American Journal of Applied Sciences

Original Research Paper

Faces Detection using Skin Color, Region-Props, Bounding-Box and Neural Networks System

Gamal Mohamed Behery

Department of Mathematics, Faculty of Science, Damietta University, Egypt

Article history
Received: 01-03-2016
Revised: 15-05-2016
Accepted: 16-05-2016

Email: behery2911961@yahoo.com

Abstract: Face detection is an important prior step for face recognition system which is widely used in security systems, face verification systems, telecommunication, video surveillance, facial expressions recognition, status authentication, etc. The proposed system is applied on many images which contain persons and extract the faces out of there automatically. The skin color, region-props and bounding-box are used as preprocessing tools for extracting regions. The Neural Networks (NNT) are used to recognize these regions are faces or not. This system has the ability to detect faces with various image conditions: Different poses and facial expirations, different and complex backgrounds and different luminance and lighting conditions. The system is tested on several color images. The detection rates for used databases are 67.6% for images which has background non-luminous, 88% for very dense images and 100% for non-dense images with a luminous background.

Keywords: Neural Network, Skin Color, Region Props, Bounding Box, Face Detection

Introduction

Human face detection plays an important role in applications such as video surveillance, human computer interface, face recognition, face image database management and expression recognition (Liu *et al*., 2003). Many existing detection techniques suffer under scale variation, pose variation, illumination changes and complex backgrounds (Osman *et al*., 2012). Face detection searching people, pornographic filtering and hand tracking usually used the skin color detection (Sultana *et al*., 2014). Pixels' color and/or pixels' texture are used to determine the skin or non-skin. Face detection involves many research challenges such as scale, rotation and pose and illumination variation. The skin color detection problems are: The representation of the skin color distribution model that is invariant or least sensitive to changes in illumination condition; many objects in the real world may possess almost similar skin-tone color such as wood, leather, skin-colored clothing, hair and sand; and skin color is different between races and can be different from a person to another even with people of the same ethnicity. Many articles are developed a skin color classifier based on pixel-based using RGB ratio model. It is a newly method that belongs under the category of an explicitly defined skin region model (Gondaliya *et al*., 2015; Siddiqui and Wasif, 2015; Ban *et al*., 2014).

Combining geometry constraints, face mask and skin-color model are used for face detection algorithm of color images which have the complex background (Ayachi *et al*., 2012). The skin color segmentation and edge detection are used to separate all non-face regions from the candidate faces. The system uses self-organizing Takagi-Sugeno-type fuzzy network with support vector to determine which is face or non-face (Chen *et al*., 2013; Al Haj *et al*., 2007; Juang and Shiu, 2008).

The wavelet with lower vanishing moments is better than the wavelet with higher vanishing moments for solving the problems of face detection (Nasrollahi *et al*., 2008). A face detection algorithm; whose features include a fast and regular search method, a local histogram equalization of face candidates and a frontal face classifier is presented (Patilkulkarni and Lakshmi, 2013). An algorithm for eye state detection can find the face area in the input image (Chen *et al*., 2008). The concepts of facial geometry are used to locate the mouth, eyes and nose positions (Kalbkhani *et al*., 2013). Some articles presented techniques for detecting faces in color images using various color models and edge detection (Tiwari and Ahmed, 2013; Goswami *et al*., 2013;

Perumal and Perumal, 2013). A face detection method presents controlled weights on the three components of HSV color model these components are Hue (H), Saturation (S) and Intensity value (V) as explained in (Rewar and Lenka, 2013).

A lip detection algorithm is applied for finding lip area in color face images (Ikeda, 2003). A boosting color component feature selection framework is designed for finding the best set of color component features from various color spaces, aiming to achieve the best face recognition performance for a given face recognition task. A detection model of beach image is used to solve problems of erotic image detection (Zali-Vargahan *et al*., 2013; Yu and Zhu, 2013).

In this article, a robust and efficient system is applied for face detection of color images. The rest of paper is organized as follows. Section 2 presents the proposed system. The Neural Networks (NNT) system is illustrated in section 3. Section 4 shows the applications and discussion. The last section concludes the work.

Proposed System

The proposed system consists of two main phases as shown in Fig. 1. The first phase is used for object detection and the second phase is to design, train and test the proposed NNT system for face classification.

Objects Detection Phase

This procedure is started by determining the objects that may be faces. The object detection phase has three processes. They are skin color segmentation, cover objects region and the objects acquisition. The details of these processes are described as follows.

