An Approach of Face Recognition Applied for Smarthome using System – On – Chip Technology

Minh Son Nguyen
Faculty of Computer Engineering
University of Information Technology
HoChiMinh City, Vietnam
e-mail: sonnm@uit.edu.vn

Ngo Van Huynh, Dai Duong Tran, Hieu Truong Ngo
Faculty of Computer Engineering
University of Information Technology
HoChiMinh City, Vietnam
e-mail: duongtd@uit.edu.vn, truongnh@uit.edu.vn

***Abstract*— This paper presents a real-time face recognition system consisting of face detection and recognition using an System-on-Chip (SoC) Technology. Our system provides an integration of hardware and software solution for face recognition that receives video input from a camera, detects the locations of the faces using the Haar Like Feature proposed by Viola – Jones, subsequently recognizes each face using Local Binary Pattern Histogram (LBPH) algorithm. Beside, a subsystem of using Histogram of Oriented Gradients (HOG) feature combine with Support Vector Machine (SVM) classifier to recognize human body before face detection. Experimental results show maximum execution time not exceeded 220ms and success rate of face recognition up to 80% within 1 meter around camera on SoC Cyclone-V from Terasic DE10-NANO development board.**

Keywords - Face detection, Face recognition, SoC, HoG, LBPH, Haar Feature, SVM.

I. Introduction

Face recognition based on the geometric features of a face is probably the most intuitive approach to face recognition. A number of face recognition algorithms have been developed in the past decades with various hardware implementations and applied in many different fields as surveillance, security, biometric [1],[2],[3]. Face recognition is a challenging research area in terms of both software (developing algorithmic solutions) and hardware (creating physical implementations) [4],[5]. A smart camera is a camera that has the ability not only to take pictures but also, more importantly, to make intelligent decisions of what is happening in the image and in some cases, take appropriate actions on behalf of the camera user.

In this paper, we research and design a smart camera with capture images, detect face in each frame, and send only the detected face images to the face recognition system which identifies the face images. The face features are detected by Haar-Like and face image recognized by Local Binary Pattern Histogram Algorithm. Besides, the system can detect body in image as well. For recognition phase, a dataset consisting 30 face images to experiment by ourselves training.

This paper is organized as follows. The related works are shown in the second section. In the third section, the face detection and recognition algorithms are explained in order to apply for Smarthome application using SoC Architecture. The next section shows the SoC Architecture consisting of hardware and software integration. In Section 5, the implementation and experimental result of face detection and recognition algorithms on SoC hardware are shown in DE10-Nano development kit. The paper ends with a short summary of related works and conclusion of this article.

II. RELATED WORKS

Miguel Contreras's thesis [8] develops a smart camera applied on a Terasic DE0 FPGA development board. The case study developed the global vision system for a robot soccer implementation. The algorithm identified and calculated the positions and orientations of each robot and the ball. The techniques demonstrated in this thesis are meant as a guide to help others new to the field to quickly and efficiently develop their own projects. With proper usage, these techniques can be used to simplify the development process and reduce the development time for future FPGA-based smart cameras.

An FPGA based embedded vision system for face detection [9] is presented by Michael Drozdz. The sliding detection window, HOG+SVM algorithm and multi-scale image processing were used and extensively described. The applied computation parallelizations allowed to obtain real-time processing of a 1280 - 720 @ 50Hz video stream. The presented module has been verified on the Zybo development board with Zynq SoC device from Xilinx. It can be used in a vast number of vision systems, including diver fatigue monitoring.

Janabek Matai et al. [10] presented a complete real-time face recognition system consisting of one face detection, one recognition and a downsampling module using an FPGA. Their system provides an end-to-end solution for face recognition; it receives video input from a camera, detects the locations of the face(s) using the Viola-Jones algorithm, subsequently recognizes each face using the Eigenface algorithm, and outputs the results to a display. Experimental results show that complete face recognition system operates at 45 frames per second on a Virtex-5 FPGA.

DreamCam [11]: a modular smart camera constructed with the use of an FPGA like main processing board. The core of the camera is an Altera Cyclone-III associated with a

978-1-7281-4723-9/19/$31.00 ©2019 IEEE
DOI 10.1109/ACOMP.2019.00033

CMOS imager and six private Ram blocks. The main novel feature of their work consists in proposing new smart camera architecture and several modules (IP) to efficiently extract and sort the visual features in real time. Extraction is performed by a Harris and Stephen filtering associated with customized modules.

