High Performance Colored Image Segmentation System Based on Neural Network

Shefa A. Dawwd

Computer Engg. Dept./College of Engg. /Univ .of Mosul,
shefadawwd@yahoo.com

Abstract

image segmentation is often the most time-consuming part of image processing systems. Traditionally, systems employing real-time color-based segmentation are either implemented in hardware, or in very specific software systems. This paper describes an FPGA implementation of a skin color segmentation based on a neural network. The proposed segmentation approach is an essential stage for face detection. The system uses a multilayer feedforward neural architecture with three-inputs, one hidden layer, two output neurons and a pipelined saturating linear activation function to simplify the FPGA hardware implementation. The system was tested by using different colored face images for face segmentation problem and its performance was compared with the results obtained using advanced software system designed specifically for face segmentation. A comparable performance was achieved and a speed up of (64583) was estimated compared to a Pentium 4, 2.4 GHz general purpose sequential computer and when it is compared to reduced instruction set computer IBM *RISC* 350 station, it was (407).

Keywords: Neural network implementation, image Segmentation, FPGA based systems

منظومة عالية الاداء مبنية على الشبكات العصبية لتقطيع صورة ملونة

شفاء عبدالرحمن داؤد

جامعة الموصل – كلية الهندسة – قسم هندسة الحاسبات
shefadawwd@yahoo.com

الخلاصة

تعتبر خوارزميات تقطيع الصور من الخوارزميات الاكثر استهلاكا للزمن ضمن خوارزميات معالجة الصور. عادة ما يتم تنفيذ منظومات الزمن الحقيقي لتقطيع الصور الملونة بأستخدام كيان مادي او بأستخدام كيانات برمجية عالية التخصيص. في هذا البحث تم تنفيذ نظام مادي مبني على استخدام الشبكات العصبية لتقطيع الاجزاء المحتواة على لون الجلد بأستخدام مصفوفة البوابات المبرمجة حقليا. تعتبر عملية التقطيع المقترحة مرحلة اساسية تستخدمها منظومات كشف الوجوه. تستخدم المنظومة المقترحة في هذا البحث معمارية عصبية متعددة الطبقات ذات ثلاثة ادخالات وطبقة مخفية واحدة وخليتين عصبيتين في طبقة الاخراج مع دالة تفعيل خطية مشبعة ذات تقنية انبوبية وذلك لتسهيل التنفيذ المادي المطبق على مصفوفة البوابات المبرمجة حقليا. تم فحص النظام اعتمادا على صور ملونة متعددة تحتوي على عدة صور وجوه وذلك لتقطيع الوجوه الموجودة في هذه الصور. تم مقارنة انجاز المنظومة من خلال النتائج المستحصلة باسنخدام كيان برمجي متخصص في هذا المجال منفذ على حاسبة تسلسلية ذات استخدام عام نوع Pentium 4, 2.4 GHz والحصول على تسارع مقداره 64583 وعند مقارنة النظام مع حاسبة الايعازات المختزلة ذات الاستخدامات الخاصة نوع IBM *RISC* 350 تم الحصول على تسارع مقداره 407.

Received 17 Jan. 2007 **Accepted 3 May 2008**

1 INTRODUCTION

Image segmentation, which aims to divide a given image into homogeneous and meaningful regions, is the crucial step in image processing since it directly affects the performance of subsequent operations such as image analysis, measurements, detection and classification. Despite the multitude of image segmentation methods proposed in the last three decades[1]., the quest for new more effective methods continues. This is partly due to the necessity to handle as broad a category of images as possible, partly to meet the real-time demands in practical applications.

Artificial neural networks(ANNs) have been widly used in image segmentation. One of the early applications of the use of ANNs in medical image segmentation was by Ozkan et al., who used the backpropagation learning to segment medical images[2]. Uchiyama and Arbib used competitive learning(CL) to cluster colors in image[3]. Littmann and Ritter developed an ANN, named local linear maps(LLM), for adaptive color segmentation and compared it with statistical methods[4]. It has shown that ANNs are well-suited for using in segmentation problems than conventional methods.

