FPGA ARCHITECTURE FOR FACIAL-FEATURES AND COMPONENTS EXTRACTION

Nadia NACER[1], Bouraoui MAHMOUD[2], Mohamed Hedi BEDOUI[3]

1,3 Laboratory of Medical Technology and Imaging, Faculty of Medicine at Monastir, University of Monastir – Tunisia
nd.nacer@gmail.com, Medhedi.bedoui@fmm.rnu.t

2 Laboratory of Medical Technology and Imaging (FMM-Monastir), National Engineering School at Sousse,
University of Sousse-Tunisia
bouraoui.mahmoud@eniso.rnu.tn

ABSTRACT

Several methods for detecting the face and extracting the facial features and components exist in the literature. These methods are different in their complexity, performance, type and nature of the images and the targeted application. The facial features and components are used in security applications, robotics and assistance for the disabled. We use these components and characteristics to determine the state of alertness and fatigue for medical diagnoses. In this work we use plain color background images whose color is different from the skin and which contain a single face. We are interested in FPGA implementation of this application. This implementation must meet two constraints, which are the execution time and the FPGA resources. We have selected and have associated a face detection algorithm based on the skin detection (using the RGB space) with a facial-feature extraction algorithm based on tracking the gradient and the geometric model.

KEYWORDS

Face detection, face components, face features, skin detection, RGB, gradient, implementation, FPGA

1. INTRODUCTION

The localization of facial features and components is an important step in many applications of security (biometrics, surveillance), robotics, assistance for disabled (face communication) and driving safety (detection of decreased alertness, fatigue). The location of these features allows the facial expressions analysis. The facial expressions consist in a temporary distortion of a facial structure. There are two types of facial structure: permanent (mouth, eyes, hair, deep wrinkles, and brow) and transient (wrinkles, swelling). We use these components and features to determine the state of alertness and fatigue for medical diagnoses.

The embedded systems for acquiring, processing and analyzing facial expressions require increasingly complex algorithms. These algorithms require computational power and a large memory space exceeding the possibilities offered by most conventional processors. The FPGA (Field Programmable Gate Array) offers a solution that combines the programming flexibility and with the specialized-architecture power. The implementation of application of acquiring, processing and analyzing the facial expressions must meet a real-time execution (the camera frame rate), while minimizing the resource consumption if we aim at low cost systems.

DOI : 10.5121/ijcseit.2013.3204

Several studies in the literature have proposed face detection implementation son processor and on FPGA. The authors in [1] proposed a face detection implementation (Haar-classifier) on FPGA (Viertx5Xilinxfamily) of images (320*240) with a time execution of 34.7ms (28.8 frames/s). The same application implemented on a computer (Intel Core 2 CPU (2.80 GHz), 2.98 GB DDR2 SDRAM (800 MHz), Windows XP Professional, and Visual Studio) allows treating 0.71 frames/s [1]. The work done by [2] is a face detection parallel implementation (Convolutional Face Finder) on FPGAs (Virtex 4) designing 32 Elementary Processors (PE) that operate at a frequency of 350MHz. In [3] the authors implemented a face detection algorithm (color model) for an image (800*600) on FPGA (Virtex II) which was used to treat up to 90frames/s. In [4] the authors made a face detection implementation (color model) of an image (176*144) on Altera APEX FPGA that could handle 434 frames/s. Several other studies have conducted on FPGA implementations such as [5] (processing time is 200 ms for an image (180*180) on FPGA Stratix II), [6] (processing speed is 50 frames/s),[7] (face detection using neural network for images (256*256) on Virtex FPGA with an 99.67% occupancy rate).
We remark that the face detection part with various methods is not an easy task. It requires material resources with a varied execution time depending on the method used and the image size. Our objective is implementing an application of localizing and extracting facial-features and components on an FPGA circuit. We plan to exploit the algorithm parallelism and the memory management by reducing the memory space and the execution time. In this work we are interested in two main parts: face detection and interest-region localization (components) (eyes, mouth and forehead). We have chosen and have combined several methods and algorithms that have a reduced complexity and a maximum performance. We have validated these algorithms using MATLAB Tool. Then we realize the description of these algorithms by VHDL hardware language. Having made this choice we are also interested in the optimized implementation of these algorithms on FPGA.

2. FACE DETECTION AND FACIAL COMPONENTS EXTRACTION ALGORITHM

Several methods for face detection and facial-feature extraction exist in the literature. These methods are different in their complexity, performance, type and nature of the images and the application. In this work we use plain color background images whose color is different from the skin and witch contain a single face. The image size (n,m) is (192*256) (figure1).We have chosen a face detection algorithm based on the skin detection (using RGB space) developed in [8,9,10,11] and an algorithm of extracting the facial-features based on tracking the gradient developed in [12] and the geometric model developed in [13].The method consists of two parts. The first part is the face detection. We have chosen a simple algorithm of detection that does not have a lot of calculations and constraints. The algorithm eliminates the regions that do not have the skin color using the RGB space [8,9,10,11](Figure 1.b).This first part comprises the following step: segmentation and binarization, determination of the coordinates of the face, and the face extraction.

