U.P.B. Sci. Bull., Series C, Vol. 78, Iss. 1, 2016 ISSN 2286-3540

HUMAN SKIN DETECTION USING NEURAL NETWORKS AND BLOCK PROCESSING TECHNIQUES

Cătălin Mircea DUMITRESCU[1], Ioan DUMITRACHE[2]

The first stage in algorithms for face detection, hand gesture recognition and many other, is represented by the "skin" / "non-skin" segmentation. Increasing the accuracy of the segmentation algorithm, we improve the overall system performance. In this work we develop and test a human skin recognition system. The system will be optimal in terms of classification performance over processing time ration. To obtain a high accuracy, our system combines information regarding the texture and colour features obtained by using vector processing techniques and classic techniques. To reduce the overall processing time, the features will be extracted using block processing techniques.

Keywords: skin recognition, skin detection, neural networks, vector processing, texture analysis

1. Introduction

The skin detection / segmentation algorithm is an important step for computer vision applications. In general, skin segmentation methods rely only on skin colour information. By using colour information only, these techniques present a high processing speed, but also a low accuracy. The human skin colour depends on many variables such as: subject parameters (age / race / sex), body part, image parameters (lighting, camera position). All these variables lead to a high error classification rate. Although the different lighting conditions can be partially avoided, by using the ***YCbCr*** colour space, there are still many unanimated real world objects that have a chrominance in the range of the human skin. Thereby using only colour information, those objects will be wrongly classified as human skin. By combining texture features with skin colour features the overall classification accuracy can be increased.

A colour image ($NxMx3$ matrix) represented in any colour space, such as ***RGB***, is composed out of three colour layers (in ***RGB*** space these layers are R-red, G – green, B – blue). Each individual channel that composes a colour image can be considered a monochrome image, thus it can be processed separately.

[1]PhD student, Facultyof Automatic Control and Computers, University POLITEHNICA of Bucharest, e-mail: dumitrescu.catalin.m@gmail.com
[2] Prof., Facultyof Automatic Control and Computers, University POLITEHNICA of Bucharest

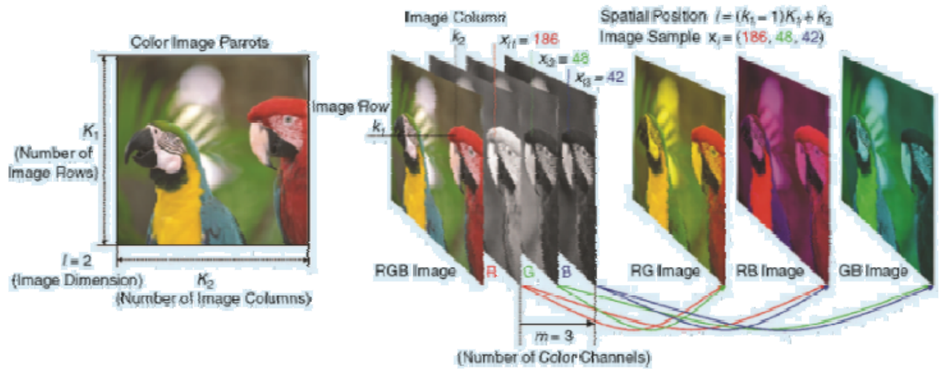


Fig. 1. Colour image decomposition

In all existing methods, the colour features extraction is done by treating each individual colour channel as a monochrome image. However, by doing so, they don't take into account the correlation that exists between the colour components of an ***RGB*** image. In this way, the correlation that exists between the colour components of natural images represented in a correlated colour space, is disrupted. In order to take advantages of this correlation, a ***RGB*** image must be treated as a 3-dimesions vector and processed using vector processing techniques. Some vector processing techniques for colour images are presented in [1].

In [2], it was presented the first human skin segmentation system that uses vector processing techniques to extract colour features. The image is treated as a vector field, each colour pixel representing a vector with three dimensions.

The algorithm described in [2] presents a high accuracy, over 98% detection rate, but the processing speed is very slow. This is due to the fact that each image pixel is treated separately and so, the correlation that exists between neighbour pixels is ignored.

In this paper an optimal *"skin"* pixel segmentation method will be presented; it uses vector processing techniques and texture features. The proposed system maximizes the classification performance over the processing time ratio. The features are extracted using a square processing window that is shifted using a fix step across the entire image.

The optimization criteria that must be maximized is:

$$C = \frac{R}{t}\left(\frac{1}{FP} + \frac{1}{FN}\right) \tag{1}$$

where: C – optimization criteria, R – classification rate $R = TP + TN$, TP – true positive, TN – true negative, FP – false positive rate, FN – false negative rate and t – processing time

2. State of the art

Most of the existing skin segmentation techniques classify each pixel of a colour image into *"skin" / "non-skin"* categories, only on the basis of pixel colour information. They are based on: Bayes classifier, Kohonen neural networks (Self Organizing Map), single Gaussian skin distribution model, multiple Gaussian clusters. A survey that compares these methods is presented in [3]. The detection rate results, as reported by the authors, using the Compaq dataset [4] are between 78% - 94.7% and the false positive rate ranging between 8.5% - 30.2%, as shown in (Table 1).