But in the field of computational speed, using of sequential computers to implement neural based segmentations seems to be practically unsuitable in real time image processing applications such as mobile robot applications, or other domains where interaction with humans or dynamic world is required such as videophone. To speed up calculations in ANNs which are softwarely implemented, computers with high performances are required. Littmann and Ritter[4] used an IBM *RISC* 350 station for implementation of LLM network. Hardware implementation of ANNs speeds up the computational performance and in consequently speeds up the required bit rates for such real time applications. Boussaid et al., described an analog VLSI implementation of a skin detector based on multilayer feedforward neural network[5]. Fiesler developed a high speed image segmentation system based on single, onchip CMOS analog processor[6].

We address in this paper a digital techniques coupled with parallel processing potential of neural networks for segmentation of skin in colored images. A hybrid adaptive system which combined the advantages of knowledge based and neural methods is proposed. This system is a special-purpose object detector that can segment arbitrary objects in real images with a complex distribution in the feature space in real time. This is achieved after training with one or several previously labeled image(s). The proposed adaptive segmentation system uses local color information to estimate the membership probability in the object, respectively, background class. The classification methodology is more similar to that used in [4] and [5], while the implementation technique differs. The system can be applied to detect and to localize the human face in colored images in real time. The system is hardwarely implemented using FPGA techniques (see Fig. 1)..

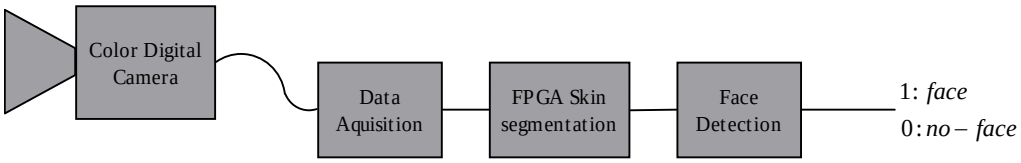


Fig. 1: The system of Face detection

2 COLOR REPRESENTATIONS

So far, the RGB system is the dominating representation of color due to its importance for television and camera systems. Physiologically, this representation is similarly realized in the human retina that consists of three color-sensitive photoreceptor types with a maximum spectral sensitivity corresponding to red, green, and blue. Besides the RGB system there is a variety of other representation that can be constructed by linear or nonlinear transformations from the RGB values[4]. One major disadvantage of the RGB system is the dependency of all three parameters from the light intensity. This disadvantage is avoided in other representation schemes where color and intensity are represented independently. One such system is the Yuv color space. The color plane forms an equilateral triangle that can be constructed in the RGB cube by connecting its R,G, and B corner. The intensity Y is perpendicular to the image plane. The value *u* and *v* are the Cartesian coordinated of the color value. The value for Y , *u*, and *v* are calculated as $Y=(R+G+B)/3$, $u=3(R-B)/2Y$, and $v=\sqrt{3}(2G-R-B)/2Y$.

The formulation of this system in polar coordinates the HIS system, distinguishes *hue*, specified by the angle, *saturation* as radial component, and *intensity* again vertical to the uv plane. We prefer the Yuv system because of its continuous representation of the color components. A similar effect of intensity independence can be achieved by simply calculating *color ratios* $Q_{RG}=R/(R+G)$ and $Q_{RB}=R/(R+B)$. This color representation is referred to as YQQ space.

3 SKIN DETECTION USING NEURAL NETWORKS

Neural networks have the ability to learn complex data structures from a set of example patterns. They have the advantage of working fast (after the training phase) even with large amount of data. The results presented in this paper are based on a multilayer feedforward network architecture, known as the multilayer perceptron(MLP). The MLP is a powerful tool that has been used extensively for classification, nonlinear regression, speech recognition, hand-written character recognition and many other applications [7]. The elementary processing unit in a MLP is called a neuron or perceptron. It consists of a set of input synapses, through which the input signals are received, a summing unit and a nonlinear activation transfer function. Each neuron performs a nonlinear transformation of its input vector; the input-output relationship is given by:

$$\varphi(X) = f\left(\sum_{j=1}^{P} w_j x_j + \theta\right) = f(W^T X + \theta) \qquad (1)$$