Skin Color Segmentation

This process converts the RGB image to YCbCr color image using a specific RGB-YCbCr conversion matrix (Li *et al*., 2013). This conversion makes the detection of the skin regions easier. The YCbCr image is obtained by converting the loaded image to YCbCr color space; Fig. 2a. YCbCr color space is useful for skin color segmentation which will identify some pixels that have potential skin color. In order to obtain the skin color values, separate the three components of YCbCr into Y, Cb and Cr.

The resulting YCbCr color image should satisfy the following three rules to aid the task of human skin color segmentation (Osman *et al*., 2012):

$$Y > \alpha,\ Cb > b \ \ and \ \ Cr > c$$

where, the initial values of a, b and c are 120, 95 and 110 respectively. They kept looking for better values in order to enhance the skin color segmentation results. They found other values that have been used in some researches (Lin, 2007) and these values gave better results than the results obtained using the initial values. The new values are 80, 85 and 135. The values of Cb, Cr and Y are restricted as follows:

$$85 \le Cb < 135, 135 \le Cr < 180 \ and \ Y \le 180 \quad (1)$$

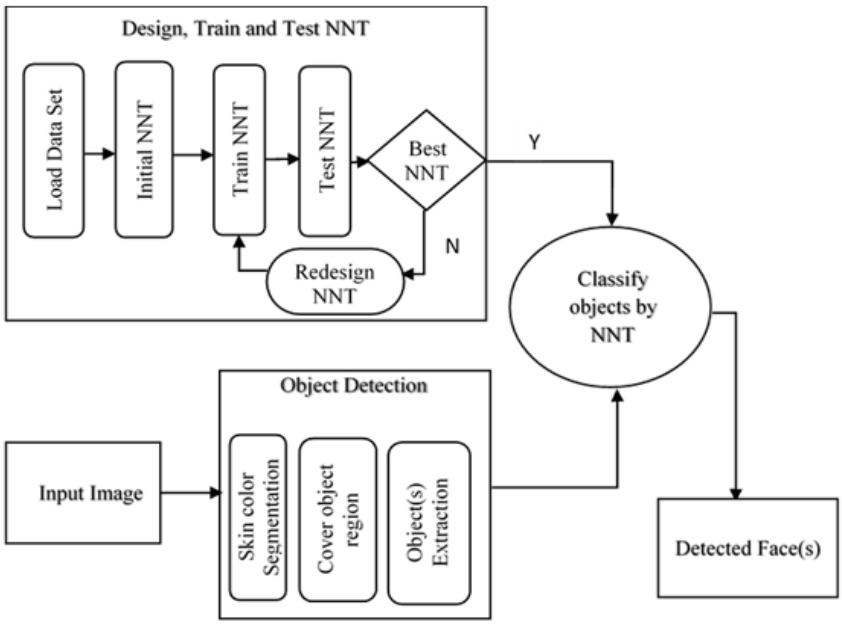

Fig. 1. The overview of proposed system

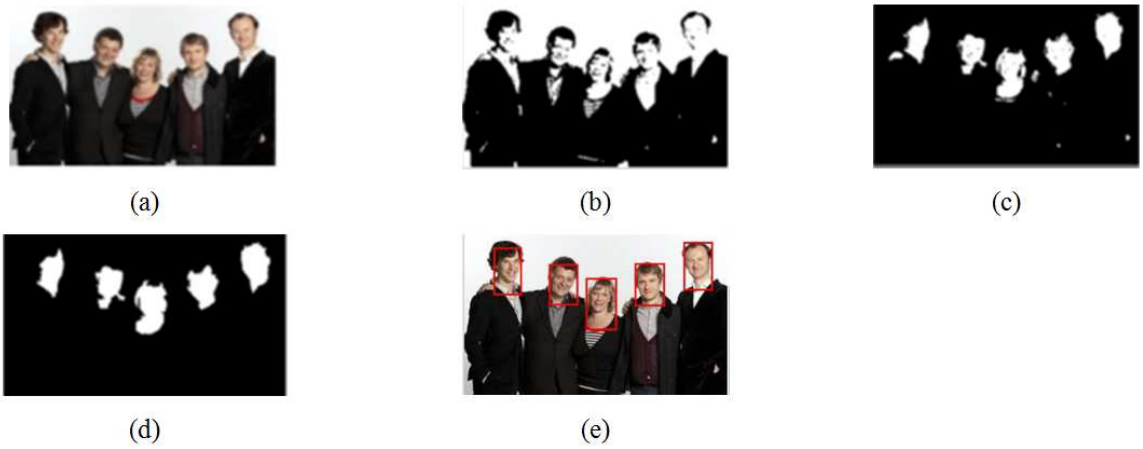

Fig. 2. Object detection sequences, (a) The original input image, (b) Binary image after the skin color segmentation using initial values, (c) Binary image after skin color segmentation using new values, (d) The connected components in the image, (e) Bounding boxes around the labeled regions

After that, a binary image is obtained via pixel values comparison, the pixels that satisfy skin color values will hold 1 (white) others will hold 0 (black) according to Equation 1; Fig. 2b. This is done by first creating a zero matrix with rows and columns as height and width of the original image respectively. Then the positions of values 1 are used to identify the objects in the original image. These objects pixels may contain black noise pixels; Fig. 2c. These pixels must be transferred into white; Fig. 2d.