III. FACE DETECTION AND RECOGNITION ALGORITHMS

A. Face Detection Algorithm

Object Detection using Haar feature-based cascade classifiers is an effective object detection method proposed by Paul Viola and Michael Jones in their paper, "Rapid Object Detection using a Boosted Cascade of Simple Features" in 2001 [6].

Each feature is a single value obtained by subtracting sum of pixels under the white rectangle from sum of pixels under the black rectangle. Computing the rectangle features in a convolutional kernel style can be long, very long. For this reason, the authors, Viola and Jones, proposed an intermediate representation for the image: the integral image. The role of the integral image is to allow any rectangular sum to be computed simply, using only four values.

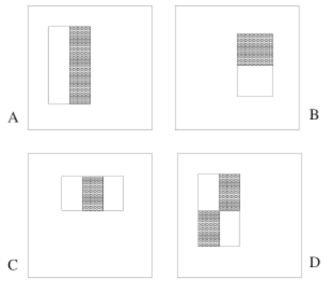


Figure 1. Haar-Like Feature

Suppose to determine the rectangle features at a given pixel with coordinates (x, y). Then, the integral image of the pixel in the sum of the pixels above and to the left of the given pixel.

$$ii(x,y) = \sum_{x' \le x, y' \le y} i(x', y') \qquad (1)$$

, where ii(x, y) is the integral image and i(x, y) is the original image. There is a form a recurrence which requires only one pass over the original image, define the following pair of recurrences:

$$s(x.y) = s(x,y\text{-}1) + i(x,y) \qquad (2)$$

$$ii(x,y) = ii(x\text{-}1,y) + s(x,y) \qquad (3)$$

, where $(x,\ y)$ is the cumulative row sum and $(x, -1) = 0$ & $ii(-1,y)$=0).p

With an image of 24x24, results over 160000 features. How do we select the best feature out of 160000 features? It is achieved by Adaboost and Cascade of Claasifiers. The features are grouped into different stages of classifiers and applied one-by-one. If a window fails in the first stage, discard it. We do not consider the remaining features on it. If it passes, apply the second stage of features and continue the process. The window which passes all stages is a face region.

B. Face Recognition Algorithm

The Local Binary Patterns methodology has its roots in 2D texture analysis. The basic idea of Local Binary Patterns is to summarize the local structure in an image by comparing each pixel with its neighborhood. Take a pixel as the center and threshold its neighbors against. If the intensity of the center is greater-equal its neighbor, then denote it with 1 and 0 if not. End up with a binary number for each pixel, for example, 11001111. So with 8 surrounding pixels you will end up with 2^8 possible combinations, called Local Binary Patterns or sometimes referred to as LBP codes. The first LBP operator described in literature actually used a fixed 3 x 3 neighborhood just like this:

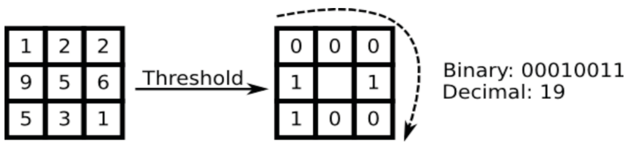


Figure 2. An example of the basis LBP operator.

A more formal description of the LBP operator can be given as:

$$LBP(x_c, y_c) = \sum_{p=0}^{p-1} 2^p s(i_p\text{-}i_c) \qquad (4)$$

, with (x_c, y_c)as central pixel with intensityi_c, and i_n being the intensity of the neighbor pixel, s is the sign function defined as:

$$s(x) = \begin{cases} 1\ if x\ \ge 0 \\ 0\ else \end{cases} \qquad (5)$$

This description enables you to capture very fine grained details in images. In fact the authors were able to compete with state of the art results for texture classification. Soon after, the operator which was published was noted, that a fixed neighborhood fails to encode details differing in scale. So the operator was extended to use a variable neighborhood in [1]. The idea is to align an arbitrary number of neighbors on a circle with a variable radius, which enables to capture the following neighborhoods:

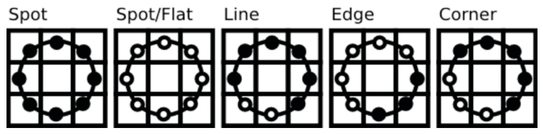


Figure 3. Examples of LBP neighborhood operator circle.