Table 1

Performance of different skin detectors

Method	Detection Rate	False Positive
Bayes SPM (RGB) [4]	80% / 90%	8,5% / 14,2%
Bayes SPM (RGB) [5]	93,4%	19,8%
Maximum Entropy Model in RGB [6]	80%	8%
Gaussian Mixture models in RGB [4]	80% / 90%	9,5% / 15,5%
Self-Organizing Map in TS [7]	78%	32%
Single Gaussian in CbCr [8]	90%	33,3%
Elliptical boundary model in CIE-xy [8]	90%	20,9%
Thresholding of I axis in YIQ [5]	94.7%	30.2%
Gaussian Mixture in IQ [8]	90%	30%

Another study regarding the skin pixel segmentation is presented in [9], by S. Phung et al., treating three issues of the skin pixel classification: colour representation, colour quantization, and classification algorithm. The study was conducted on the ECU face and skin detection database. The MLP (multi-layer perceptron) classifier tested by S. Phung achieved an 89.54% detection rate.

A neural network *"skin"* classifier is proposed by K. Bhoyar and O. Kakde in [10]. The neural network has a *"feedforward"* architecture with 3 neurons in the input layer, 5 neurons in the hidden layer and 2 neurons in the output layer. The three colour components of each pixel (in ***RGB*** space) represent the inputs of the neural network. The first neuron of the output layer is the *"skin"* class and the second neuron, the *"non-skin"* class. The best performance obtained by this classifier was a 95% detection rate and a 3% false positive rate.

The first method that combines the skin colour features with texture features is proposed by N.K. Al Abbadi, N.S. Dahir and Z.A. Alkareem in [11]. By using both colour and texture features, the system has a detection rate of 96%.

The system presented in [2] is the first *"skin" / "non-skin"* classification system that uses vector processing techniques for extracting the texture and colour features. The classification rate in this case is over 98%.

The big drawback of the system presented in [2] is the time needed to process the image, over 400 seconds, depending on the image size. This is due to the fact that, in the features extraction phase, the correlation that exists between the neighbouring pixels is ignored and every pixel is processed separately.

3. Feature extraction

To extract the features vectors, the image is scanned from left to right and top to bottom, with a predefined fix step using a square processing window, starting from the top left corner (see Fig. 2). For each position of the processing window, a features vector, composed by colour information (extracted using vector and classic techniques) and texture information, is extracted.

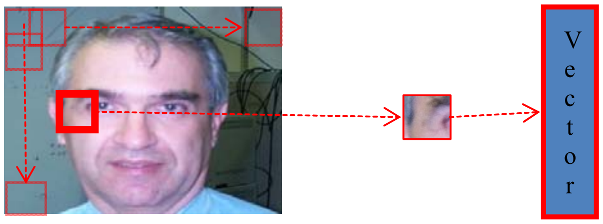


Fig. 2. Features vectors extraction

3.1. Colour features extracted using vector processing techniques

Assuming that the texture is a random process ($f(x) \in \mathbb{R}^3$), for a given region R containing N pixels, several k order statistical moments can be defined:

$$m_k = \frac{1}{N} \sum_{x \in R} f(x)^k \tag{2}$$

$$\overline{m_k} = \frac{1}{N} \sum_{x \in R} (f(x) - m_1)^k \tag{3}$$

The main statistical moments used to characterize the colour distribution are: the first-order moment (m - mean colour), the second-order moment (σ - standard deviation), and the third-order moment (θ - skewness of colour). In this work only two statistical moments are used to characterize the colour distribution: m mean colour and σ standard deviation.

$$m = \frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} p_{ij} \tag{4}$$

$$\sigma = \left[\frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} (p_{ij} - m)^2 \right]^{1/2} \tag{5}$$

where: M and N are the image dimensions, p_{ij} is the colour pixel in the i^{th} row and j^{th} column of the image.

An ***RGB*** colour space image is composed of three colour components (R – red, G – green, B – blue). Each colour component can be regarded as a monochrome image. Traditional methods of extracting the k order statistical moments often involve the application of the previously defined formulas on each colour channel separately, resulting a separate value for each colour component:

$$m_c = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} p_{ij}^{c} \tag{6}$$

$$\sigma_c = \left[\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}\left(p_{ij}^{c} - m_c\right)^2\right]^{1/2} \tag{7}$$

where: M and N are the image dimensions, p_{ij}^{c} is the value of the c^{th} colour component of the colour pixel in the i^{th} row and j^{th} column of the image.