Cover Objects Region

In this stage, boundaries of objects are determined by applying 4-connected methodology on the filled binary image. This step will identify the coordinates of potential face regions. The following four steps are applied to specify these objects:

- Find the 4-connected components in the image
- Usethe Matlab *regionprops* function to find the structure array for this region
- Isosceles triangle is formed to prepare as cropped potential face region
- Use the Matlab *bound box* function to capture the smallest rectangle by determining top left coordinate, width and height of the labeled region. So, the potential face regions are specified as shown in Fig. 2e

These four steps must be applied for all objects.

The Objects Acquisition

The determined objects from previous step are cropped from the original image. Then, the cropped objects are resized to 64×64 and enhanced by applying histogram equalization. Now the objects are organized as inputs to the NNT by converted each one to a vector.

By using the pervious information of object(s) coordinates, one can capture that object (s). *Extract Face and Nonface* function represents the main functionality and is the last implemented function in this phase, so the output of this function is the original input image of NNT system.

Finally a red square will be drawn on the original input image using the coordinates of sub-image/s that results in a face from previous step; Fig. 2e.

The above three stages are presented in two functions. The first function is called *Object Detection Regions* and designed for the first two stages. The other function is called *Extract Face and Nonface* and designed for the third stage. These functions are given as follow:

Function [connected_component]= Object Detection Regions (Image)
{
Gray_image = convert the color image to gray image
Ycbcr_image = convert the color image to Ycbcr image
Y = first layer of Ycbcr_image
Cb = second layer of Ycbcr_image
Cr = third layer of Ycbcr_image
create a zero matrix its rows and columns the height and the width of the inserted image.
find the position of nonzero pixels in the image that satisfy the condition (Cb>85 and Cb<135 and Cr>135 and Cr<180 and Y>80) and assign it to the raw, column and vector array
FOR i = 1 TO number of elements in raw array DO
change the value of that position in the created zero matrix by 1
END FOR

image_filled_ holes = fills the holes in the binary matrix
find the 4-connected components in the filled binary matrix that possibly contain face
}
End Function

Function =Extract Face and Nonface (Network, Image)
{
load the trained NNT
call Object Detection _Regions function
FOR m = 1 TO the total finding 4-connected components DO
crop the expected m^{th} face region from the gray image
resize the cropped image to 64×64
apply histogram equalization function to return enhanced image
convert the enhanced image to a vector
simulate the network that had been trained on the vector image
IF the output from the network > 0.5 THEN
the cropped image is face
ELSEIF the output from the network < -0.5 THEN
the cropped image is non- face
ELSE
I do not know
END IF
END FOR
Draw a red square for every cropped image that result in face.
}
End Function

Neural Networks Phase

The second phase is used to design, train and test the NNT system. The NNT is divided into two groups; the Multilayer Perceptrons (MLPs) and Probabilistic Neural Networks (PNNs) (Kukharev and Novosielski, 2004). The PNNs require a large amount of memory, because all the training data must be stored in the pattern layer. This is the main disadvantage of PNNs and the main reason for chosen MLPs rather than the other types of PNNs.

Figure 3 shows the architecture of the MLPs network employed for modeling a nonlinear system. This architecture consists of a set of layers. They are input, hidden and output layers. The input layer contains N neurons. Each hidden layer has specified number of neurons. The output layer has one neuron. For each layer, the weights and the biases are specified. The activation function s is used for the region recognition (face or non-face).

The pattern data is constructed as a database set to train and test the NNT. This database contains images of faces and non-faces. The database contains two sets: The first has 740 images of 10 subjects. Each subject has 74 images, where 37 images were taken every 5 degree from right profile (defined as +90°) to left profile (defined as -90°) in the pan rotation. The remaining 37 images are generated (synthesized) by the existing 37 images using commercial image processing software in the way of flipping them horizontally (http://robotics.csie.ncku.edu.tw/Databases/FaceDetect_PoseEstimate.htm#Our_Database_). The second set has 14 images for each of 50 individuals, a total of 700 images. All images are colorful and taken against a white homogenous background in an upright frontal position with profile rotation of up to about 180 degrees. Scale might vary about 10% and the original size of each image is 640×480 pixels (http://fei.edu.br/~cet/facedatabase.html). The database is divided into two parts, two-thirds for training and one-third for test.

