorithms. IEEE Trans Pattern Anal Mach Intell 22(10):1090–1104. https://doi.org/10.1109/34.879790
19. Phillips JP et al (2003) Face recognition vendor test 2002. In: IEEE international workshop on analysis and modeling of faces and gestures. IEEE Computer Society

20. Phillips PJ, Flynn PJ, Scruggs T, Bowyer KW, Chang J, Hoffman K, Marques J, Min J, Worek W (2005) Overview of the face recognition grand challenge. In: Proceedings of the 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05) - Volume 1, CVPR '05. IEEE Computer Society, Washington, pp 947–954. https://doi.org/10.1109/CVPR.2005.268
21. Savvides M, Kumar BVKV, Khosla PK (2004) "igenphases vs. eigenfaces". In: ICPR (3). IEEE Computer Society, pp 810–813
22. Senouci B, Charfi I, Heyrman B, Dubois J, Miteran J (2016) Fast prototyping of a soc-based smart-camera: a real-time fall detection case study. J Real-Time Image Process 12(4):649–662. https://doi.org/10.1007/s11554-014-0456-4
23. Shan Y, Hao Y, Wang W, Wang Y, Chen X, Yang H, Luk W (2014) Hardware acceleration for an accurate stereo vision system using mini-census adaptive support region. ACM Trans Embed Comput Syst (TECS) 13(4s):132:1–132:24. https://doi.org/10.1145/2584659
24. Smitha KG, Vinod AP (2015) Facial emotion recognition system for autistic children: a feasible study based on FPGA implementation. Med Biol Eng Comput 53(11):1221–1229. https://doi.org/10.1007/s11517-015-1346-z
25. Sun Y, Wang X, Tang X (2014) Deep learning face representation from predicting 10,000 classes. In: Proceedings of the 2014 IEEE conference on computer vision and pattern recognition, CVPR '14, pp. 1891–1898. IEEE Computer Society, Washington. https://doi.org/10.1109/CVPR.2014.244
26. Taigman Y, Yang M, Ranzato M, Wolf L (2014) Deepface: closing the gap to human-level performance in face verification. In: Proceedings of the 2014 IEEE conference on computer vision and pattern recognition, CVPR '14, pp. 1701–1708. IEEE Computer Society, Washington. https://doi.org/10.1109/CVPR.2014.220
27. Tian L, Fan C, Ming Y (2016) Learning spherical hashing based binary codes for face recognition. Multimed Tools Appl 76(11):13271–13299
28. Turk M (2013) Over twenty years of eigenfaces. TOMCCAP 9(1s):45:1–45:5
29. Turk M, Pentland A (1991) Eigenfaces for recognition. J. Cogn Neurosci 3(1):71–86. https://doi.org/10.1162/jocn.1991.3.1.71
30. what-when how.com: Introduction to face recognition (2014). http://what-when-how.com/face-recognition
31. Xilinx I (2012) AXI Reference Guide, vol 14. Xilinx
32. Xilinx: Vivado design suite - hlx editions (2016). www.xilinx.com/products/design-tools/vivado.html
33. Yin DBM, Omar S, Talip BA, Muklas A, Norain NAM, Othman AT (2017) Fusion of face recognition and facial expression detection for authentication: a proposed model. In: Proceedings of the 11th international conference on ubiquitous information management and communication, IMCOM '17. ACM, New York, pp 21:1–21:8. https://doi.org/10.1145/3022227.3022247

Talal Bonny received the M.Sc. degree in Computational Sciences and Engineering (CSE) from the Technical University of Braunschweig-Germany in 2002, and the Ph.D. degree in computer engineering from Karlsruhe Institute of Science (KIT), Karlsruhe, Germany in 2009. He worked as Post-Doc fellow for two years at King Abdullah University for Science and Technology (KAUST) in 2011. He joined University of Sharjah, UAE, in 2013 as assistant professor in the department of Electrical and Computer Engineering. Dr. Bonny served as reviewer/TPC member in many IEEE/ACM journals/conferences. His current research interests include embedded systems and hardware digital design, optimization algorithms, and bioinformatics.

Tamer Rabie received the M.Sc. degree in adaptive image restoration in 1993 from the Department of Electrical and Computer Engineering at The Universty of Calgary in Canada, and the Ph.D. degree in active computer vision applied to dynamic virtual environments in January 1999 from the Edward S. Rogers Sr. Department of Electrical and Computer Engineering at the University of Toronto in Canada, where, in collaboration with world-renowned computer scientist Professor Demetri Terzopoulos, they pioneered the Animat Vision Paradigm, widely cited today in the virtual vision literature. During the period from May 1998 to September 1998 Dr. Rabie held a five-month software engineer contract with the visualization and image processing group at ISG Technologies Inc. in Mississauga, Ontario, Canada, where he was involved in a critical project to develop image quality enhancements for leading edge volume rendering servers for the medical imaging industry. From October 1998 to December 1999 he held a postdoctoral fellow in the Department of Computer Science at the University of Toronto where he was involved in further research and development of the Animat Vision project. From January 2000 to August 2001 he held an assistant professor appointment in the Department of Electrical and Computer Engineering at Ryerson University in Toronto, Canada, and an associate professor position in the Faculty of Information Technology at the UAE University from September 2001 to 2011. He is currently Associate Professor of Computer Engineering at the University of Sharjah. He was also an adjunct professor at the University of Toronto, Intelligent Transportation Systems Centre, conducting collaborative research work in active computer vision-based traffic surveillance and control. His current research interests include digital image processing, computer vision, image/speech watermarking, and intelligent robotic systems. Dr. Rabie has published papers in digital image processing, computer vision, and artificial intelligence. Dr. Rabie is a Senior Member of the IEEE, and a member of the IEEE Signal Processing Society. He is also a member of the Professional Engineers of Ontario (PEO) Association in Toronto, Canada, and licensed as a professional engineer in the province of Ontario since 2001.

A. H. Abdul Hafez is an Assistant Professor at Hasan Kalyoncu University, Gaziantep, Turkey. He completed his PhD in Computer Science and Engineering (2008) from Osmania University, jointly with IIIT Hyderabad, India. His present areas of research include robotic vision, Machine learning, and vision-based control.