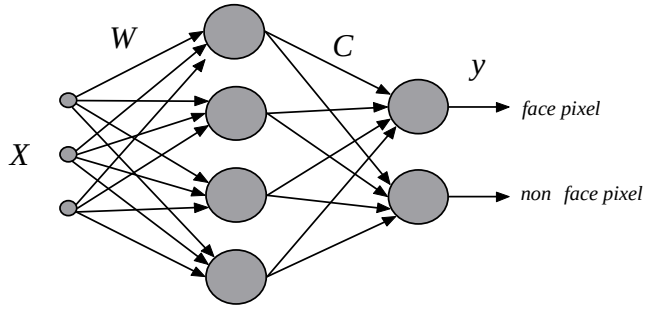
where *W* is the synaptic weight vector, *X* is the input vector, θ is a constant called the bias, *f* is the activation function, superscript *T* is the transpose operator, and $\varphi(X)$. is the neuron output signal.

An MLP architecture consists of a layer of input units, followed by one or more layers of processing units, called hidden layers, and one output layer. Information propagates, in a feedforward manner, from the input to the output layer; the output signals represent the desired information. The input layer serves only as a relay of information and no information processing occurs at this layer. Before a network can operate to perform the desired task, it must be trained. The training process changes the parameters of the network in such a way that the error between the network outputs and the target values (desired outputs) is minimized.

In this paper, we propose a method to detect skin color. The skin detector uses an MLP with three inputs, one hidden layer and two output neurons (see Fig. 2). Each pixel is represented by either RGB (red, green and blue) or Yuv color components. These three color components are used as inputs by the neural network. The output of each hidden neuron is given by (1), and the network output is given by:

$$y = \sum_{j=1}^{Q} c_j \varphi_j(X) + \beta \qquad (2)$$

where $\varphi_j(X)$ is the output of the j-th hidden neuron, and c_j is the synaptic weight of the output neuron.



where X = {R, G, B} or {Y, u, v}

Fig. 2: Neural Network structure for pixel face detection

To estimate the neural network parameters (i.e. synaptic weights and biases), a training set containing 8800 skin and non-skin pixels was extracted from set of images. The network was trained using backpropagation algorithm [8]. The generalization ability of the trained network is tested using a set containing several thousands of skin and nonskin pixels. The training and test sets were extracted from images containing skin colors of people from different races and under different lighting conditions.

4 ANN IMPLEMENTATION PLATFORMS

The implementation of applications such these involve different processes like image processing and neural networks on a general purpose computer can be easier but not very efficient in terms of speed. The reason being the additional constraints put on memory and other peripheral device management.

An important aspect of computer architecture is the design of the instruction set for the processor. The instruction set chosen for a particular computer determines the way that machine language programs are constructed. Early computers had small and simple instruction sets, forced mainly by the need to minimize the hardware used to implement them. As digital hardware became cheaper with the advent of integrated circuits, computer instructions tended to increase both in number and complexity. Many computers have instruction that include more than 100 and sometimes even more than 200 instructions. These computers also employ a variety of data types and a large number of addressing modes. The trend into computer hardware complexity was influenced by various factor, such as upgrading existing models to translation from high- level language into machine language programs, and striving to develop machines that move functions from software implementation into hardware implementation. A computer with a large number of instructions is classified as a complex instruction set computer, abbreviated *CISC*.

A number of computer designers recommended that computers use fewer instructions with simple constructs so they can be executed much faster within the CPU without having to use memory as often. This type of computer is classified as reduced instruction set computer or *RISC*. The major characteristics of *RISC* processor are:1) Relatively few instructions 2) Relatively few addressing modes 3) Memory access limited to load and store instructions 4) All operations done within the registers of the CPU 5) Fixed-length, easily decoder instruction format 6) Single-cycle instruction execution 7) Hardwired rather than microprogrammed control.

System implementation on *CISC* or *RISC* is considered as a software implementation. Application specific hardware offers much greater speed than a software implementation. Most artificial neural network models have been implemented in software, but the size and complexity of many problems has quickly exceeded the power of conventional computer hardware.

It is the goal of neural network engineers to transfer the progress made into new hardware systems. These are intended to accelerate future developments of algorithms and architectures, and to make possible the use of dedicated neural networks in industrial applications.