However, treating each colour channel separately disrupts the correlation that exists between the colour components of natural images represented in a correlated colour space, such as ***RGB***. Each processing step is usually accompanied by certain inaccuracy that leads to colour artefacts. In order to reduce the probability of colour artefacts appearance, we propose that the statistical moments, m - mean colour and σ - standard deviation be determined using vector processing techniques. Vector processing techniques treat the colour image as a vector field. Assuming a colour image in the ***RGB*** space $p:\mathbb{N}^2 \to \mathbb{N}^3$, each pixel $p_{ij} = \left[p_{ij}^{R}\ p_{ij}^{G}\ p_{ij}^{B} \right]^T$ represents a three-component vector in a colour space. The colour image p is a two-dimensional matrix of three components vectors.

Because each pixel is treated as a three-dimensional vector, a distance between two vectors must be defined. The most commonly used distance between two colour vectors $p_1 = [p_1^{R}\, p_1^{G}\, p_1^{B}]^T$ and $p_2 = \left[p_2^{R}\ p_2^{G}\ p_2^{B} \right]^T$ is the generalized weighted Minkowski metric:

$$d\left(p_1, p_2\right) = \left|p_1 - p_2\right|_L = c\left(\sum_{k=1}^{3}\xi_k \left|p_{1k} - p_{2k}\right|^L\right)^{\frac{1}{L}} \tag{8}$$

The non-negative scaling parameter c is a measure of the overall discrimination power. The exponent L defines the nature of the distance metric. The parameter ξ_k measures the proportion of attention allocated to the dimensional component k and, therefore $\sum_{k=1}^{3} \xi_k = 1$.

To determine the mean colour of an image region, we will not be using the mathematical equation (6), but a three-dimensional median filter. Each output pixel of the filter represents the median value in the M by N by 3 neighbourhood centred on the corresponding pixel.

To determine the standard deviation, we are using the $L2$ metric (Euclidian distance) as the distance between two vectors:

$$\left|p_{ij} - m\right| = \left[\sum_{c=1}^{3} \left(p_{ij}^{c} - m_c\right)^2 \right]^{1/2} \tag{9}$$

Substituting (9) in (5), yields:

$$\sigma = \left[\frac{1}{MN} \sum_{i=1}^{M} \sum_{j=1}^{N} \sum_{c=1}^{3} \left(p_{ij}^{c} - m_c\right)^2 \right]^{1/2} \tag{10}$$

3.2. Colour features extracted using classic processing techniques

To characterize the colour distribution in a non-vectorial manner, we use the Shannon entropy for each colour channel of the ***RGB*** triplet. Each colour component is treated separately as a monochrome image. For each of the three colour planes (R G B), we are calculating the value of the Shannon entropy:

$$H_c = -\sum_{i=0}^{255} p_i * log_2\left(p_i\right) \tag{11}$$

where: p_i represents the probability of a pixel whose colour component c has the value $i \in [0 \ldots 255]$.

3.3. Texture features

In the 1970's, Haralick et al. [12] propose the usage of the grey level co-occurrence matrix (GLCM) as a method of texture characterization. For a defined region R of the studied texture and for a given spatial displacement vector t, the components of the grey-level co-occurrence matrix are defined for all possible pairs of grey levels (a,b) as:

$$M_t(a,b) \triangleq Card\left\{(x, x+t) \in RxR \middle| f(x) = a \,\&\, f(x+t) = b\right\} \tag{12}$$

where: $M_t(a,b)$ represents the number of pixel pairs in the defined region R, separated by the spatial displacement vector t, whose grey level values are equal to a and respectively b.

Because co-occurrence matrices are typically large and sparse, various metrics of the matrix are often taken to get a more useful set of features. In this work, the following texture features are computed using the grey level co-occurrence matrix:

- Homogeneity:

$$O = \frac{1}{N_{nz}} \sum_a \sum_b \frac{1}{1+|a-b|} M_t(a,b) \tag{13}$$

Contrast:

$$C = \frac{1}{N_{nz}} \sum_{a,b} |a-b|^2 M_t(a,b) \tag{14}$$

Correlation:

$$B = \frac{1}{N_{nz}} \sum_a \sum_b \frac{(a-m_a)(b-m_b) M_t(a,b)}{\sigma_a \sigma_b} \tag{15}$$

where: N_{nz} represents the number of non-zero elements in the co-occurrence matrix, m_a and m_b are the mean values along the lines / columns, σ_a and σ_b represents the corresponding dispersions.

The spatial displacement vector t, used in this work, has two values $t = \{(0,1)$ and $(-1,0)\}$. As a result, six texture features measures are extracted (three for each value of the displacement vector t).

4. The proposed algorithm

In this section, we present the new segmentation algorithm. By taking advantage of the of the correlation that exists between neighbouring pixels in natural images, the proposed algorithm reduces the amount of processed data, increasing the processing speed.