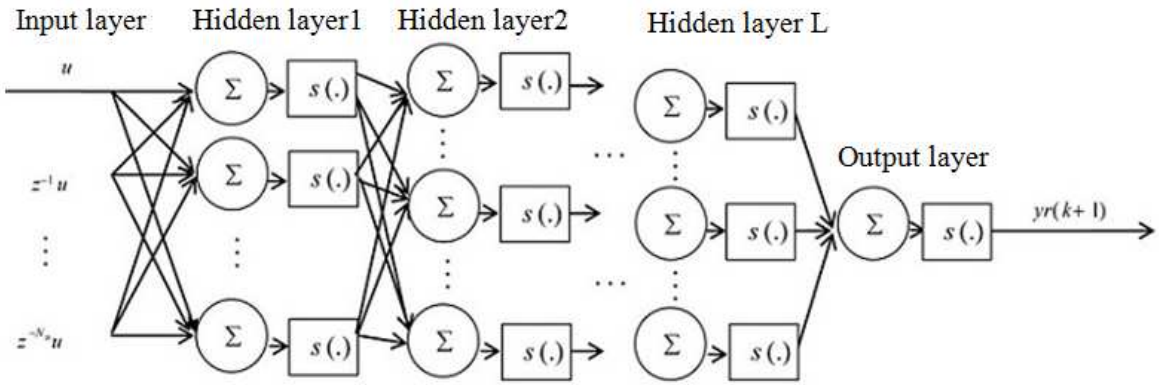

Fig. 3. Multilayer perceptron feed for ward neural network

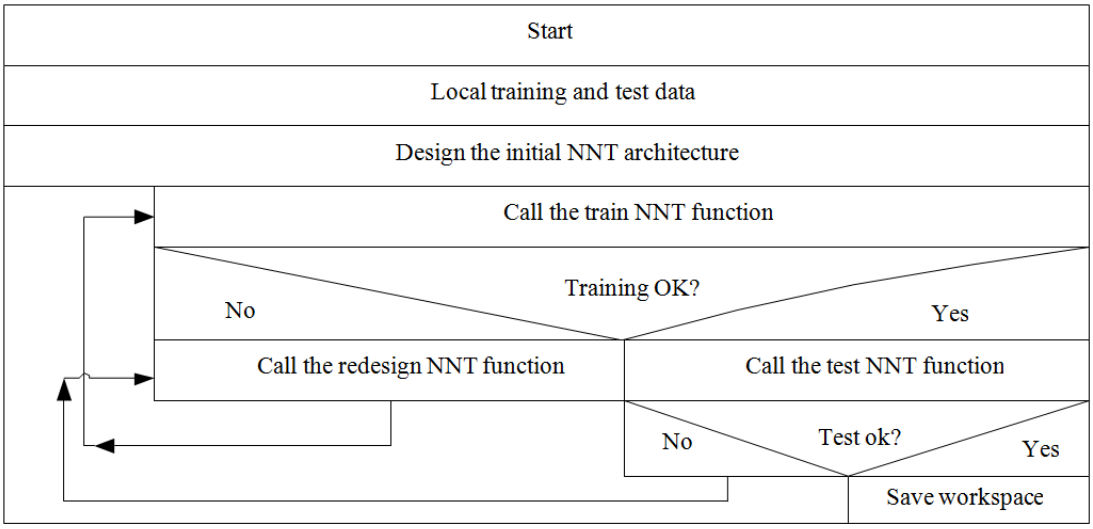

Fig. 4. Training process module

Training Process

The training is an important process to decide whether this object is a face or non-face. The proposed system is trained. In the case of the failure of the training process, redesign the architecture. Then retraining the NNT again and so on until the training process completed successfully. After that the system is tested on test data. In the case of system failure in the test, redesign the architecture of NNT, then retraining the NNT again and so on until get the best trained network. Training process has five basic functions. They are load training and test data, initial NNT architecture, training, test and redesign NNT functions; Fig. 4. These training functions are presented in the following five subsections.