A pixel is said a skin pixel, if the components R (Red), G (Green) and B (Blue) of the color satisfy the following conditions (segmentation and binarization) [8,9,10,11]:

$$\begin{cases} \textbf{\textit{If}}\ (R > 95)\ and\ (G > 40)\ and\ (B > 20)\ and \\ \quad ((\mathrm{Max}[R,G,B] - \mathrm{Min}[R,G,B]) > 15)\ and \\ \quad\quad (Abs\ (R - G) > 15)\ and\ (R > G) and \\ \quad\quad (R > B) \qquad\qquad \textbf{\textit{then}} \\ \quad skin\ pixel\ (1) \\ \textbf{\textit{Else}} \\ \quad not\ skin\ pixel\ (0) \end{cases} \tag{1}$$

After selecting the skin pixels, we obtain a binary image whose various-pixel values are equal to 0 or 1.The principle of determining the coordinates of the face is to achieve a horizontal projection (HIb) and a vertical one (VIb) of the binary image.

$$HIb = \sum_{y=1}^{m} Ib(x, y) \qquad VIb = \sum_{x=1}^{n} Ib(x, y) \tag{2}$$

We search the positions of the first two values that are different from the zero of both sides of the projection vectors HIb and VIb. These positions are the coordinates of the face in the original image (L1, C1, L2, C2) (Figure 1.a).

The second part is the extraction of the facial-features. Several extraction methods use the color information [14]. These methods have two limitations. They work only with color images, so illumination changes can affect the results. There are some methods based on the gradient information that are robust to illumination changes [15,16,17]. We have chosen a method developed in [12]. The method is based on the fact that human faces are constructed in the same geometrical configuration. To model the face with a geometric model, the method uses a gradient tracking to locate each component of a frontal face on plain color background. Thus, it presents robustness against small rotation angles, lighting conditions, skin color or accessories (mustache, glasses, beards)[12]. We search through this method the extraction of two eyes, the mouth and the forehead. The method uses a gradient tracking to locate each facial component [12]. By exploiting the property of the face (since it is parameterized), it is easy to retrieve an object characteristic referring to its coordinates in the face with regards to the center of the face and the level of the eyes.

To locate the facial components, we first determine their horizontal axes, and then we determine the area of each component by applying a geometric model [13]. This assumes that the vertical distances between the eyes and the nose and between the eyes and the mouth are proportional to the horizontal distance between the centers of the eyes [13]. Distances related to this model will be described later. The proposed method comprises the following steps:

- Grayscale conversion (Figure 2.a),

The grayscale RGB space transformation algorithm is based on the following formula [14]:

$$I = R \times 0.299 + G \times 0.587 + B \times 0.114 \tag{3}$$

Where I expresses the gray level for a given pixel, and R, G and B are the color components of the pixel

- Calculating the gradient in the horizontal direction (Figure 2.c):

$$\nabla Iy = \frac{\partial I}{\partial y}\vec{j} \tag{4}$$

Where y and j is the index and the horizontal direction, and I is the grayscale image.

- Eye level location (N_{eyes}): realizing of the horizontal projection of the gradient image, and searching for the maximum of the horizontal projection (Figure 2.d).

$$HIy = \sum_{y=1}^{m} \nabla Iy(x, y) \quad N_{eyes} = Max(HIy) \tag{5}$$

- location of the central axis (N_{center}): search for the position of the highest gray level on the eye axis (Figure 2.b):

(6)

$$N_{center} = Max(Iy(N_{eyes}, y))$$

- Location of the nose (N_{nose}): search for the highest gradient in a narrow window by horizontal projection, below the eyes axis and around the central axis with a width of pixels Δx (Figure 2):

$$H_{N_{nose}} Iy = \sum_{y=N_{center}-\Delta y}^{y=N_{center}+\Delta y} Iy(x + N_{eyes} + \Delta x, y) \quad (7)$$

$$N_{nose} = Max(H_{N_{nose}} Iy) \quad (8)$$

- Location of the mouth (N_{mouth}): same principle as the previous step (Figure 2):

$$H_{N_{mouth}} Iy = \sum_{y=N_{center}-\Delta y}^{y=N_{center}+\Delta y} Iy(x + N_{nose} + \Delta x, y) \quad (9)$$