At present there are three main avenues of research to implement the ANN in hardware, each with its own merits and associated problems, namely the digital VLSI, analogue VLSI and optical approaches. This section does not consider optical implementations which are altogether different from the VLSI approach. The employed technology defines the cost, performance and reconfiguration of ANN:i) analog implementation has good performance and low cost , but it is difficult to implement due to required references voltages ii)digital implementation (ASIC: Application Specific Integrated Circuit) has higher cost and poor performance compared with analog one, but it is faster to implement. Both approaches (analog and digital-ASIC) have fixed topology, resulting in an ANN suited only for one type of target application. Modern reconfigurable (FPGA) devices can be used to implement neural networks in hardware[9,10]. Furthermore, digital implementation, using Field Programmable Gate Arrays (FPGAs), allows the redefinition of the topology using the same hardware. FPGAs have traditionally been configured by hardware engineers using a Hardware Description Languages (HDL). The disadvantage of an implementation using FPGA over ASIC is the performance. FPGA normally run slower than ASICs.

4.1 Learning algorithm implementation

The learning algorithms used for modifying weights values using inputs and training data are as an important part of the network system as the architecture itself. Implementation of learning in VLSI systems takes three forms; *off-chip*, *'chip-in-the-loop'* and *on-chip* learning. In *off-chip* learning, weights values are calculated externally by software and are then downloaded to the neural network which is then used only for recall. This is the easiest but least favored method, since training times can be long. Off-chip learning does have the advantage in that it is easy to change the learning algorithm simply by modification of software. It also allows the use of floating point arithmetic for the algorithms which may not be feasible on a neural network chip. *'Chip-in-the-loop'* training chip may also be considered as an off-chip method since the training algorithm is still run in software. However, in this case the neural network is used in the training loop which removes the need for a software model of the network itself, and compensates for device variability. The main drawback of this method is the communications overhead in continually reading and writing data across

the network/host interface. *On-chip* learning must be seen as the most desirable method, since it may open the way to stand-alone neural network chips. The main advantage of running the learning algorithm in hardware is the gain in speed.

4.2 Numerical Representation

In general, neural networks have low-precision requirements, even though the exact specification is algorithm and application dependent. Digital neurohardware can profit from this property by using fixed-point arithmetic with reduced precision. Fixed-point implementations are less complex and less area consuming than floating-point arithmetic and therefore their use helps to reduce system cost[11,12].

4.3 Mapping Neural Networks on Parallel Computers

How can we map a specific neural network on a parallel computer to achieve the a maximum performance? [13]. The key concepts of an efficient mapping are load balancing, minimizing inter-PE communication and minimizing synchronization between the PEs. Furthermore, the mapping should be scalable both for different network sizes, and for different number of processing elements. In Fig. 3, the weight matrix presentation of a simple neural network (four neurons with four synapses each) is shown in the middle, while the left side shows the conventional presentation of the same network. The rectangle N in the mid part of Fig. 3 denotes the activation function of the neuron.

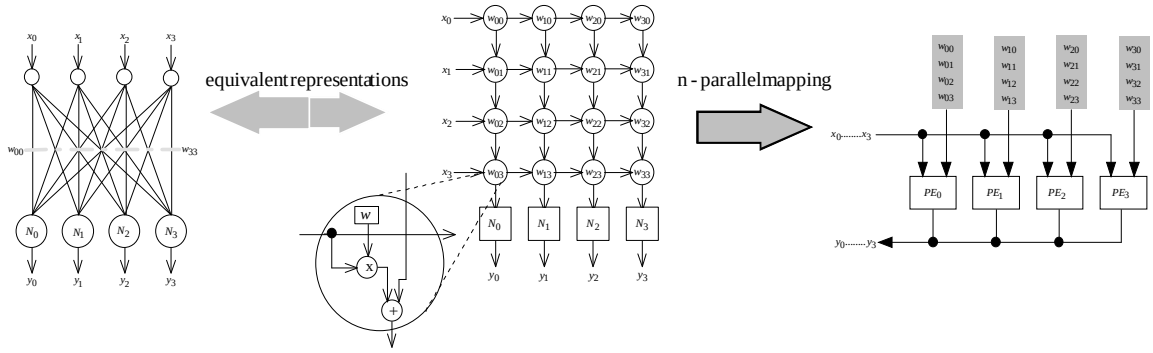


Figure 3. Presentation of a neural network: conventional (left), weight matrix (middle), mapped neuron-parallel (right)

The circle wij represents the computation of the synapse: yi = wij*xj +yi-1 where yi-1 is the result from the proceeding synapse.