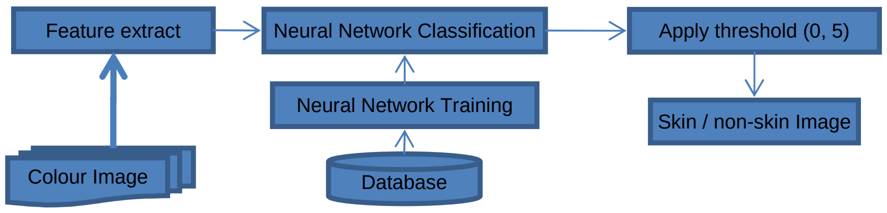


Fig. 3. System architecture

The proposed algorithm (see Fig. 3) is composed of the following main processes: <u>SKIN SAMPLES DATABASE CREATION, NEURAL NETWORK TRAINING AND IMAGE SEGMENTATION INTO “*SKIN*” / “*NON-SKIN*” REGIONS USING THE NEURAL NETWORK.</u>

The “*skin*” / “*non-skin*” image segmentation process consists of several stages. First, using a fixed dimensions square processing window, we extract the feature vectors. The image is scanned with a predefined step from left to right and top to bottom and for each central pixel we compute a features vector. Next, with the help of a neural network, we determine the degree of belonging to one of the two classes (“*skin*” and “*non-skin*”) for each features vector.

By using a processing window step greater than 1 pixel, the output of the system will be sparse matrix (if we consider the segmented image as a matrix). The values of the pixels that are between two centre points of the processing window are equal to zero because these pixels are excluded from processing. For the intermediary pixels, the membership degree values at the two classes (“*skin*” and “*non-skin*”) will be computed using an interpolation filter.

Let $f \in \mathbb{R}^2$ be the image obtained from the application of the neural network for a processing window of $n \times n$ pixels (n odd) and a window shift of k pixels, the interpolation filter equation is:

$$F(x,y) = \frac{\sum_i \sum_j f(x+i, y+j) \Big/ \sqrt{(1+\mathrm{i})^2 + (1+\mathrm{j})^2}}{\sum_i \sum_j 1 \Big/ \sqrt{(1+\mathrm{i})^2 + (1+\mathrm{j})^2}} \left| \begin{array}{l} i, j \in \left[-\frac{n\text{-}1}{2}, \ldots, \frac{n\text{-}1}{2} \right] \\ i \bmod k = 0 \\ j \bmod k = 0 \end{array} \right. \qquad (16)$$

Using a predefined threshold value and the degree of belonging to neural network “*skin*” output, the input pixels are classified as human skin, if the network output is greater than the threshold, or non-skin otherwise. The used value for the threshold is 0.5.

The database used to test the algorithm is composed of 942 human skin samples (Fig. 4) and 1348 non-skin samples (Fig. 5), with the minimum dimensions of 51 by 51 pixels, extracted from Georgia Tech Face Database [13]. Resulting in more than two million pixels of “*skin*” and over three million pixels of *“non-skin”*. The samples were extracted manually from more than 400 colour images.

Fig. 4. Human skin samples

Fig. 5. Non-skin samples

To classify the pixels into the *“skin”* / *“non-skin”* classes, a *“feed-forward”* multilayer perceptron is used. The proposed architecture of the neural network consist of three layers, as follows: Input layer with 16 neurons, Hidden layer with 60 neurons and Output layer with 2 neurons. The architecture is identical to the one presented in [2]. The number of hidden neurons was established through trial and error method.

The activation function used is the tan-sigmoid, for both the hidden and the output layer. This function satisfies the differentiability and monotonicity requirements imposed by the *“back-propagation”* training algorithm.

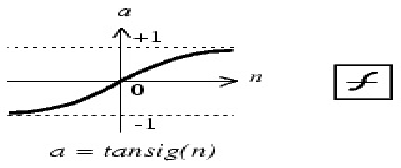


Fig. 6. Neuron activation function

The input of the neural network is the features vector and it consists of 16 components:

- three that represent the central pixel’s colour in ***YCbCr*** colour space;
- six from the texture features defined by (13), (14) and (15);
- three values that represent the Shannon entropy (11) of each ***RGB*** colour component;
- one value for standard deviation (10);
- three values representing the mean colour, calculated using a three-dimensional median filter.

The neural network classifier has two outputs, one for each segmentation class *“skin”* / *“non-skin”*. The output of the neural network represents the degree of membership of the input vector to one of the two classes *“skin” / “non-skin”*, the value is between 0 (0%) and 1 (100%).

The network is trained using *“back-propagation”* algorithm, *Levenberg - Marquardt* method. This algorithm minimizes the error between the desired output and the actual output. As performance criteria we use the mean square error (MSE).

5. Case study

Once the neural network architecture has been established, the system performance is influenced by two critical parameters: the size of the processing window and the window shift size. The optimum values for these parameters are experimentally determined in the following section.

The neural network training process is done using *“skin”* and *“non-skin”* samples from the database. More than 400.000 *“skin”* pixels and 500.000 *“non-skin”* pixels are used to train the neural network. The training samples are processed using five processing windows (9x9, 11x11, 13x13, 15x15 and 21x21 pixels) and a window shift of 3 pixels. The first output of the neural network is assumed to be 1 for *“skin”* inputs and the second output to be 1 for *“non-skin”* inputs.