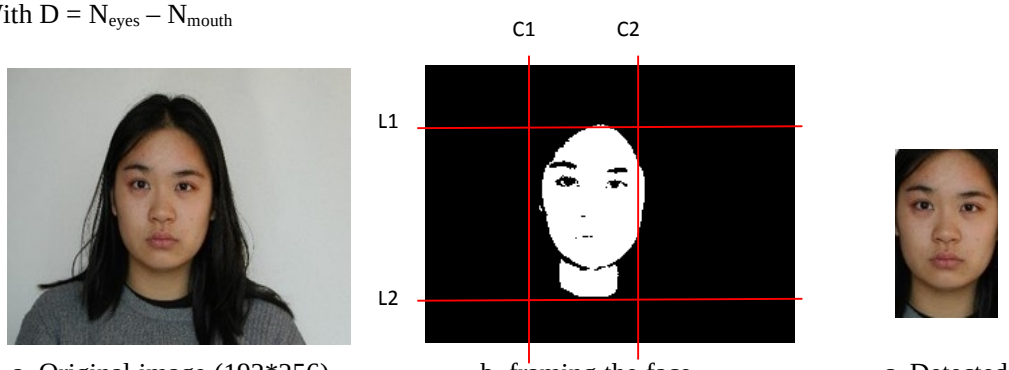
$$N_{mouth} = Max(H_{N_{mouth}} Iy) \quad (10)$$

- Application of the geometric model (Figure 3).

The distance between the two-eye centers is D. The geometric face model (Figure 3) and the related distances are described below [18]:

- The vertical distance between the center of the mouth and the two eyes is D.
- The vertical distance between the center of the nostrils and the two-eyes is 0.6D.
- The mouth width is D.
- The nose width is 0.8D.
- The vertical distance between the eyebrows and the eyes is 0.4D.

With $D = N_{eyes} - N_{mouth}$



a. Original image (192*256) b. framing the face c. Detected face (119*74)

Figure 1: Detection and extraction of the face

Once the eyes axis and the mouth axe are located, we apply the geometric model of the face [13]. Figure 3 shows the results obtained for the extraction of the facial components (eyes, forehead and mouth).

The validation of these algorithms is carried out under MATLAB tool on a computer (CPU: Intel Corp. Duo Frequency: 1.8 GHz RAM: 3 GB). Table 1 shows the execution time of

this method. The execution is done in a sequential manner on the computer processor; it explains the high execution time (Table 1).

In the next section we use the VHDL language and FPGA circuit implementation to reduce execution time of the application and optimize the memory space.

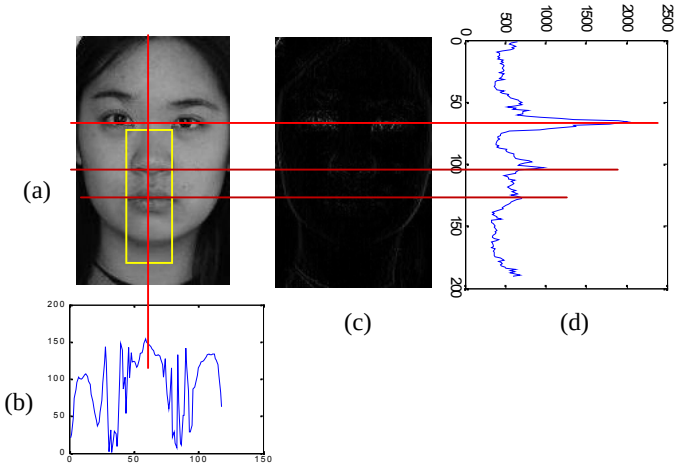


Figure2: Location of the axes of the eyes, the nostril, the mouth and the median

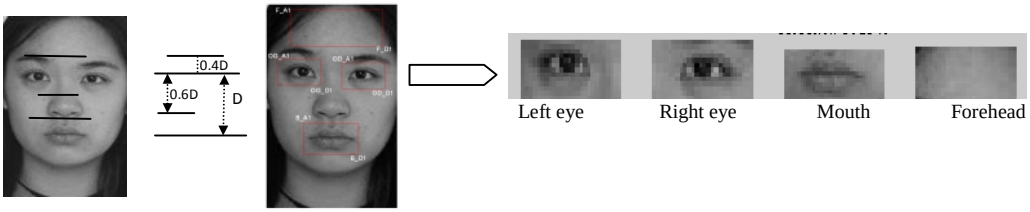


Figure 3: Application of the geometric model and extraction of the eyes, the mouth and the forehead.

Table 1: Execution time of the method

Algorithm	Time (ms)
Face detection	261.8
Facial component extraction	23
Total	284.8 (3.5 images/s)

3. IMPLEMENTATION OF THE ALGORITHM OF FACE DETECTION AND COMPONENTS EXTRACTION

The architecture of the application is composed of two blocks: a block of face detection and a block of components extraction (interest regions) (Figure 4). We explore the possible execution parallelism in algorithms to design the optimal architecture of the application. We designed an architecture for each block. We realized the descriptions of these tow architecture in VHDL, the simulation with ModelSim, the synthesis and implementation with Xilinx ISE tool.