4.4 Implementation of ANNs Activation Function

Direct implementation for non-linear sigmoid activation functions is very expensive. There are two practical approaches to approximate sigmoid functions with simple FPGA designs[14]. **Piece-wise linear approximation** describes a combination of lines in the form of $y=ax + b$ which is used to approximate the sigmoid function. Note that if the coefficients for the lines are chosen to be powers of two, the sigmoid functions can be realized by a series of shift and add operations. Many implementations of neuron activation functions use such piece-wise linear approximations one of them is. The second method is **lookup tables**, in which uniform samples taken from the center of sigmoid function can be stored in a table for look up. The regions outside the center of the sigmoid function are still approximated in a piece-wise linear fashion.

4.5 Implementation of the Pixel Skin Detection

The design and all of the work are geared towards the implementation of the MLP neural network for skin detection in a modular fashion. The modular design means that the network can become as large as practically possible thus providing a structure for more complex application. In this implementation, the MLP are trained *off-chip* and the convergent parameters then fed to the hardware for test and synthesis. The processing elements and the complete architecture of the proposed neural network is shown in Fig. 4(a,b). The processing element does the algebraic equations of the electric model of the neuron, that is, the multiplication and sum required in the neuron's internal processing(see Fig. 4a).

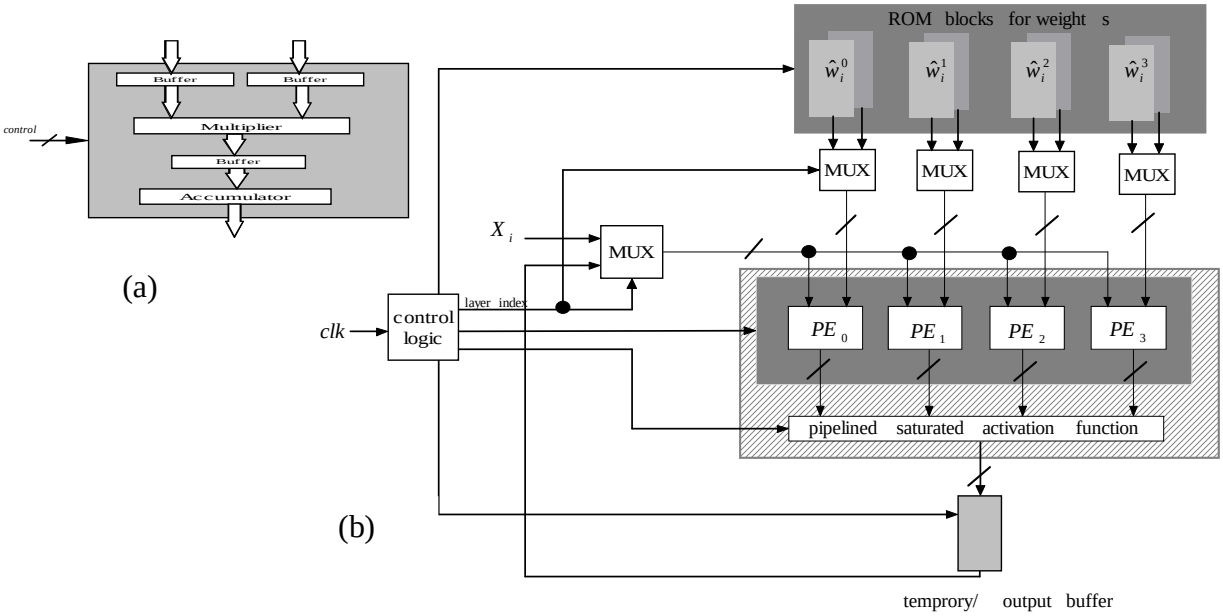


Figure 4: (a) A block diagram of the processing element (*PE*) (b) The architecture layers implementation cycle of the Neural Network

Buffers are used to make the processing element works in a pipelined fashion to speed calculations of input and weight vectores.

