2016 IEEE International Parallel and Distributed Processing Symposium Workshops

Face recognition using Local Binary Patterns Histograms (LBPH) on an FPGA-based System on Chip (SoC)

Nikolaos Stekas
Delft University of Technology
Delft, Netherlands
n.stekas@student.tudelft.nl

Dirk van den Heuvel
TOPIC Embedded Systems
Best, Netherlands
dirk.van.den.heuvel@topicproducts.com

***Abstract*—The need for facial recognition systems that are fast and accurate is continuously increasing. In this paper, a face recognition implementation on a System on Chip (SoC), integrated with an FPGA, is presented. This implementation utilizes Local Binary Patterns Histograms to extract features from test face images and Manhattan distance to retrieve the correct match from the system's face database. The SoC utilized is a Zynq-7030. The feature extraction and the distance computations, between the database, are implemented on the FPGA. The ARM processor of the SoC is responsible for receiving the input stream and presenting the output result, using the acquired distances. Real-time face recognition, with an execution time of 8.6 ms and accuracy of 79%, is achieved through this implementation.**

***Keywords*—Face Recognition, FPGA, Local Binary Patterns Histograms, SoC.**

I. Introduction

Facial Recognition is a subject that researchers have focused on for years. In the past decades, the advances in technology allowed computing systems to perform face recognition. The computer applications evolved from utilizing simple semi-automated models, in the early years [1], to efficient sophisticated mathematical models, in the present. This progress in the area of facial recognition, enabled solutions in a number of different fields, varying from medical assistance to security systems. The introduction of face recognition in these areas of interest, combined to the wide spreading into industry solutions highlighted the importance of the subject and its mandatory evolution. Thus, the bar for face recognition technologies is raised constantly, with a need on implementing systems that are faster and of higher accuracy.

FPGA devices are good candidates for developing such systems, as they provide the means for parallelization of computations, low power consumption and simple integration with other processing units. Suitable design of a model for efficient use of the FPGA, can lead to the achievement of real-time face recognition.

The current paper presents a system for facial recognition on a System on Chip (Soc), containing an FPGA. The Local Binary Patterns Histograms (LBPH) model is utilized for this purpose. The proposed system constitutes a novel approach in face recognition, as implementations of LBPH in FPGAs for face recongition have not been yet reported, as known to the authors.

LBPH is utilized for the features extraction of the test image and Manhattan distance for computing the distance from the database images, loaded in the system. Those two models were chosen due to their efficiency-complexity ratio. Both of them are designed efficiently and are processed completely by the FPGA to achieve real-time speed. The ARM processor of the platform is utilized to handle the input stream of images and to output the result, depending on the distances calculated.

The outline of the paper is as follows: In section II, the System Specifications are described. In Section III, the utilization of LBPH in facial recognition is explained. In section IV, the complete implementation of the system is described and in section V, the derived results are presented. Finally, in section VI, conclusions about the completed work are drawn.

II. System Specifications

Initially, the system specifications were analyzed, for optimal decision making, during the design and implementation process.

The platform used for the system is a TOPIC Development Kit with an integrated Zynq-7030 SoC and 512MB of DDR RAM. The input stream consists of fixed-resolution, 200x200, grayscale images. The images are processed as 1x(200x200) size arrays, for convenience in computations. The face database is composed of the already extracted features from 100 images of 20 different persons (5 images per person). The described database are loaded in the DDR of the platform. The system utilizes LBPH for the face extraction and Manhattan Distance for the distance computations. Finally, the real-time goal is translated into 60 frames per second.

III. LBPH in Face Recognition

The Local Binary Patterns Histograms (LBPH), proposed in 2006 [2], is an algorithm used for face recognition, which is based on the local binary operator [3]. It is a very popular algorithm, widely used, because of its discriminative power and computation simplicity [4].

Concerning the functionality, a NxM sized image is used and is split into regions. The same size is preferred, in both width and height, resulting in mxm regions.

978-1-5090-3682-0/16 $31.00 © 2016 IEEE
DOI 10.1109/IPDPSW.2016.67

In each region, the local binary operator is utilized. This operator, when applied on images, compares a pixel to its eight closest neighbor pixels. The comparison checks if the value of the neighbor pixel is higher than the center pixel, returning a value of '1' in this case. In any other case, a '0' value is returned. Applying this process to all 8 neighbors, 8 binary values are derived. An 8-bit binary number is formed, next, by merging those values together. The obtained binary number can be translated into a decimal value, called the pixel LBP value, in a range between 0-255. The operation can be observed in Figure 1. The process described, is performed for every pixel in the region.