The results obtained after the neural network training stage are presented in the confusion matrices from

Fig. 7 till Fig. 11. Fig. 12 shows evolution of the performance coefficient.

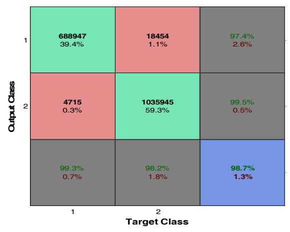


Fig. 7. 9x9 pixels

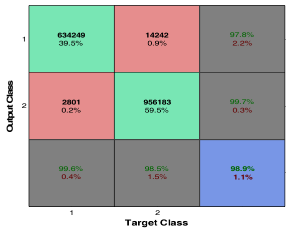


Fig. 8. 11x11 pixels

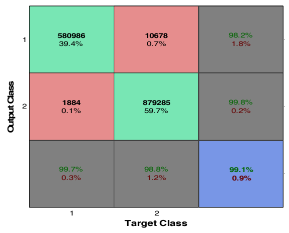


Fig. 9. 13x13 pixels

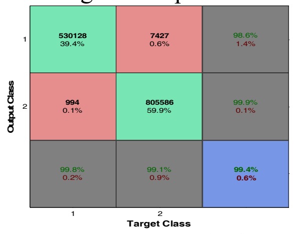


Fig. 10. 15x15 pixels

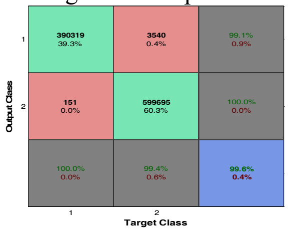


Fig. 11. 21x21 pixels

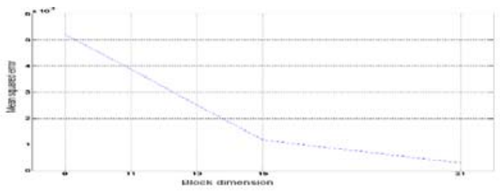


Fig. 12.The performance coefficient variation

By analysing the confusion matrices (

Fig. 7 to Fig. 11) it can be observed that, when a processing window of 21x21 pixels is used, the system presents the best performance. By looking at the evolution curve of performance coefficient in relation to the size of the processing window (Fig. 12), the minimum value corresponds also for a 21x21 pixels processing window.

After the training phase, the following recognition rates have been obtained: *"skin"* detection rate (output class number 1) 99.1%; *"non-skin"* detection rate (output class number 2) 100%; overall detection rate 99.6%; error rate 0.4%.

Looking at the neural network error histogram (Fig. 13), we can observe that the majority of pixels present a membership error that is less than 5%.

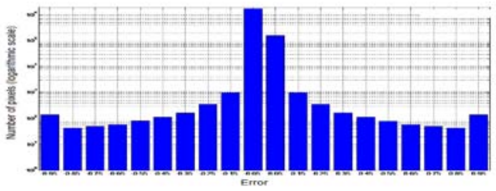


Fig. 13. Neural network error histogram

The second configuration parameter, the window step size, is determined in an experimental manner. To do this, we randomly selected 50 *"skin"* samples and 50 *"non-skin"* samples from the database. After selecting the samples, each of the five neural networks (one for every block size) is tested using a window step size between 1 and 7 pixels.

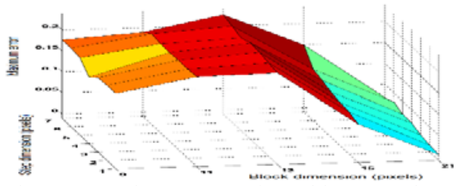


Fig. 14. Maximum error for „skin" samples

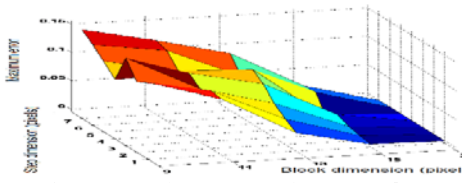


Fig. 15. Maximum error for *„non-skin"*

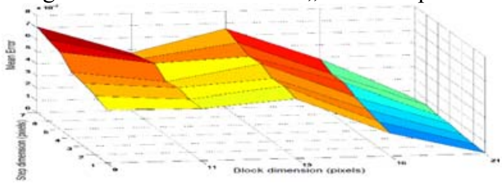


Fig. 16. Mean error for „skin" samples

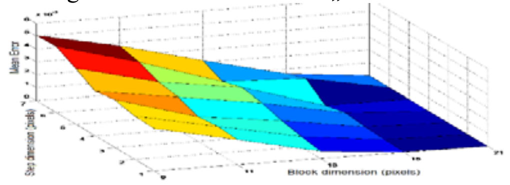


Fig. 17. Mean error for *„non-skin"* samples

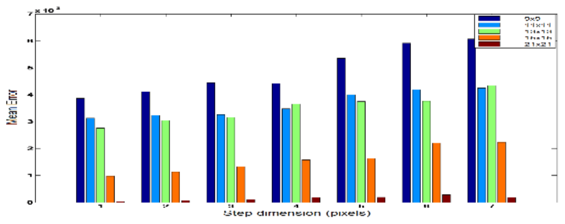


Fig. 18. Mean total error

Analysing the evolution of the mean and maximum errors (Fig. 14, Fig. 15, Fig. 16, Fig. 17 and Fig. 18), we can observe that the algorithm presents the smallest error when it uses a window shift of 1 pixel and a processing window of 21x21 pixels.