Load Training and Test Data Function

This function is used to generate both input array which consist of images to be used for training the NNT and the desired output array. The loaded database (face and non-face) will be stored in cell array. The function is presented as follows:

```
Function = Load_train_test_data ()
{
Get Face folder path
Get Non face folder path
IF Image Database Exist
    Load the Image Database;
Else
   Calculate number of images in face folder
  FOR i = 1 TO Number of face images DO
      Read Image
     Convert image to vector
     Save image in face image database
  END FOR
  Calculate number of images in non-face folder
  FOR m = 1 TO Number of non-face images DO
    Read Image
    Convert image to vector
    Save image in non-face image database
  END FOR
  Save image database
END IF
Calculate total_number_of_images
FOR  j = 1 TO Number of face images DO
     Store images from face database to input data array
    Store 1 in desired output array;
END FOR
FOR n = Number of face images + 1 TO Number of total
images DO
Store images from face database to input data array
Store -1 in desired output array
END FOR
Return (input Data array AND Desired Output array);
}
End Function
```

The Initial NNT architecture Function

This function is used to design the initial architecture for NNT. Its details are shown as follows:

```
Function = Init_NNT_Archit()
{
  1- Create a list of training functions names.
  2- Create a list of activation functions names.
  3- Specify the following components of NNs:
      - number of hidden layers
      - number of neurons for each layer
      - number of epochs
      - training function name.
```

- activating function name.
- experment_counter = 0.
4- Build NN architecture.
5- Initialize weight values randomly.
}

The Train NNT Function

This function is used to train NNT and calculate the performance. It is described in the following:

```
Function=Train_NNT()
{
   Call Load_train_test_data ()
   Call Init_NNT_Archit()
  Train the NNT
  Calculate network performance for each output
  Calculate performance percentage
  If the performance is greater than 0.95
  Save the workspace
  Call Test_NNT_Archit();
}
End Function
```

The Test NNT Function

This function is used to test NNT and calculate the performance. It is given in the following:

```
Function=Test_NNT()
{
    Call Load_train_test_data ()
    Load the workspace
   Test the NNT
  Calculate network performance for each output
  Calculate performance percentage
  If the performance is less than 0.95
  Call Redesign_NNT_Archit();
}
End Function
```

The Redesign NNT Architecture Function

In case if the required performance is not obtained, this function is applied to redesign the architecture of the NNT. The function is illustrated in the following:

{
1- Create a list of training functions names.
2- Create a list of activation functions names.
3- Change the following variable as a linear combination:
- number of hidden layers
- number of neurons for each layer
- number of epochs
- training function name from the list.
- activating function name from the list.
4- Build NN architecture.
5- Initialize weight values randomly.
}

The Applied System

The proposed system was designed in details in the section 2. This section presents more applications for this system. These applications contain a set of processes. The first process for this system is started by training the NNT. It is called *Train_NNT()*. After that, the next process is applied to test the trained network. This process is called *Test_NNT()*. Finally, the best trained network is assigned. These processes are presented in details in the following three subsections.

Training Process

This process is done by training the NNT using the above mentioned database for face and non-face as an input. The process is designed to work in automatic way without any help from the user. Eleven training functions are applied to find the best one. These functions are *trainrp, traincgf, traingdx, traingdm, traingd, trainoss, trainscg, traincgp, traincgb, trainlm and trainbr*. Different NNT structures and training parameter's values are chosen to get the best training network.

The process is firstly started with building initial NNT architecture without hidden layer by selecting the first training and activation functions from list. Then, the initial number of neurons and epochs are specified. The weighted values are initialized randomly. It was noticed that the best three training functions are *traincgf, trainoss and traincgb* and their performances are 80, 99.8 and 100% respectively. Moreover, the obtained best networks of these functions are reached at 2965, 645 and 1000 epochs using three hidden layers. The number of neurons for the three functions are (33, 11, 1), (55, 12, 2) and (10, 5, 1) respectively.

During the applying of training functions, some problems were noticed, such as with *trainlm* and *trainbr* functions that raised an out of memory error. Also *traingd* function did not perform well, it needed small number of epochs and did not give a good training initial performance percentage and for large epochs number no training was performed with this function. After that, the training process is completed and the test process is started as illustrated in the next section.

Test Process

This process is applied to check the performance of the trained networks using the test data. If the required performance is reached the best network is obtained.

Table 1. The training functions performances

	Train-										
Training function	*cgb*	*oss*	*scg*	*gd*	*cgf*	*gdx*	*gdm*	*rp*	*cgp*	*br*	*lm*
Performance (%)	100	99.8	90	90	90	70	60.6	60.4	60	Out of memory	

Otherwise, this experiment is repeated again for ten times using the same architecture of the NNT hoping to get a random weighted values lead to improved performance. In the case if the required performance is not reached, the process rebuilds the architecture of the NNT according to a linear combination among the number of epochs, the number of neurons, training functions, activation functions and the number of hidden layers.

During training and test processes, the processing time was about one minute. The most networks performed well with a small number of hidden layers and on initial performance measurement number of networks scoured a high performance percentage up to 100% with respect to input images simulation. The performance differs greatly depending on the training function, NNT structure and parameters. If the performance of the trained networks is less than 95%, they are discarded. Performance percentage (initial measurement) was calculated with respect to same database as in the input data for every network and for some NNTs it was excellent results.