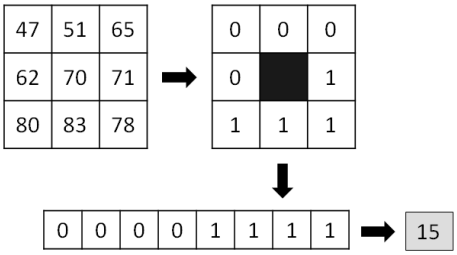

Fig. 1: LBP operator example.

The histogram for the current region is then created, by calculating the number of appearance of all the LBP values in this region. Through this process, the histogram for each region consists of 256 bins. This process is represented through the following equation:

$$H_i = \sum_{x,y} I\{LBP(J(x,y)) = i\}, i = 0, ..., 255 \qquad (1)$$

where H_i is the bin of value i, $J(x, y)$ is the (x, y) pixel of the image and I is a conditional operator, returning '1' if the statement is true or '0' otherwise. Once the histograms for all regions have been calculated, a single histogram is created by unifying all the histograms for each section. This final histogram will contain $256 * m * m$ bins and is defined as the feature vector of the image.

To proceed with facial recognition, a database (db) of feature vectors corresponding to face images is required. The distances between the different feature vectors are then calculated. The image that corresponds to the feature vector with the closest distance is returned as the best match for the probe image. This can be derived from the following equation:

$$(M, Index) = min(dst[\text{db size}]) \qquad (2)$$

where dst is the array of the calculated distances and min is a function returning the minimum value (M) of the array and the value's index ($Index$)

A threshold is also applied to exclude faces that do not correspond to any of the faces in the database.

$$Output = \begin{cases} "Index \text{ is a match}", & \text{if } M < Threshold \\ "\text{No match}", & \text{otherwise} \end{cases} \qquad (3)$$

As stated above, Local Binary Patterns is a very popular model, with multiple, differently modified, implementations over time [5], [6]. Efficiently accelerated implementations, by utilizing different means such as GPUs and SIMD instructions, have been reported in the past [7], [8], [9]. The only FPGA implementations of LBPH known to the authors are [10] and [11], both concerning face detection. Therefore, the proposed system introduces a novel LBPH implementation for face recognition.

IV. Implementation

In the current section the implementation of the proposed system will be explained thoroughly.

A. Model Development

The utilization of LBPH is followed by necessary decisions for developing the overall model for the proposed system. Those decisions come down to the choice of the number of the regions used in the images, the weight of each region and the model that is going to be used for measuring the distances between the feature vectors.

Regarding the number of regions, the increment of the number of regions leads to a larger size of the feature vector, more calculations for the distance between two feature vectors, but also to a higher accuracy. Through a set of experiments for different number of regions, the 10x10 segmentation was selected as it returns a good trade-off between accuracy and feature vector size. In this point, it should be mentioned that, due to this segmentation, each region will include 400 pixels. Therefore, the value of a bin in the histogram varies between 0-400 and two byte values are needed for each histogram bin to cover this range.

It is expected that in an image some regions play a more important role in the recognition process. Hence, proper weights should be added to the different regions, in order to retrieve a better recognition accuracy. Taking into account the weight selection presented in [2] and by performing a set of experiments, the weights presented in Figure 2 are applied.

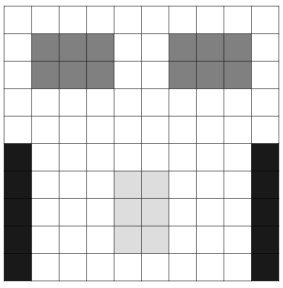

Fig. 2: Applied region weights. Black regions indicate weight 0.5, dark gray 4.0, light gray 2.0 and white 1.0.

Next, a model for calculating the distance between the feature vectors is needed. By making a comparison between the most popular models: Euclidean distance, Chi-square and Manhattan distance; similar distance deviation is observed. Thus, the Manhattan Distance [12] is chosen for calculating the distance between the feature vectors, as the needed computations can be implemented more efficiently in the FPGA, compared to the other models that require division operators.