When using processing windows of 9x9, 11x11, 13x13 and 15x15 pixels, the system has a recognition error greater than 6% only for 3 samples out of 100 (see sample error rate plots **Error! Reference source not found.** to Fig. 22). This is due to the fact that those 3 samples contained facial hair, or were poorly illuminated. In the case of the 21x21 pixels processing window (**Error! Reference source not found.**), the samples error rate is always under 2% (for all window step values).

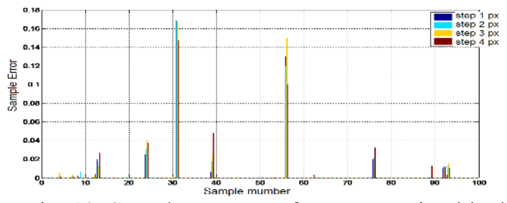


Fig. 19. Sample error rate for a processing block of 9x9 pixels

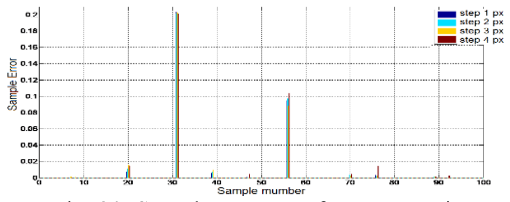


Fig. 20. Sample error rate for a processing block of 11x11 pixels

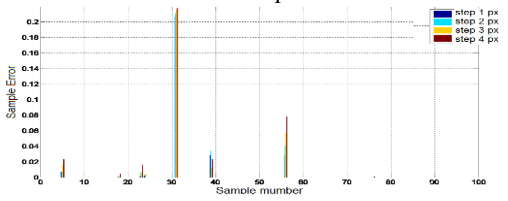


Fig. 21. Sample error rate for a processing block of 13x13 pixels

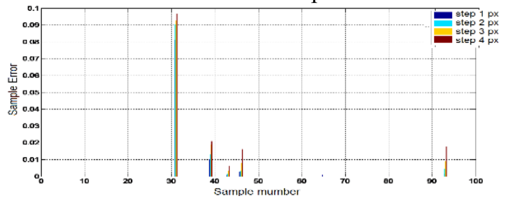


Fig. 22. Sample error rate for a processing block of 15x15 pixels

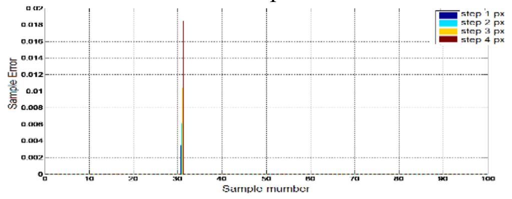


Fig. 23. Sample error rate for a processing block of 21x21 pixels

Next phase is to test the trained neural networks on real images. Because the highest errors are obtained for a window step size of 5, 6 and 7 pixels, these three values are excluded for the next phase. The images are picked in a random order and manually segmented into *"skin" / "non-skin"*.

In Fig. 24 (a – block size 9x9 pixels, b – block size 11x11 pixels, c – block size 13x13 pixels, d – block size 15x15 pixels, e – block size 21x21 pixels, f – original image) is presented one of the test results for a window shift of 3 pixels.

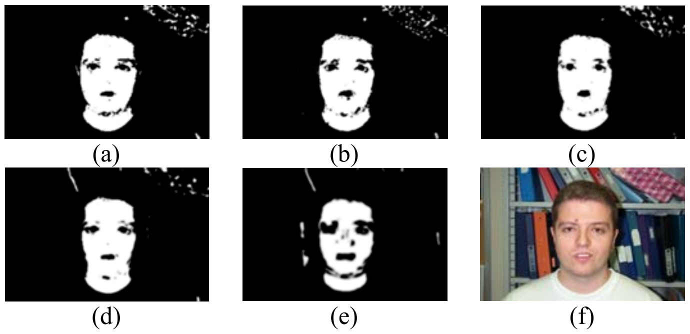


Fig. 24. Results obtained on a real image

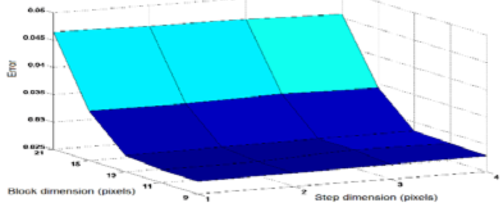


Fig. 25. Classification error according to the window size and window shift

Even if the smallest false positive error rate was obtained for a processing window size of 21x21 pixels, the overall segmentation error in this case is the biggest (Fig. 24 and Fig. 25). This is because the processing window is too large compared to the face size. In Fig. 24 the face shape is distorted.