For a given Point (x_c, y_c) the position of the neighbor $(x_p, y_p), p \in P$ can be calculated by:

$$x_p = x_c + R\cos\left(\frac{2\pi p}{p}\right) \quad (6)$$

$$y_p = y_c - R\sin\left(\frac{2\pi p}{p}\right) \quad (7)$$

, where R are the radius of the circle and P are the number of sample points.

The operator is an extension to the original LBP codes, so it's sometimes called Extended LBP (also referred to as Circular LBP). If a point coordinated on the circle does not correspond to image coordinates, the point gets interpolated. Computer science has a bunch of clever interpolation schemes; the OpenCV implementation does a bilinear interpolation:

$$f(x,y) \approx [1 - xx]\left[\frac{f(0,0)f(0,1)}{f(1,0)f(1,1)}\right]\left[\frac{1-y}{y}\right] \quad (8)$$

So what's left to do is how to incorporate the spatial information in the face recognition model. The representation proposed by Ahonen et. al [1] is to divide the LBP image into $\mathfrak{m}$ local regions and extract a histogram from each. The spatially enhanced feature vector is then obtained by concatenating the local histograms. These histograms are called Local Binary Patterns Histograms.

IV. SOC ARCHITECTURE FOR FACE DETECTION AND RECOGNITION

A. *SoC Hardware Architecture*

The SoC Architecture of this research is applied on a Terasic DE10-NANO development board, a robust hardware platform is built around the Intel FPGA as below Figure 4, which combines the latest dual-core Cortex-A9 and 720p USB Camera.

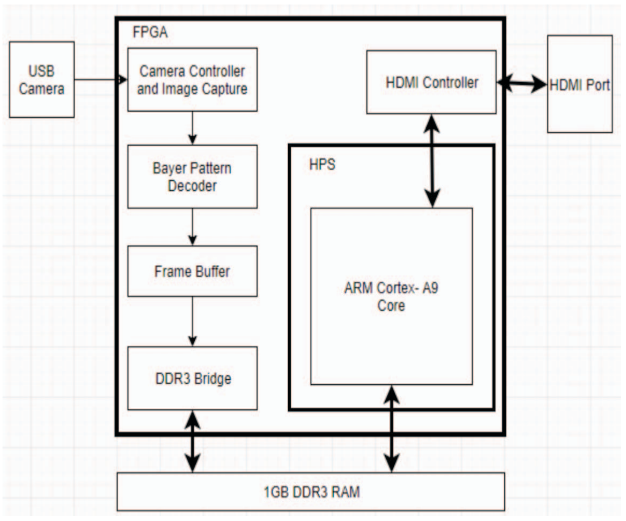


Figure 4. Block Diagram of SoC Architecture

Camera Controller: configure USB Camera and it gets the valid pixels from camera sensor using the sync signals respectively.

Bayer Pattern: demosaicking transforms the bayer image format to grayscale image.

Frame Buffer: gives access to the current frame data, in order to perform primary processing without accessing external DDR3 RAM.

DDR3 Bridge: allows to access external DDR3 RAM from different processing modules. It allows the use of a shared memory schema in clock domains.

B. *Software Architecture integrated on SoC Architecture*

The software architecture of this system is shown in Figure 5:

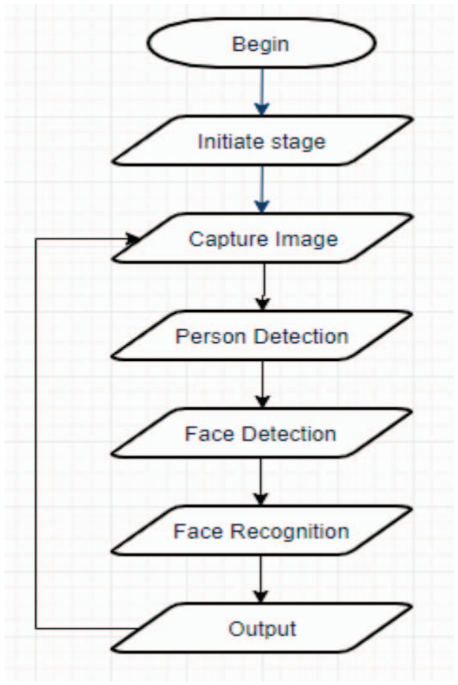


Figure 5. Software flow of this system implemented on SoC architecture

Initial stage: system initiates and configures peripheral related.

Capture image: capture image on camera and send to other function.