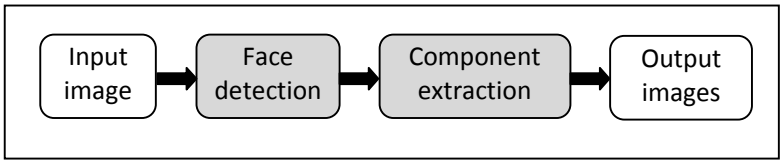


Figure 4: Architecture of application

3.1. Face Detection

The description of this part for an implementation is shown in Figure 5.The architecture comprises 8 blocks: two memory blocks (for the original image and the image gradient), a block of reading the component pixels (R,G,B), a block of segmentation and binarization, a block of search for line L1 and L2 using horizontal projection, a block of searching for columns C1 and C2 using vertical projection, a block for face extraction and a block of grayscale to RGB conversion.

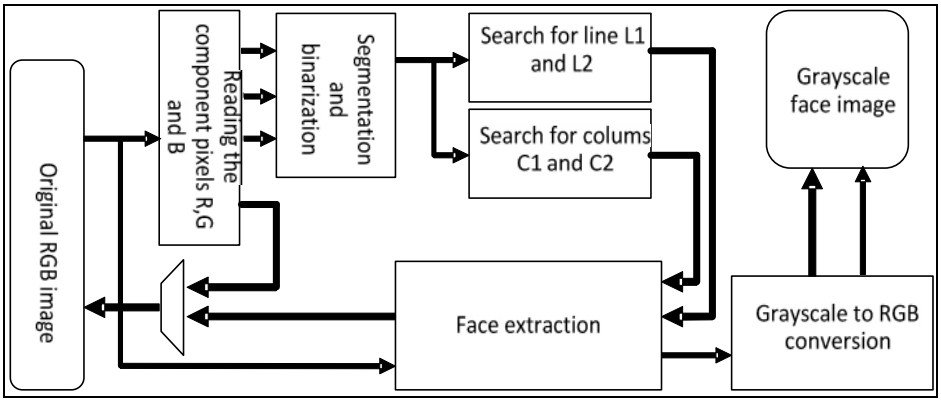


Figure 5: Block diagram of the hardware architecture of the face detection part using the RGB component

Organization and size of the memory space for the face detection portion architecture is presented in figure6.

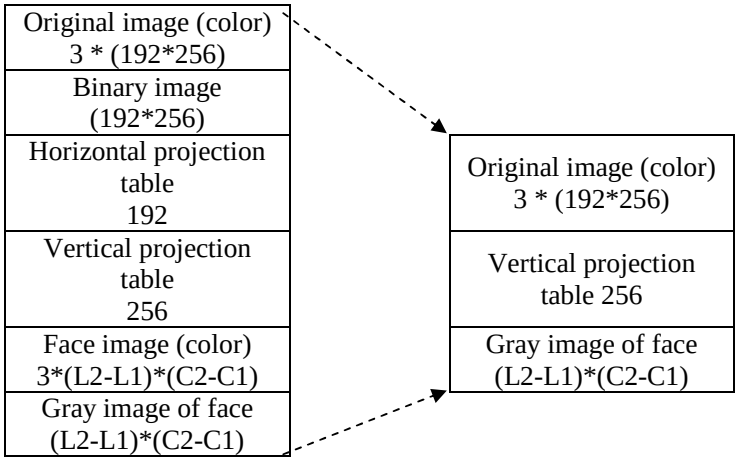


Figure 6: Size of the memory space before and after optimization

In this hardware description we have optimized the memory space and we have exploited the parallelism of some functions in order to decrease the execution time.

3.1.1. Parallel execution of bloc

We proceed to perform in parallel the segmentation of the skin pixel algorithm, the research-line (L1 and L2) algorithm and the search-column (C1 and C2) algorithm. The face-color extraction (RGB) algorithm, the grayscale conversion algorithm and the in memory recording are performed in parallel.

```
for i = 1 … n do
        for j = 1 … m do
            if ((R(i,j) > 95) AND (G(i,j) > 40) AND (B(i,j) > 20)) AND
             (Max(R(i,j),G(i,j),B(i,j))-Min(R(i,j),G(i,j),B(i,j)))>15) AND
             ( abs(R(i,j) - G(i,j)) > 15) AND (R(i,j) > G(i,j)) AND (R(i,j) >
             B(i,j))then
                 S(i,j) = 1;
             else
                 S(i,j) = 0;
           End if
        Enf for
End for
```