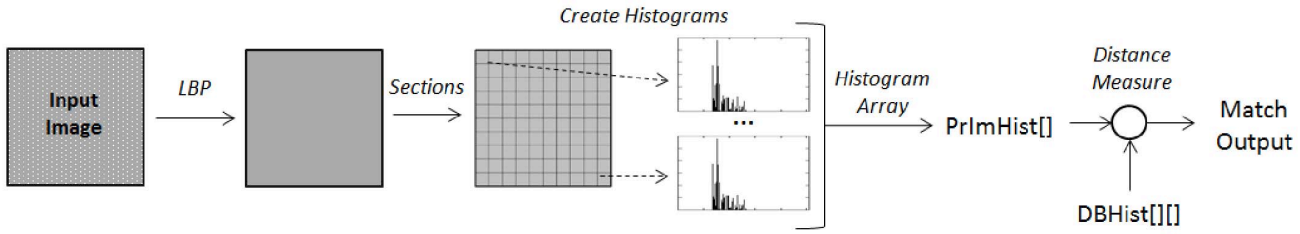

Fig. 3: Overview of LBPH for face recognition (PrIMHist[], DBHist[][] are feature vectors of the probe image and the database images respectively.)

The Manhattan distance $\Delta(x, y)$, between two feature vectors x and y is:

$$\Delta(x, y) = \sum_{i=0}^{255x10x10} |x_i - y_i| \quad (4)$$

Finally, classification is used for the database images that correspond to the same person. Through classification, the matching process becomes less inaccurate as it takes in account all the images of a person in order to retrieve a best match. This way, for N number of classes, c_n, of size s the distance is given from the following equation:

$$\Delta(x, c_n) = \frac{\sum_{i=0}^{s} \delta(x, y_i)}{s}, n = 1, ..., N \quad (5)$$

Moreover, through classification the average histogram of the images in the same class can be computed and reduce the distance calculations needed by the size of the classes. In the current system, the distance measure calculations can reduce five times, as five images per person are included.

B. System Overview

The complete system overview is presented in Figure 4. In more detail, the source of the input stream is not specific and is depending on the user. The same applies to the output, which can be just a print message in the console or an illustration on a screen connected to the platform.

The average feature vectors of the 20 different persons, consisting the database, will be computed and loaded on the DDR, prior to the utilization of the system. The total size of the DDR occupied for this purpose will amount to 20x25600x2=2560000=0.9765 Mbytes (0.0019% of the total DDR size)

Concerning the FPGA part, all blocks were designed using VHDL. The communication between the DDR and the FPGA has been established through the AXI Direct Memory Access (DMA) IP block, provided by Xilinx. Through DMA, 32-bit words are transferred within one clock cycle [1]. In the following subsections, IV-C and IV-D, the block design will be explained in depth.

[1]Presently, the maximum burst size of DMA is 256-bits. In the current system the 32-bit size burst is utilized with the maximum burst to be examined in the future scope.

C. LBPH implementation

In this subsection, the complete LBPH block will be described. The input of this block will be a continuous stream of 32-bit words that will eventually transfer the complete array corresponding to the probe image. Therefore, the limitation of processing only a word per clock cycle is reported. The output of the LBPH block will be the complete feature vector of the probe image.

The block can be divided into two main parts:

1) Compute the LBP value of the pixel.
2) Locate the correct histogram bin and increment its value.

The first part's computational simplicity should not incommode the implementation process. The delay in this part is caused by awaiting the neighbor pixels to be loaded, in order to proceed with the computation. Once the computation is completed, threshold blocks are utilized to return the binary values for each comparison. Next, a demultiplexer is used, in order to merge these numbers in an 8-bit binary, LBP value.

Concerning the second part, block ram is used to save temporarily the bin values. The issue here is to locate the bin which needs to be increased. The computed LBP value can be utilized to locate the address, but only within a region. To locate the region starting address, conditional statements are applied. The following piece of pseudo-code checks whether the examined pixel belongs in region 1 of the image:

```
i= input number;
if i<4000:
    if modulo(i,200)<200:
        region=1;
    else ...
else ...
```

Listing 1: Inspecting if i-pixel is located in the 1st region

The modulo operator, utilized during the conditional statements, is prohibitory in the FPGA design as its computation takes a high number of clock cycles to complete, which causes severe deceleration for the system. Therefore, a different computation approach for modulo is proposed, in order to overcome this issue. In the beginning, range blocks are used to check in which 200-range i is found. All those blocks can be executed in the same cycle and 200 of them are utilized,