Person detection: system automatically identifies person from captured frame.

Face detection: detect whether a face has appeared in frame has been sent. System analytics to detect the shape of a face, noses, space between the eyes. Once a face is detected, system immediately sends data to recognition subsystem to analyze and process the data.

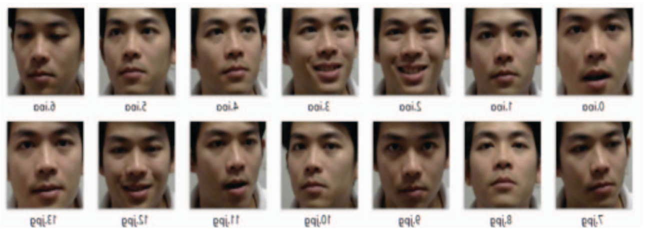


Figure 6. Dataset used to train the classifier

Face recognition: with integrated algorithm and trained dataset, output the results to a display is known or unknown face.

V. IMPLEMENTATION OF THE FACE DETECTION AND RECOGNITION USING SYSTEM – ON – CHIP

A. Implementation

The input dataset consisting 30 face images used to train the cascade classifier, with resolution scale to 150x150 pixels. It takes 1.5 hours for the training dataset to xml file contain face features of author on CPU Intel Core i7-6700 2.6GHz, and transferred to SoC. Language used for functions in system is C++.

Face detection: detected faces using OpenCV library and Haar-Like feature. The optimum distance from the camera which are obtained from about 0.5 m – 1 m.

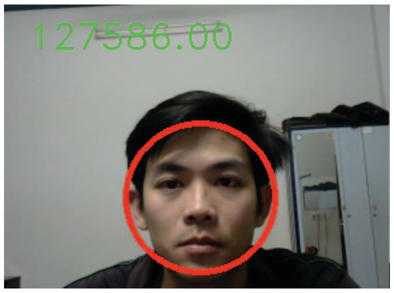


Figure 7. Face detected

Face recognition: with cascade classifier and trained dataset, system shows the face recognized by green circle as in Figure 8.

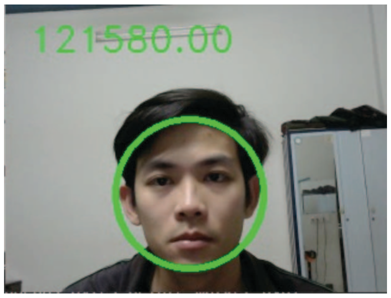


Figure 8. Face recognized

The system can detect and recognize two faces at the same time appear in front of camera.

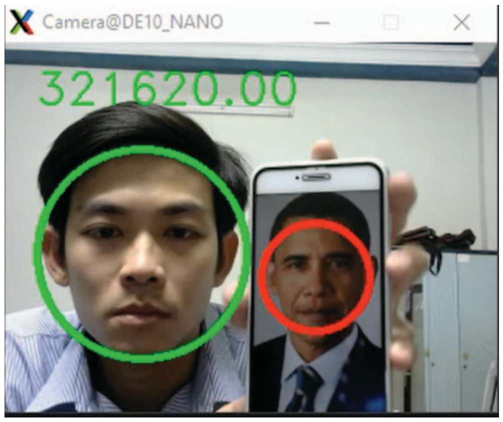
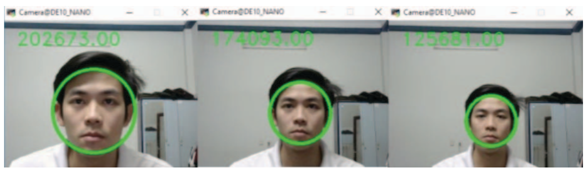


Figure 9. Obama's face is detected and author's face is recognized.

B. Experiment and Comparison

Figure 9 gives three distances between face and camera used in experiment of face recognition system: 0.5 meter, 0.7 meter and 1 meter. Besides, experiment is extended with three distances for comparison: 1.1 meter, 1.3 meter, 1.5 meter.



a) 0.5 meter b) 0.7 meter c) 1 meter

Figure 10. Distance from camera to face

Table 1 below is the experiment of system. The table contains the execution time of algorithms to recognition faces in milliseconds (ms). The numbers of experiments performed are 50 times in conditions of sufficient light, no obstacles environment and face appears fully in the frame of camera. Then, the results on the test data will be taken as the average to determine the relative results. Abbreviations used in the table:

Ex. Time: Execution time for face recognition module, calculated by milliseconds (ms)

Result: Result of face recognition module (Y: successful recognition, N: failure recognition).