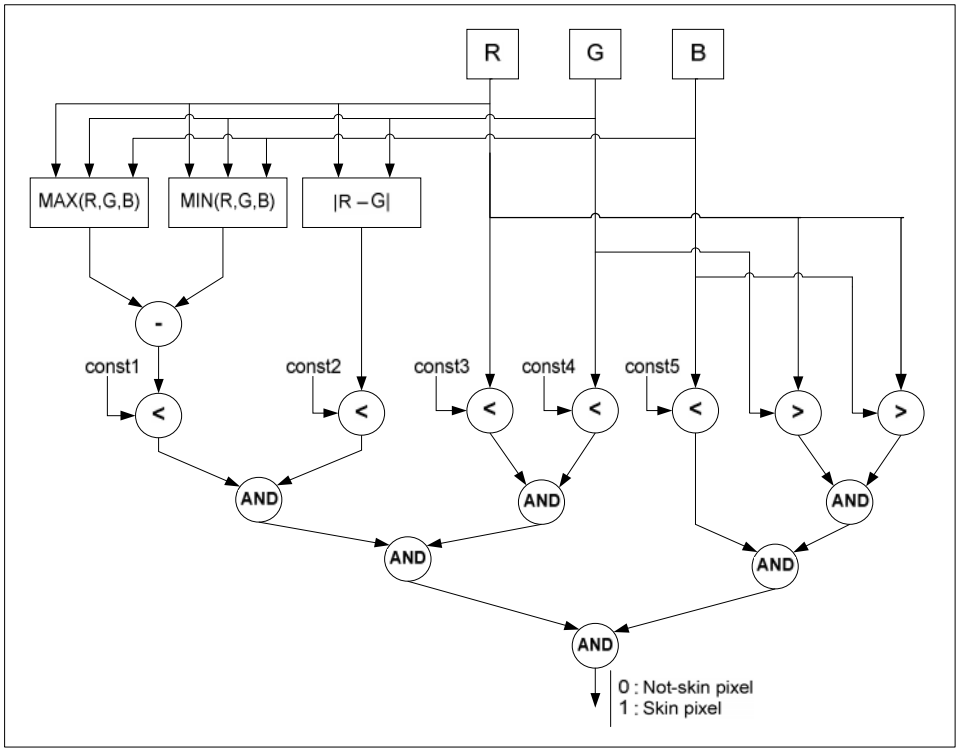


Figure 7: Architecture of the hardware description for selecting and segmenting regions of the skin algorithm

Figure 7 represents the hardware description of the algorithm for selecting and segmenting the skin regions for an FPGA implementation. In this description we read the three components of the pixel in parallel. We reserve 3 memories (each of size (192*256 bytes)) for the three matrices of the original image. We realize the execution of the absolute value of (RG), the search for the maximum of (R, G, B) and the minimum of (R, G, B) in parallel. Different comparisons are performed in parallel.

3.1.2. Memory Optimization (figure 6)

Depending on the method used, we have 4 images: original color image (3 matrices (192 * 256), binary image (one matrix (192 * 256)), RGB color image extracted from face (3 matrices (L2-L1)*(C2-C1)) and gray level image (1 matrix (L2-L1)*(C2-C1)) : there is also an array of size 192 bytes for the horizontal projection and another of size 256 bytes for the vertical projection. The all the memory space is:

$$\text{Memory space} = 4 * (192 * 256) + 4 * (L2-L1) * (L2-L1) + 448 = 232280 \text{ bytes}$$

Since the binary image is used only once, and since the determination of the coordinates L1, C1, L2, C2 is in parallel, we do not need to save the binary image in memory. The horizontal projection allows defining the two lines L1 and L2; therefore we have used a method to determine these two indices without creating a memory table. Using the coordinates we extract the facial image from the original image and for each read pixel, we execute the grayscale conversion. The new memory space required for this party becomes:

$$\text{Memory space} = 3 * (192 * 256) + 1 * (L2 - L1) * (C2 - C1) + 256 \text{ bytes} = 156518 \text{ bytes}$$

3.2. Facial components extraction

The hardware description of this part for an implementation is shown in Figure 8. In this description we have optimized the memory space by exploiting the parallelism of functions in order to minimize the execution time. The architecture comprises 10 blocks: five memory blocks (for gradient image, image of left eye, image of right eye, mouth image and forehead image), a block for gradient calculation, a block for search for the eye level, a block for search for the center, a block for search the nose and mouth level, a block for application of the geometric model. The block of grayscale RGB pixel conversion and the memory block of grayscale image are belonging to the first part. They are present only to show the complete data path in the second part.