The best segmentation results are achieved when processing windows of 9x9 pixels, 11x11 pixels and 13x13 pixels are used; fact also confirmed by analysing Fig. 26 to Fig. 28.

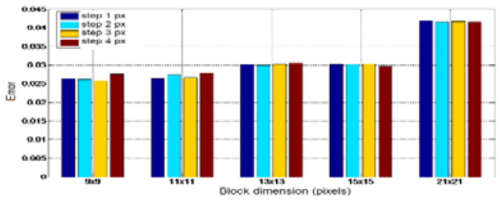


Fig. 26. The classification error by window size and window shift

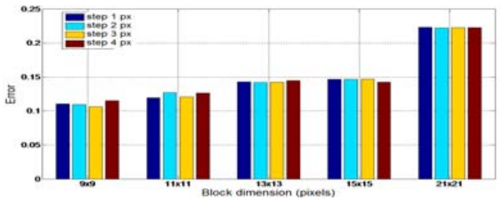


Fig. 27. The false negative error rate by window size and window shift

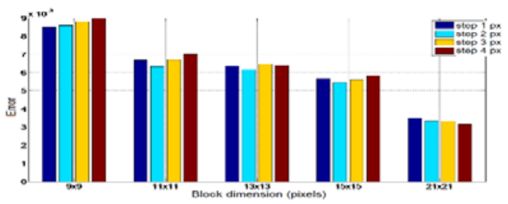


Fig. 28. The false positive error rate by window size and window shift

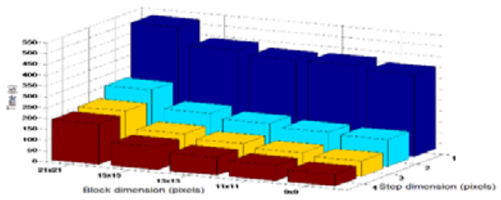


Fig. 29. The evolution of the processing time by window size and window shift

Fig. 26 depicts the classification error variation by window size and step size. The smallest classification error is achieved when the processing window has a size of 9x9 pixels and the window shift is 3 pixels.

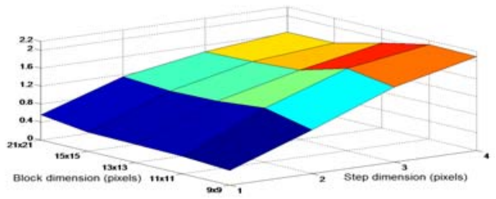
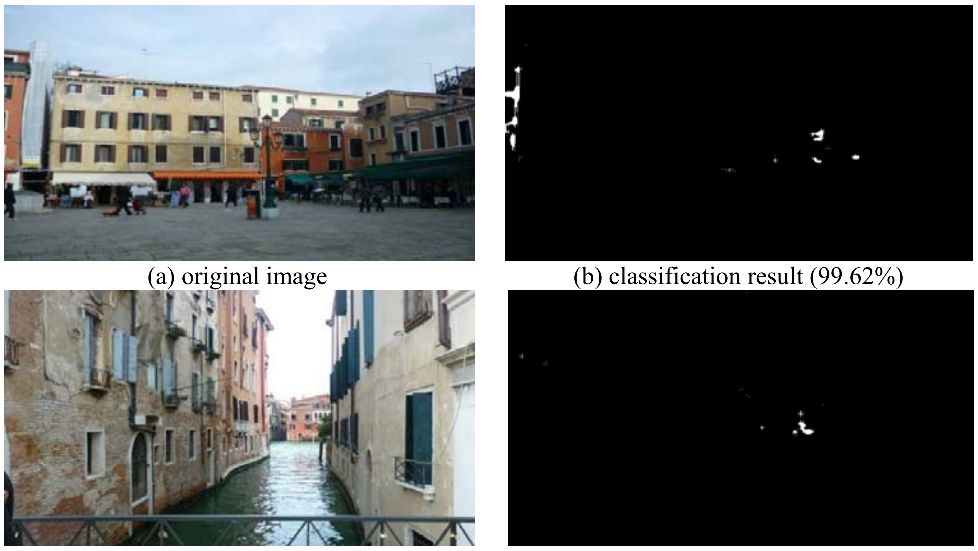


Fig. 30. The optimization criteria versus the window size and shift

Fig. 30 shows the evolution of the optimization criteria (1) versus the processing window size and shift. When a window of 11 pixels and a step size of 3 pixels is used, the optimization criteria reaches the maximum value.

According to the results presented in Fig. 31, testing the system with *"non-skin"* images, the classification rate is over 99.5%.