So, with distance of 0.5 meter: system shows maximum execution time not exceeded 220 ms, with success rate reached 40 over 50 times experiment. In far distances, 0.7 meter and 1 meter, success rate drop hardly, half of experiments detected without recognized face. Besides, in far distance, the number of Haar features on frame have face decreases a little, the same Haar features between face on the camera and face in dataset will be less but the matching still within allowable level, so execution time will be faster.

TABLE 1. EXPERIMENT OF 50 TIMES EXECUTING FACE RECOGNITION MODULE WITH DISTANCES FROM 0.5 METER TO 1 METER

Nº	Experiment					
	0.5 meter		0.7 meter		1 meter	
	Result	Ex. Time (ms)	Result	Ex. Time (ms)	Result	Ex. Time (ms)
1	Y	210	Y	165	Y	127
2	Y	214	N	174	Y	133
3	N	219	Y	170	N	148
...	...	...	...	...	...	...
49	Y	220	Y	160	Y	136
50	Y	214	N	170	N	137
Total	40/50	209.88	32/50	166.18	23/50	135.58

TABLE 2. EXPERIMENT OF 50 TIMES EXECUTING FACE RECOGNITION MODULE WITH DISTANCES FROM 1.1 METER TO 1.5 METER

Nº	Experiment					
	1.1 meter		1.3 meter		1.5 meter	
	Result	Ex. Time (ms)	Result	Ex. Time (ms)	Result	Ex. Time (ms)
1	Y	140	N	97	N	91
2	Y	135	N	120	N	103
3	N	131	Y	138	N	99
...	...	...	...	...	...	...
49	N	129	N	96	N	106
50	Y	139	Y	135	N	110
Total	20/50	133.42	7/50	115.02	0/50	99.1

With distance between face and camera exceeds 1 meter, success rate of face recognition function decrease hardly, only face detection function working with execution time about 90 ~ 110 ms.

TABLE 3. SUMMARY OF EXPERIMENT IN TABLE 1 & 2

Face Recognition System with LBPH algorithm		
Distance	Average Execution Time (ms)	Average Success Rate
0.5 meter	209.88	80 %
0.7 meter	166.18	64 %
1 meter	135.58	46 %
1.1 meter	133.42	40 %
1.3 meter	115.02	14 %
1.5 meter	99.1	0 %

Then, we have implemented the system using Eigenface algorithm proposed by [7] with the same dataset, distance and experiment times.

Eigenface finds a mathematical description of the most dominant features of the training set as a whole. LBPH analyzes each face in the training set separately and independently. In LBPH each image is analyzed independently, while the Eigenface method looks at the dataset as a whole. Consequently, execution time of LBPH are faster than Eigenface method when implemented on our system.

TABLE 4. SUMMARY OF EXPERIMENT OF PROPOSED SYSTEM WITH EIGENFACE METHOD

Face Recognition System with Eigenface algorithm		
Distance	Average Execution Time (ms)	Average Success Rate
0.5 meter	250.36	82 %
0.7 meter	181.2	66 %
1 meter	149.2	54 %
1.1 meter	150.32	44 %
1.3 meter	102.4	0 %
1.5 meter	104.74	0 %

Figure 10 below is comparison between 2 experiment systems using LBPH algorithm and Eigenface method.

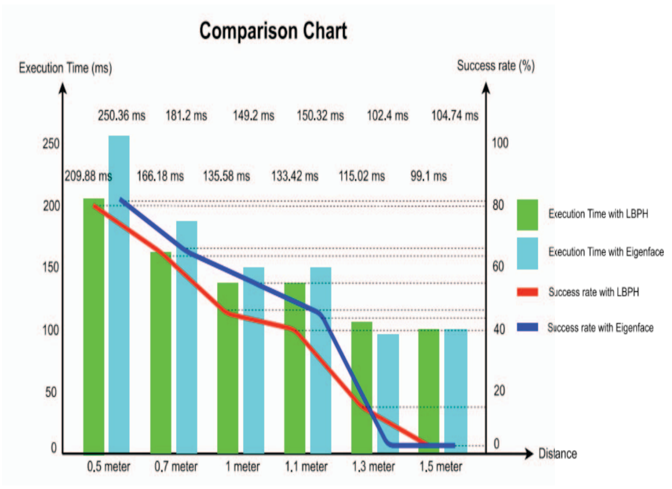


Figure 10. Comparison chart of two experiment systems using LBPH and Eigenface algorithm.