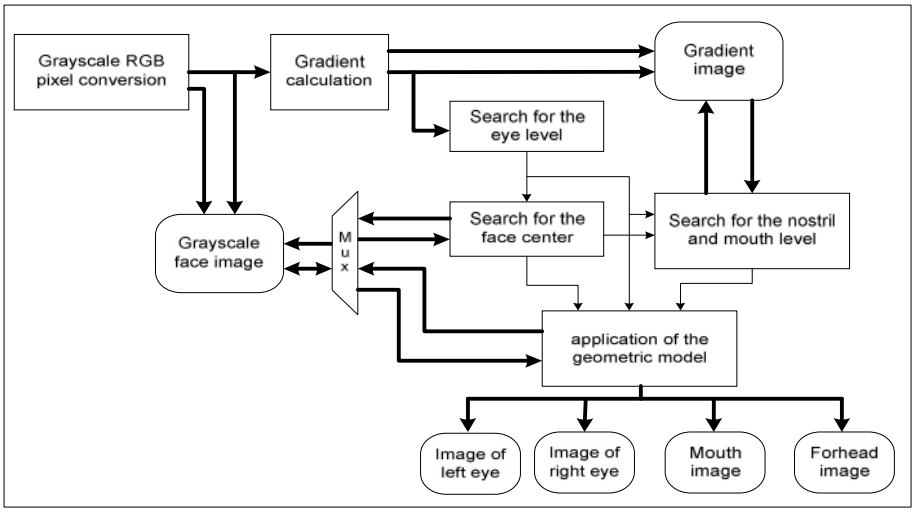
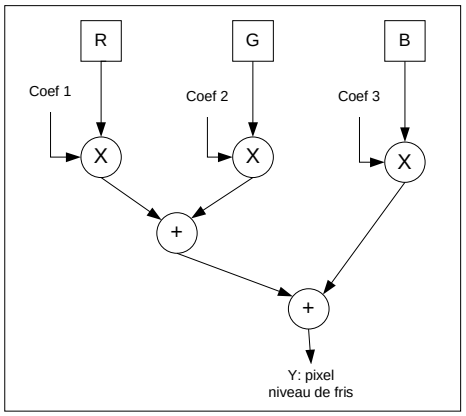


Figure 8: Block diagram of the components extraction hardware architecture

3.2.2. Parallel bloc execution

We have chosen that the memorization of grayscale calculated in the memory, the calculation of the gradient and its storage in the memory and the eyes level localization are to be executed in parallel.

Figure 9 shows the description of the grayscale RGB conversion function. Using the indices of the lines (L1, L2) and the columns (C1, C2) of the face rectangle, we convert them in to addresses and we read the components R, G and B of the three memories at the same time. We execute the three multiplications in parallel (figure 9).



With coef1 = 0.299, coef2=0.587 and coef3= 0.114

Figure 9: Hardware description of the algorithm for grayscale RGB conversion of the face image

Figure 10 shows the description of the gradient calculation algorithm of the grayscale image. The gradient of a pixel is the difference between the pixel (i,j+1) and the pixel (i,j). We have calculated the gradient for each pixel converted in to a grayscale. We perform the gradient calculation in parallel with the grayscale conversion. We apply a delay of the j pixel and then we calculate the difference in P(i,j +1) and P(i,j) (with P is the pixel value).

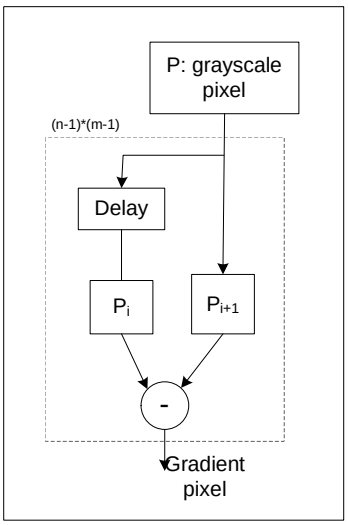


Figure 10: Description of the gradient calculation algorithm

3.2.2. Memory Optimization (Table 11):

In this section we have many memories: memory of gradient image ((L2-L1)*(C2-C1)) bytes, memory for the horizontal projection of size (L2-L1) byte of the gradient image for localizing the axis of the eyes, memory for the horizontal projection of size ((L2-L1)-N_{eyes}) bytes of a part of the gradient image for localizing the nose and the mouth, memory for the left eye image with (10*28) bytes, memory for the right eye image with (10*28) bytes, memory for the mouth image with (16*40) bytes, and memory for the front image with (20*68) bytes.

Once we have determined the eyes level, we determine the face center. We search for the face center on the line that corresponds to the eye level in the grayscale image. Using the luminance component face, the light intensity at this level is clearer (value of highest intensity) which matches the center of the face. We need just to find the maximum intensity of the line. We use the eye level and the face center to locate the levels of the nostril and the mouth. Then we take an area of the gradient image. This area is centered on the median axis of the face. This space has a width of$\pm\Delta y$ of the center of the face and the length of (((L2-L1)-N_{eyes})+Δy), with $\Delta y=20$.This area includes the nose and the mouth. The choice of this area reduces the effect of the beard, mustache and other structures in the face. The horizontal projection of the area has two peaks. The first greater peak is for the nose level and the second one is for the mouth. We need just to find these two peaks. Organization and size of the memory space for the component extraction part are presented in figure 11.