The Best Trained Network

The proposed system is applied using 11 training functions as mentioned above to have a meaningful comparison. After applying the training and test processes, 11 best trained networks are obtained one for each training function. The obtained performances for them are shown in Table 1. The best training function is *traincgb*. Its trained network is the best and is chosen as best classifier for face detection. Its structure consists of 4096 neurons in the input layer, 1 neuron in the output layer and three hidden layers (10, 5, 1) neurons. The training parameters are 1000 epochs, 1e-6 error rate and 0.5 learning rate.

Results and Discussion

The best NNT is used as a whole application to detect the faces from input images. The tested database consists of three sets. The first contains 550 images (MWCIT, 2003). The proposed system is applied on 10 images for 1 face, 22 images for 2 faces, 39 images for 3 faces, 45 images for 4 faces, 40 images for 5 and 6 faces and 40 images for 7 faces. The second contains 20 images above 15 faces (https://goo.gl/qSmJOU). The last contains 20 images with 11 faces that have African faces. This database contains images of faces and non-faces (https://goo.gl/knpygn). Some of results with special cases are presented in order to show the performance of the system.

Figure 5 shows the results of the proposed system with an image of four people, one of them wears glasses. In the first stage, four big white areas and others small; as given in Fig. 5b. Figure 5c presents the output of the second stage by removing the small white areas and enhancement the big ones. The next stage is used to identify these areas as objects according to their coordinates of the big white ones; Fig. 5d. In the last stage, the four faces are detected correctly with 100% as shown in Fig. 5e. It is noticed that the four faces are detected correctly and the result is not affected by the person wearing a glasses.

Figure 6 shows the results of the proposed system with an image of seven people, three women and four men two of them wearing T-shirts. So, their hands color look seems as the faces. The faces and the hands are determined as white color as shown in Fig. 6b. Then, the separated hands are removed and the white objects are enhanced; Fig. 6c. After that the objects are identified according to their coordinates of the big white holes; Fig. 6d. Finally, the NNT are recognized the seven faces correctly and discarded the connected hands as shown in Fig. 6e.

Figure 7a shows an image containing a lot of number of people. The result of the first stage contains small and big white objects as given in Fig. 7b. The white big objects are enhanced, while the other small are removed as shown in Fig. 7c. Then, the enhanced objects are identified according to their coordinates of the big white holes; Fig. 7d. After that, the NNT are recognized the most faces correctly with accuracy 88% as shown in Fig. 7e.

Figure 8a contains an image of eleven African players. Their faces' colors change from black to white. The result of the first stage contains small and big white objects as shown in Fig. 8b. The white big objects are improved, while the other small are removed as shown in Fig. 8c. Then, the determined objects are identified according to their coordinates of the big white objects; Fig. 8d. In the last stage, it noted that seven faces are only detected correctly with 67.6% as shown in Fig. 8e. However, the objects of very black faces are not determined correctly due to their skin were very black consequently they are not detected as white objects. Due to this problem, the most four black faces are missed.

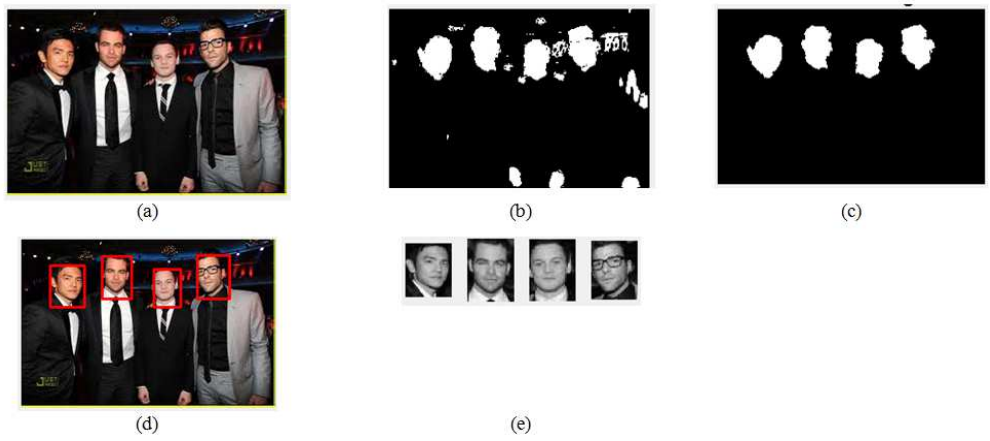

Fig. 5. The first result, (a)The original input image, (b) Binary image after skin color segmentation using new values, (c) The connected components in the image, (d) Face detected output image , (e) Recognized faces by NNT