Because the number of processing circuits integrated in a FPGA chip is restricted and because that it is not possible to make use of simultaneous computation since the a given layer needs the result of the predecessor in order to operate, all layers use the same resources to build a compact resources shared system, therefore, all layers are executed on the same processing element units by using multiplexers and temporary buffers. The 1st layer uses all the four PEs exist, while the 2nd layer uses only two of them, leaves the two other PEs unused. Weights for each layer can be stored in the FPGA RAM Blocks. Layers implementation cycle is shown in Fig. 4(b). The n-parallel mapping shown in Fig.3 is used to map the MLP in parallel architecture.

The sigmoid function has been substituted by a piecewise linear function. The substitution is based on the selection of a linear equation that has a minimum least square error with the original sigmoid function. This is achieved in software, where the sigmoid equation that is used during the learning phase is approximated with a linear function that best fit it, then the last function parameters are adapted. Using this approximation leads to reduce the at most bottleneck problem in FPGAs (the multiplication). Each processing element output feeds the input of the activation function. But using two multipliers for each feedforward node may be seemed a critical problem if the number of layer nodes exceeds the number of embedded multipliers in the FPGA chip. To solve this problem, only one activation

function unit is built and made common to all feedforward nodes, such that the output nodes use this unit sequentially in a pipelined manner. All the mentioned operations are controlled by one control unit (see Fig. 4(b)). These operations are based on the fixed point representation that is described in section 4.2. The minimum word length for each weight and input elements is the one that should preserve the same generalization ability that achieved from software. It is selected to be 10-bits.

5 EXPERIMENTAL RESULTS

The color segmentation system mentioned in previous sections have been applied to detect and to localize the human face in colored images in real time. In this paper a FPGA VLSI architecture that implements the segmentation network is proposed. Xilinx Spartan-3 FPGA of 200,000 gates is used for implementation. The FPGA digital hardware model has been designed using Xilinx Foundation environment.

Table 1 presents the utilization summary of the Spartan-3 XC3S200, 50MHz chip when the word length is set to 10 bits. The FPGA resources shown in this table are exploited from all units of the designed architecture. From this table

Table 1: Utilization summary for the selected device: 3s200ft256-5

Component	Number	%utilization
Slices	206 out of 1920	10%
Slice Flip Flops	266 out of 3840	6%
4 input LUTs	248 out of 3840	6%
bonded IOBs	33 out of 173	19%
MULT18X18s	5 out of 12	41%
GCLKs	1 out of 8	12.5%
DCMs	1 out of 4	25%
7.361ns (Maximum Frequency: 135.851MHz)		

one can see that only five (18×18bits) integrated multiplier is used. Four multipliers are used by a layer which its index in the order sequence, each belongs
to every node, and one multiplier is used by the unique network's activation function that operates in a pipelined fashion. The remaining seven integrated multipliers still available. One device clock manager(DCM) among three other available is used. This DCM is connected to the single used global clock(GCLK) to solve many common problems that may occur. There are two inputs and one output, thus 33 FPGA bounded I/O port are used. The 1st is the input clock (clk) and its word length equal to 1 bit. The 2nd input is the colored pixel(X_i) and its word length equal to 3×10=30 bits. The output port is the pixel class and its word length equal to 2 bits.

The last row of table 1 shows that the designed network can operate as fast as of frequency 135.851MHz, but of coarse if the FPGA chip works on this frequency.

The implementation speed performances are compared to the speed of software implementation of the same segmentation network running on a 2.4GHz Pentium 4, General Purpose Processor:GPP, with 256 Mb RAM and for the IBM *RISC* 350 station that is used in [4]. The time required to classify each pixel by using the FPGA hardware model equals to (23×1/50000000=0.46) µs. This module used a 50MHz, Xilinx Spartan-3 200,000-gate Platform FPGA - XC3S200. While the same function needs (31) ms when implemented

in Pentium 4 GPP, resulting a speedup of (67391). While the speedup that achieved over the IBM *RISC* 350 station is (407).

The speedup can be further increased by factors of 2.71. This is achieved when the designed systems are mapped onto an FPGA model that can operate on the maximum allowable operating frequencies.