(a) original image (b) classification result (99.62%)

(c) original image (d) classification result (99.90%)

Fig. 31. Proposed algorithm *"non-skin"* test results

By comparing the new system with the one presented in [2], we can conclude that the new version (processing window size of 11 pixels and a window shift of 3 pixels) offers better classification performances (Fig. 32). It also has a lower false negative error rate (Fig. 33) and a lower false positive error rate (Fig. 34). The processing time drops from almost 400 second to 83 seconds. The tests were performed on a 3.2GHz AMD Phenom x64 (6 cores) with 8GB RAM system running Windows 7 x64 and Matlab R13.

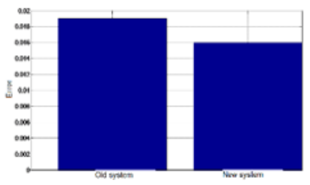


Fig. 32. Classification error old vs. new system

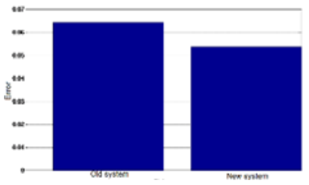


Fig. 33. False negative error rate old vs. new system

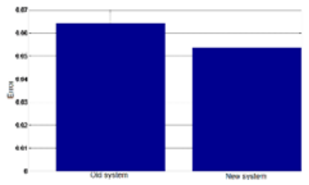


Fig. 34. False positive error rate old vs. new system

6. Conclusions

Looking at the results presented above (Fig. 14 to Fig. 28), we can conclude that the system provides a high *"skin"* pixel classification rate. When using real images, the system achieved a pixel classification rate of 97.4%, with 1.5% smaller than the one achieved during the neural network training stage (Fig. 8).

The *"skin"* pixels classification rate obtained both during the neural network training stage, as well as in previous test, was over 97%.

Even if the system that used a processing window of 21x21 pixels had the best results during the neural network training stage (classification rate of 99.6%, false negative error rate of 0.9% and a false positive error rate of 0.0%), during the test on real images, the system performance declined. The performance degradation is because the processing window size is too large compared to the images used in the tests (on special the size of the human face).

The presence of facial hair (beard, moustache) or other details (like necklaces, earrings), and a poor lighting of the subject lead to a worsening of the pixel classification rate.

In conclusion, the algorithm developed in this work presents better performances than the one in [2] and a five-times lower processing time.

REFERENCES

[1] *R. Lukac, B. Smolka, K. Martin, K. N. Plataniotis and A. N. Venetsanopoulos*, "Vector Filtering for Color Imaging", in IEEE Signal Processing Magazine, 2005, pp. 74-86

[2] *C. M. Dumitrescu and I. Dumitrache*, "Human Skin Detection Using Texture Information and Vector Processing Techniques", The 18th International Conference On Control Systems and Computer Science, 2011

[3] *V. Vezhnevets, V. Sazonov and A. Andreeva*, "A Survey on Pixel-Based Skin Color Detection Techniques", In Graphicon03, 2003

[4] *M. J. Jones and J. M. Rehg,* "Statistical Color Models with Application to Skin Detection", in Proceedings 1999 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1999

[5] J. *Brand and J. Mason,* "A Comparative Assessment of Three Approaches to Pixel Level Human Skin-Detection", In Proceedings of the International Conference on Pattern Recognition, **vol. 1**, 2000, pp.1056–1059

[6] *B. Jedynak, H. Zheng, M. Daoudi and D. Barret*, "Maximum Entropy Models for Skin Detection", In Proceedings Third Indian Conference on Computer Vision, Graphics and Image Processing, 2002, pp. 276-281

[7] *D. Brown, I. Craw and J. Lewthwaite*, "A SOM Based Approach to Skin Detection with Application in Real Time Systems", In Proceedings of the British Machine Vision Conference, 2001.

[8] *J. Y. Lee and S. I. Yoo*, "An Elliptical Boundary Model for Skin Color Detection", In Proceedings of the 2002 International Conference on Imaging Science, Systems, and Technology, 2002

[9] *S. Phung, A. Bouzerdoum and D. Chai,* "Skin Segmentation Using Color Pixel Classification: Analysis and Comparison", in IEEE Trans. Pattern Analysis and Machine Intelligence, **vol. 27**, no. 1, Jan. 2005

[10] *K. Bhoyar and O. Kakde*, "Skin Color Detection Model Using Neural Networks and Its Performance Evaluation", Journal of Computer Science 6, 2010

[11] *N. Al Abbad, N. Dahir and Z. Alkareem*, "Skin Texture Recognition Using Neural Networks", In International Arab Conference on Information Technology, 2008

[12] *R. Haralick, K. Shanmugam and I. Dinstein*, "Textural Features for Image Classification", in IEEE Trans. On Systems, Man and Cybernetics, **vol. Smc-3**, no. 6, 1973, pp. 610 – 621

[13] "Georgia Tech Face Database", available at ftp*://ftp.ee.gatech.edu/pub/users/hayes/facedb*