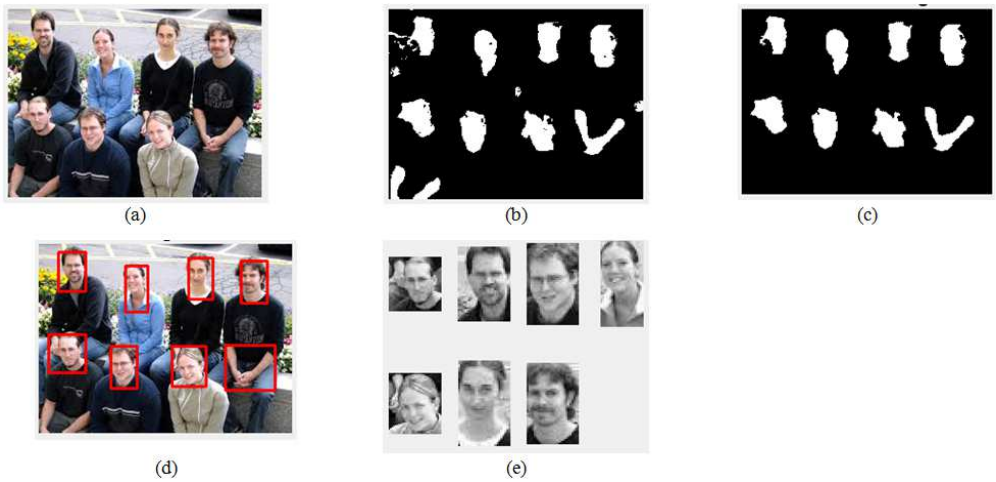

Fig. 6. The second results, (a) The original input image, (b) Binary image after skin color segmentation using new values, (c) The connected components in the image, (d) face detected output image, (e) Recognized faces by NNT

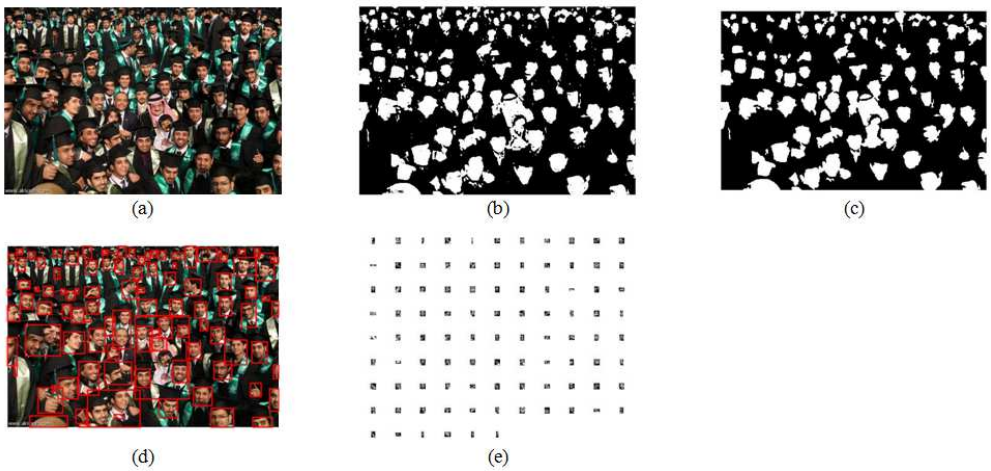

Fig. 7. The fourth results, (a) The original input image, (b) Binary image after skin color segmentation using new values, (c), The connected components in the image, (d) Face detected output image, (e) Recognized faces by NNT

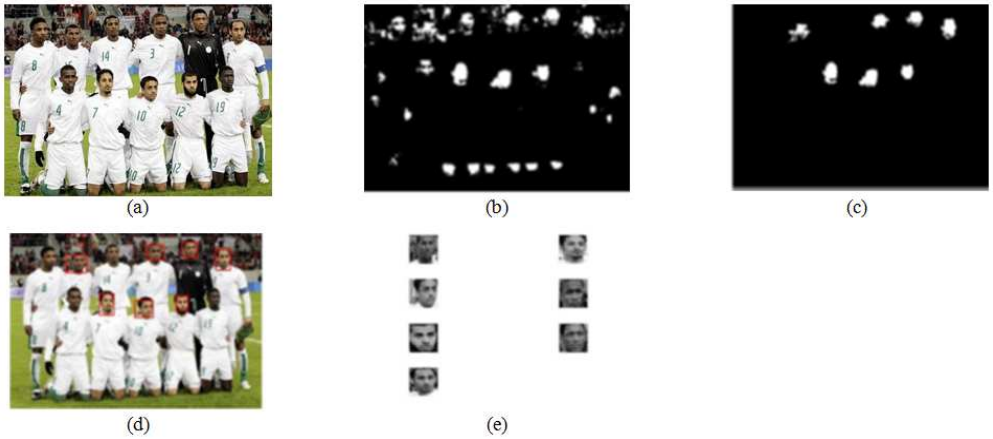

Fig. 8. Fourth exponential image, (a) The original input image, (b) Binary image after skin color segmentation using new values, (c) The connected components in the image, (d) Face detected output image, (e) Recognized faces by NNT