The training and testing pixel's samples are gathered from an image that containing different faces, skin and non skin regions taking into account that the color spectrum of this image is wide (see left side of upper row of Fig. 5). Training samples are extracted from skin and non skin regions, then each pixel is labeled with skin:1 or non skin:0 values(see right side of upper row and middle row of Fig. 5).

Figure 5: Original colored, Sample and Segmented images

The segmented images shown in the lower row of Fig. 5 are achieved when the original image pixels are represented as RGB or Yuv respectively.

6 DISCUSSION AND CONCLUSION

In this paper a method to detect and to segment skin color region that is suitable for FPGA VLSI implementation has been proposed. To simplify the FPGA implementation, a saturating linear activation function is used. Each pixel is represented by its RGB and Yuv color components, which are used as inputs by the neural network. The proposed skin detector offers a good trade-off between skin detection performance and implementation complexity. The proposed system gives better performances than sequential computers or even than *RISC* computers.

It has shown from Fig. 5 that neural networks based colored segmentation is well-suited for classification of complex objects like human faces only on the basis of the single pixel's color information. The classification turns out to be very robust even if the available training data is limited. This can be seen from the segmented images where some faces which are not included in the training samples are also segmented correctly according to the generalization property of the ANN. From the two versions of the segmented images, one can see that the performance is largely independent of the chosen color representation (both RGB and Yuv give well results).

References

[1] Kurugollu, F.;Sankur, B. and Harmanc, A. " Image segmentation by relaxation using constraint satisfaction neural network", Image and Vision Computing 20 (2002) 483-497.
[2] Ozkan, M.; Dawant B. M.; Maciunas, R. J." Neural Network based Segmentation of multi model medical images: a comparative and perspective study, IEEE Transactions of Medical Imaging 12 (1993) 534-544.
[3] Uchiyama, T.; Arbib, M. A."Color image segmentation using competitive learning", IEEE Transactions on PAMI 16 (1994) 1197-1206.
[4] Littmann, E.; Ritter, H. " Adaptive Color Segmentation- A Comparison of Neural and Statical Methodes", IEEE Transactions on Neural Networks 8 (1997) 175-185.
[5] Boussaid, F.; Bouzerdoum , A. and Chai,D. "VLSI Implementation of a Skin Detector Based on a Neural Network", ICICS (2005) 1561-1564.
[6] E. Fiesler, "Single Chip CMOS Color Segmenter," POC Final Report, U.S. Department of Energy, Albuquerque Operations Office; Contract No.: DE-FG03-97ER82452, Jan (1998).
[7] J. Zurada, "Introduction to Artificial Neural Systems", PWS publishing company, 1992.
[8] S.V. Kartalopoulos, S.V." Understanding Neural Networks And Fuzzy Logic", IEEE Neural Network Council, Sponsor Richard Saeks, NNC Liaison to IEEE Press(1996).
[9] CoriC, S.; LatinoviC, I.; PavasoviC, A. "A Neural Network FPGA Implementation", 5th Seminar on Neural Network Applications in Electrical Engineering, NEURAL-2000, Faculty of Electrical Engg., Univ. of Belgrade, Yugoslavia (2000) 25-27.
[10] Zhu, J.; Milne, G J.; Gunther, B. K. "Towards an FPGA Based Reconfigurable Computing Environment for Neural Network Implementations", Artificial Neural Networks, 7 – 10 September (1999), Conference Publication, no. 470, ©IEE 1999.
[11] Li, X.; Moussa, M.; Areibi, Sh. "Arithmetic formats for implementing Artificial Neural Networks on FPGAs" Canadian Journal on Electrical and Computer Engineering (in print) (2005).
[12] Savich, A.; Moussa, M.; Areibi, Sh. " The Imact of Arithmatic Representation on

Implementing MLP-BP on FPGAs: A Study", IEEE Transaction on Artificial Neural Networks, vol. XX, no. XX, Sept (2005).
[13] Schoenauer, T.; Jahuke, A.; Roth, U.; Llar, H. "Digital Neurohardware: Principles and Perspectives", Neuronal Networks in Applications-NN'98-Magdeburg (1998).
[14] Syian, M. M.; Klash, H. M.; Mahmoud, I. I.; Haggag, S. S. "Hardware Implementation of Neural Network on FPGA", ICFENCO (2004).

The work was carried out at the college of Engg. University of Mosul