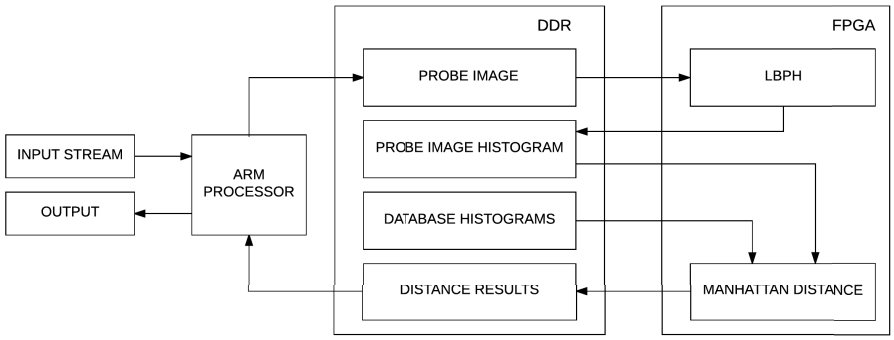

Fig. 4: System Overview

as the i value limit is 39999. These range blocks return a '1' if i is in this range and '0' otherwise. At the same time the lower bound value of the range block is subtracted from i. This value is the modulo(i,200), when i is in this range. Finally, a demultiplexer uses the binary value of the range block to drive the proper subtraction result to the output. This process can be observed in Figure 5.

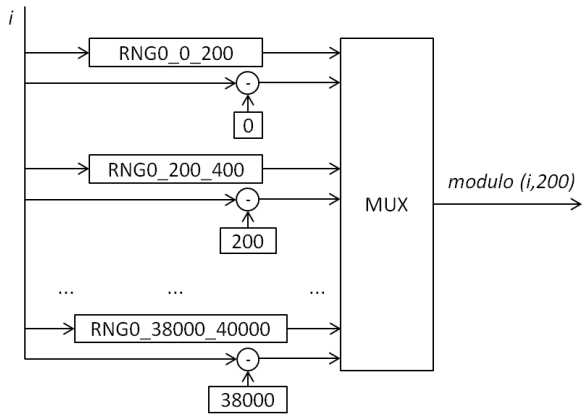

Fig. 5: Design of the modulo(i,200) operation.

The result of the first conditional statement, observed in Listing 1, alongside with the LBP value and the result of the modulo operation, will load the correct bin and increment its value. Eventually, by completing this process, the entire image's histogram is formed. This histogram consists the feature vector of the image, which is returned in the DDR as the output. This block implementation reduces the execution time of LBPH to an adequate point, for reaching real-time execution.

D. Manhattan Distance Implementation

The Manhattan distance between the newly acquired feature vector and the feature vectors of the database, needs be computed next. This will be performed in the current block for each feature vector of the database individually.

Again, the main limitation that arises is that only one word can be loaded at each clock cycle. Besides this struggle, the design of this block is quite simple. Five blocks are used: a counter to keep track of the number of the received words, a weight block to set the proper weight depending on the region, a multiplexer to drive the input values to the correct route, a weighted subtraction block to subtract the two values and multiply them with the correct weight and last, an absolute adder block, where the positive value of the subtraction will be added to the existing value. In addition, in the absolute adder block, the counter output is used to return the final result, when all input words have been processed. The above description of the overall Manhattan distance block is depicted in Figure 6

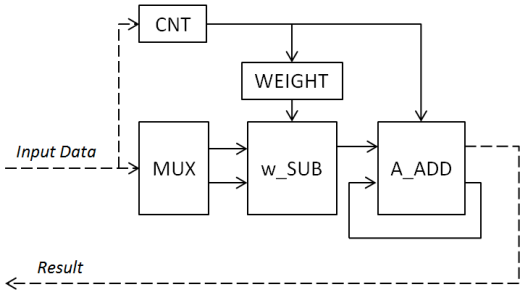

Fig. 6: Manhatan distance block overview.

The proposed design, four clock cycles are needed to compute the Manhattan distance between two words, which lays within the real-time boundaries, considering the feature vectors length and the database size in which this operation will be applied.

V. RESULTS

The experimental results of the implementation are presented in this section. As mentioned before, the TOPIC Development Kit was used as the experimental platform. The database consisted of 100 images of 20 different people, from which the average feature vector of each person was extracted and loaded in the DDR memory belonging to the platform. The

TABLE I: Execution Time Results

	LBPH	MD	Overall
Time(ms)	2.602	5.137	8.572

database was obtained from the Unconstrained Facial Images (UFI) database [13]. Example images, used for one of the persons of the database, are presented in Figure 7. Through the UFI database, 300 Test Images were also obtained. These images were loaded in the SD card of the platform and were streamed through the CPU to the DDR. Finally, the matching results were stored in an array.