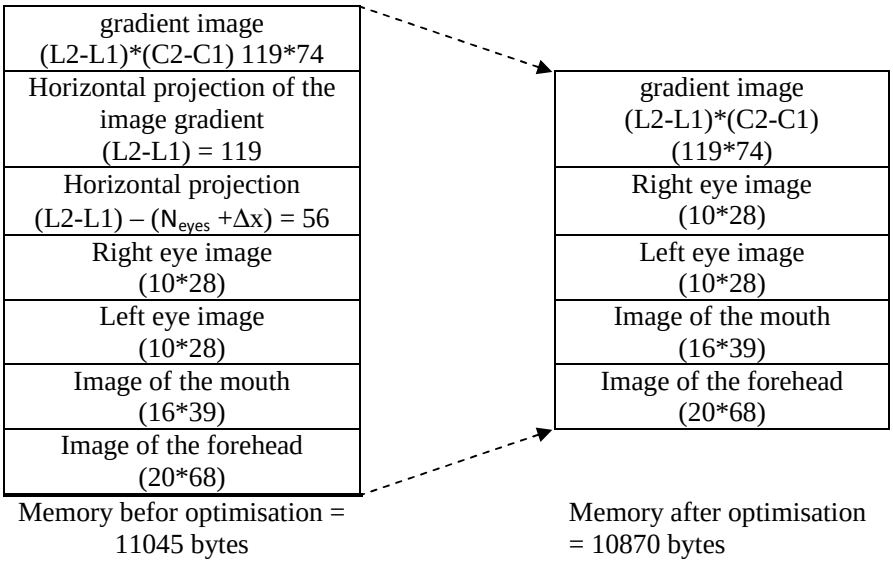


Figure 11: Size of the memory space before and after optimization

3.2.3. Application of the geometric model

Once we have determined the eyes level indices, the nostril level, the mouth level and the center of the face, we can determine the coordinates of the eyes, the mouth and the forehead while applying the geometric model. This assumes that the vertical distances between the eyes and the nose and between the eyes and mouth are proportional to the horizontal distance between the centers of the eyes D. the distances related to this model are described in [13].

3.3. Results of the simulation, synthesis and implementation

To implement the algorithms already presented, a VHDL description is simulated and synthesized to obtain an adequate description for the desired operation of the design system. We have used an FPGA family Xilinx (Virtex 5 XC5VLX30). Material resources necessary for the implementation of this application are given in Table 2. The results presented in Table 3 and 4 are given by the simulation, synthesis and implementation on Xilinx ISE tool, and we have chosen as target FPGA Virtex 5 XC5VLX30. We have not done a real physical implementation on FPGA, just a simulation and emulation on the Xilinx ISE tool.

Table 2: Resource FPGA circuit for the implementation

FPGA Circuit	Slice	BlocRAM	DSP
Virtex 5	1799	20	6
XC5VLX30	37.4%	31.2%	18.75%

Table 3 shows the memory space optimization in the application

Table 3: memory space optimization for the implementation

Algorithm	Memory before optimization	Memory after optimization	gain optimization
Face detection and grayscale conversion	232280 bytes	156518 bytes	32.61%
Components extraction	11045 bytes	10870 bytes	1.58%
Total	243325 bytes	167388 bytes	31.2%

Table 4: shows the execution time of the application. This time allows us to treat 490 frames/s with a clock frequency of 50MHz.

Table 4: Execution time of application (simulation result with a frequency of 50MHz)

Algorithm	Execution time (en ms)
Face detection and grayscale conversion	1.1646
Component extraction	0.8734
Total	2.038 (490 images/s)

Figure 12 shows the simulation results of the VHDL code architecture (left eye image, right eye image, mouth image and forehead image) obtained by the developed architecture.

Figure 12: Images extraction Results of developed hardware architecture

IV. CONCLUSION

We have chosen and associated two algorithms that have reduced complexity with a maximum performance. The face detection algorithm is based on the use of the color information. The algorithm of localizing the facial components is based on the gradient which provides robustness against illumination changes. The architectural description of the application developed in this work allows us to process 490 frames/s with a frequency of 50MHz. This allows us to consider the implementation of other algorithmic parts of analyzing the facial expressions. In addition, we have performed an optimization of the memory space with a gain of 31.2%. This optimization is obtained by exploiting the implementation parallelism offered by the FPGA circuit and the execution parallelism that we have achieved in the algorithms of this application.