About body detection, HOG algorithm and SVM classifier integrated in the system also produce positive results. But in some cases, the system could not recognize correctly with some images include the body as shown in Figure 11.

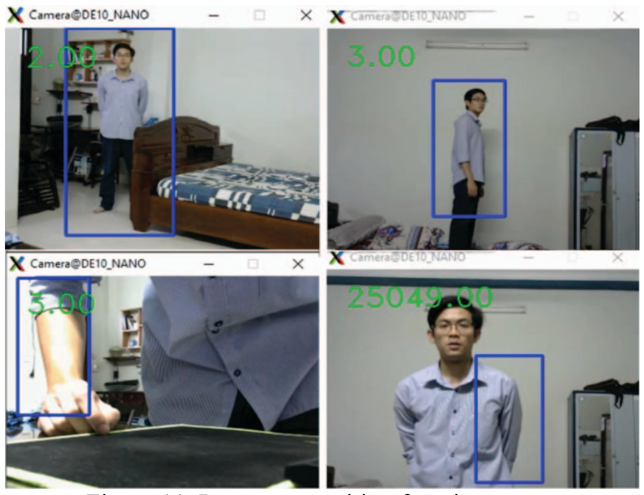


Figure 11. Person recognition function

VI. Conclusion

This paper presented the design and implementation of a real-time face recognition system which using a System-on-Chip (SoC) Technology. Two modules are face detection and face recognition which are designed and implemented on a SoC Cyclone V from Terasic DE10-NANO development board. Experimental results of the face recognition not only show that system can recognize author or the unknown's face clearly with success rate up to 80% on condition maximum distance is 1 meter from camera to face, but also reveal that execution time of LBPH algorithm are faster than Eigenfaces method proposed by [7]. Drawback of current system is whenever distance between camera and face exceed 1 meter, face recognition function become useless, this will need further improvement in the future.

Acknowledgment

This research is funded by **University of Information Technology**-Vietnam National University HoChiMinh City under grant number D1-2019-18.

References

[1] T. Ahonen, A. Hadid, M. Pietikainen (2004), "Face Recognition with Local Binary Patterns", *Computer Vision - ECCV 2004*, LNCS 3021.

[2] D. Chen and H. Jiu-qiang, "An fpga-based face recognition using combined 5/3 dwt with pca methods," in *Journal of Communication and Computer*, vol. 6, Oct 2009.

[3] H. T. Ngo, R. Gottumukkal, and V. K. Asari, "A flexible and efficient hardware architecture for real-time face recognition based on eigenface," in *IEEE Computer Society Annual Symposium* on VLSI, 2005.

[4] R. Gottumukkal, H. T. Ngo, and V. K. Asari, "Multi-lane architecture for eigenface based real-time face recognition," in *Microprocessors and Microsystems*, 2006, p. 216224.

[5] A. P. Kumar, V. Kamakoti, and S. Das, "System-onprogrammable-chip implementation for on-line face recognition," *Pattern Recognition Letters*, vol. 28, pp. 342–349, 2007.

[6] Paul Viola, Michael Jones (2001), "Rapid Object Detection using a Boosted Cascade of Simple Features", *Conference on Computer Vision and Pattern Recognition*, CVPR 2001.

[7] M. Turk, A. Pentland (1991), "Eigenfaces for recognition", *Journal of Cognitive Neuroscience 3*.

[8] D Miguel Contreas (2016) "Design of an FPGA-Based Smart Camera and its Application Towards Object Tracking", *Master Thesis, Massey University*, Manawatu. New Zealand.

[9] Michal Drozdz, Tomasz Kryjak (2016), "FPGA implementation of multi-scale face detection using HOG features and SVM claassifier" *Image Processing & Communications,* vol. 21, no. 3, pp.27-44.

[10] Janarbek Matai, Ali Irturk, Ryan Kastner (2011), "Design and Implementation of an FPGA-based Real-time Face Recognition System" *Thesis, University of California*, San Diego, United States.

[11] Merwan Birem, Francois Berry (2014), "DreamCam: A modular FPGA-based smart camera architecture", *Journal of Systems Architecture,* Elsevier, 2014, 60 (6), pp. 519 – 527.