Fig. 7: Images of a person in the database [13]

The whole process was timed, in order to determine if the real-time goal of 0.016 seconds was met as expected. The overall execution time was 8.572 ms, as represented in Table I. Consequently, the real-time performance is fulfilled. Moreover, compared to other real-time implementations, the following findings were reported. In [9], the feature extraction time was 25.9 ms, for a 128x128 size image, while in the proposed implementation 2.602 ms were required for a 200x200 size image. Also, a higher execution time of 17 ms is reported in [8], for an image size of 130x150.

Concerning the accuracy of the implementation a 79.33% percentage of correct recognition is reported with 238/300 correct recognitions. This value indicates a high accuracy system for this type of database, first rank[2] facial recognition [14]. The the number of regions selected, is justified through these results. In addition, better accuracy is achieved compared to [9], where 74.75% and 79% is reported for 8 and 16 sub-histograms respectively.

Finally, the resources utilization of the FPGA is reported in table II. The margin for further utilization of resources is reported. This can be exploited to accelerate further the implementation or to design blocks for varying system specifications.

VI. Conclusion

Advances in technology benefited the field of facial recognition by enabling solutions that were unfeasible in the past.

[2]In facial recognition the rank indicates the number of best-matching images among which the correct recognition is found. In this system only the first best-match is examined placing the accuracy results in first rank recognition

TABLE II: Resources utilization

Resource	Percentage
BRAM	42.82%
Slice LUTs	10.95%
Slice Registers	6.15%

Systems characterized by high accuracy and low execution time are now demanded for face recognition.

The system described in the current paper performs face recognition and integrates those characteristics. Local Binary Patterns Histograms (LBPH) was used for feature extraction and Manhattan distance for the matching process. The two models were designed and implemented efficiently on an FPGA-based System on Chip. In the end, the evaluation of the system reported an execution time of 8.6 ms and an accuracy of 79.33%, confirming its efficiency.

References

[1] W. W. Bledsoe, "The model method in facial recognition," *Technical Report PRI 15, Panoramic Research, Inc., Palo Alto, California.*, 1964.

[2] H. A. Ahonen, T. and M. Pietikinen, "Face description with local binary patterns," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 12, pp. 2037–2041, 2006.

[3] P. M. Ojala, T. and D. Harwood, "A comparative study of texture measures with classification based on feature distributions," *Pattern Recognition*, vol. 19, no. 3, pp. 51–59, 1996.

[4] M. Pietikinen, "Local Binary Patterns," vol. 5, no. 3, p. 9775, 2010.

[5] D. Maturana, D. Mery, and . Soto, "Face recognition with local binary patterns, spatial pyramid histograms and naive bayes nearest neighbor classification," in *Chilean Computer Science Society (SCCC), 2009 International Conference of the*, Nov 2009, pp. 125–132.

[6] Y. He, N. Sang, and R. Huang, "Local binary pattern histogram based texton learning for texture classification," in *Image Processing (ICIP), 2011 18th IEEE International Conference on*, Sept 2011, pp. 841–844.

[7] A. H. Roman Jurnek, Pavel Zemck, "Implementing the local binary patterns with simd instructions of cpu," *Proceedings of Winter Seminar on Computer Graphics*, p. 5, 2010.

[8] S. C. Tek and M. Gkmen, "Cuda accelerated face recognition using local binary patterns." *Proceedings of Winter Seminar on Computer Graphics*, 2012.

[9] C. Dwith and G. Rathna, "Parallel implementation of lbp based face recognition on gpu using opencl," *Parallel and Distributed Computing, Applications and Technologies (PDCAT), 2012 13th International Conference on*, pp. 755–760, Dec 2012.

[10] M. G. Tomasz Kryjak, Mateusz Komorkiewicz, "Fpga implementation of real-time head-shoulder detection using local binary patterns, svm and foreground object detection," *Design and Architectures for Signal and Image Processing (DASIP), 2012 Conference on*, pp. 1–8, 2012.

[11] S. Jin, D. Kim, T. T. Nguyen, B. Jun, D. Kim, and J. W. Jeon, "Application-specific systems, architectures and processors, 2009. asap 2009. 20th ieee international conference on," pp. 61–66, July 2009.

[12] E. F. Krause, "Taxicab geometry: An adventure in non-euclidean geometry," 1986.

[13] L. Lenc and P. Král, "Unconstrained Facial Images: Database for face recognition under real-world conditions," in *14th Mexican International Conference on Artificial Intelligence (MICAI 2015)*. Cuernavaca, Mexico: Springer, 25-31 October 2015 2015.

[14] "Baseline results on the feret database," 2010. [Online]. Available: http://www.cs.colostate.edu/evalfacerec