REFERENCES

[1]Junguk Cho, ShahnamMirzaei, Jason Oberg, Ryan Kastner, (2009) "Fpga-based face detection system using Haar classifiers" Proceeding of the ACM/SIGDA international symposium on Field programmable gate arrays, California, USA, Pages 103-112.
[2]Nicolas Farrugia, Franck Mamalet, S´ebastien Roux, Michel Paindavoine, Fan Yang, (2007) "F Implantation parallèle de la detection de visages : Méthodologie et implantation sur FPGA". Colloque GRETSI, pp 573-576, Troyes, 11-14 septembre 2007.
[3]Mohammad S. Sadri, Nasim Shams, MasihRahmaty, IrajHosseini, ReihaneChangiz, ShahedMortazavian, ShimaKheradmand, RoozbehJafari, (2004) "An FPGA Based Fast Face Detector", GSP 2004 pp. 1039, 2004.
[4]Paschalakis, S., Bober, M., (2003) "A Low Cost FPGA System for High Speed Face Detection and Tracking", In Proc. 2003 2nd IEEE International Conference on Field Programmable Technology (FPT '03), Tokyo, Japan, Dec. 15-17, pp. 214-221, 2003.
[5]L. PIERREFEU, J. JAY., (2007) "Détection et localisation de visage pour un système grand public d'authentification de visage implantable sur FPGA". Colloque GRETSI, Troyes, pp 361-364, 11-14 septembre 2007.
[6]T.Theocharides, G.Link, N.Vijaykrishnan, M.J.Irwin, and W.Wolf, (2004). "Embedded hardware face detection" Proceedings of the 17th International Conference on VLSI Design (VLSID 04), 2004.
[7]Smach, .M Atri, J. Mitéran and M. Abid, (2006) "Design of a Neural Networks Classifier for Face Detection" Journal of Computer Science 2 (3): 257-260, 2006.
[8]Garcia C. ,Tziritas G., (1999) "Face Detection Using Quantized Skin Color Regions Merging and Wavelet Packet Analysis" IEEE Transactions on Multimedia, 1(3), p.264-277, September 1999.
[9]BencherietChemesse-Ennehar, BouallegAbd El halim, Tebbikh Hicham, (2007) "Segmentation de la Couleur de Peau par Seuillage selon Différents Espaces de Couleur" JIG'2007 - 3èmes Journées Internationales sur l'Informatique Graphique, pp 207-211, INRIA Sophia-Antipolis, France, 2007.
[10]F. Gasparini and R. Schettini, (2006) "Skin segmentation using multiple thresholding in Internet Imaging" vol. 6061 of Proceedings of SPIE, pp. 1–8, San Jose, Calif, USA, January 2006.
[11]J. Kovac, P. Peer, and F. Solina, (2003) "2D versus 3D colour space face detection" in Proceedings of the 4th EURASIP Conference on Video/Image Processing and Multimedia Communications, vol. 2, pp. 449–454, Zagreb, Croatia, July 2003.
[12]F.ABDAT, C.MAAOUI et A.PRUSKI, (2007) "Suivi du gradient pour la localisation des caractéristiques faciales dans des images statiques" Colloque GRETSI, Troyes 11-14 septembre 2007.
[13]Frank Y. Shih, Chao-Fa Chuang, (2004) "Automatic extraction of head and face boundaries and facial features" Information Sciences 158 pp. 117–130, 2004.
[14]Michael J. JONES and James M. REHG. (1999) "Statistical Color Models with Application to Skin Detection" In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, volume 1, page 1274, Fort Collins, Colorado, June 1999.
[15]Deng X., Chang C. H. and Brandle E. (2004) "A new method for eye extraction from facial image". Proc. 2nd IEEE international workshop on electronic design test and applications (DELTA), 2 NO.4:29–34, Perth, Australia, 2004.
[16]Maio D. and Maltoni D. (2000) "Real-time face location on grayscale static images" Pattern Recognition, 33:1525– 1539, 2000.
[17]Tsekeridou S. and Pitas I. (1998) "Facial feature extraction in frontal views using biometric analogies" Proc. 9th European Signal Processing Conference, 1:315–318, September 8-11 Island of Rhodes, Greece, 1998.
[18]A. Nikolaidis, I. Pitas, (2000) "Facial feature extraction and pose determination" Pattern Recognition 33 (2000) 1783–1791.

Authors

Nadia NASR received the Bachelor's and Master's Degree in the Faculty of Science at Monastir of Manastir University, in 2007 and 2009 respectively. Since 2010, she has been working as a Research Scientist at the Research team of Medical Imaging Technology (TIM) at Faculty of Medicine at Monastir of Monastir University where she prepares his thesis. She has also two published paper in international conference in the same areas. His areas of interest include dedicated hardware architecture of image processing.

Bouraoui MAHMOUD received the Bachelor's, Master's Degree and doctorate in the Faculty of Science at Monastir of Monastir University, in 1997, 1999 and 2006 respectively. Since 2008, He is currently Assistant Professor in the Department of Industrial Electronics in the National Engineering School of Sousse, University of Sousse. He has six published papers in international journals. His areas of interest include embedded processor, embedded system, image processing Hardware architecture, codesign HW/SW.

Mohamed Hedi BEDOUI Since 2011, He is currently Professor in Biophysics in the Faculty of Medecine at Monastir of Monastir University. He has many published papers in international journals. His areas of interest include Biophysics, medical imaging processing, and embedded system, codesign HW/SW.