Table 2. Comparison with (Tiwari and Ahmed, 2013)

No. of faces	1	2	3	4	5	6	7
(Tiwari and Ahmed, 2013) (%)	100	100	94.9	93.8	93.3	---	96.9
Prop. System (%)	100	100	100	100	100	100	100

Table 3. Comparison with a set of systems

Detection methods/systems	Performance (%)
SVM (Waring and Liu, 2005; Jee *et al*., 2004)	86.7
Data Mining Method (Tsao *et al*., 2010)	87.5
Statistic of skin-color (Chen *et al*., 2013)	89.0
2DPCA (Jizeng and Hongmei, 2008)	89.7
NNT (Chen *et al*., 2010)	
Improved Adaboost (Ma and Du, 2010)	91.3
NNT (Lin, 2007)	92.2
Geometry constraints and Face-mask (Yu and Zhu, 2013)	92.8
Local binary patterns (Timo *et al*., 2006; 2004)	94.1
Facial geometry (Tiwari and Ahmed, 2013)	94.9
The Proposed system	95.1

In order to evaluate the performance of the proposed system, it is compared with other systems. From the comparison with (Tiwari and Ahmed, 2013), it is noticed that (Tiwari and Ahmed, 2013) get the highest performance (100%) when the images have up to two people while our system get this performance from images have up to 7 people as shown in Table 2. Thus the performance of our system is better than the system in (Tiwari and Ahmed, 2013). Other systems for face detection were only calculated their performance according to the average. The proposed system is compared with them, our system get the best performance; Table 3. From Table 3, some researchers are used NNT however our performance using the NNT outperform these methods such as the ones in (Lin, 2007; Chen *et al*., 2010).

Conclusion

Face detection is an important prior step for face recognition system which is widely used in many applications. The proposed system has good accuracy and low running time for face detection. The system is divided into two main approaches. The first approach is used for the image preprocessing based on skin color, region props and bounding box to create white objects which are expected to be faces. The second approach is applied to identify the severed parts of the objects by the first approach using NNT. The NNT is used to make decision about these objects, they are faces or not.

The NNT is designed and adopted to work as powerful detection for faces. The Experimental results show that the NNT works efficiently under various kinds of environments. The system gets the highest performance for detecting faces from non-dense images, acceptable performance for very dense image and low performance with images which have African faces. During the comparison with other systems, it is found that the proposed system gets the best performance.

In the future the system will be develop and enhanced to detect faces correctly from images contain African

people or very dense. In order to overcome the first problem of missing the very black faces, combine another features with the skin colors.

Acknowledgement

The author would like to thank the anonymous AJAS reviewers for their insightful comments. I would like also thank Dr. A. Elharby for his help.

Ethics

This article is original and contains unpublished material.

References

Al Haj, M., A. Amato, X. Roca and J. Gonz`alez, 2007. Face detection in color images using primitive shape features. Comput. Recognit. Syst., 2: 179-186.

Ayachi, E., S. Ihsen and B. Mohamed, 2012. A comparative study of nonlinear time-varying process modeling techniques: Application to chemical reactor. J. Intell. Learn. Syst. Applic., 4: 20-28. DOI: 10.4236/jilsa.2012.41002

Ban, Y., S.K. Kim, S. Kim, K.A. Toh and S. Lee, 2014. Face detection based on skin color likelihood. Patt. Recognit., 47: 1573-1585. DOI: 10.1016/j.patcog. 2013.11.005

Chen, A.P., L. Pan and Y.B. Tong, 2010. Face detection technology based on skin color segmentation and template matching. Proceedings of the 2nd International Workshop on Education Technology and Computer Science, Mar. 6-7, IEEE Xplore Press, Wuhan, pp: 708-711. DOI: 10.1109/ETCS.2010.465

Chen, T.W., C.S. Tang and S.Y. Chien, 2008. Highly efficient face detection in color images. Adv. Multimedia Inform. Process., 5353: 919-922. DOI: 10.1007/978-3-540-89796-5_114

Chen, Z.X., C.Y. Liu, F.L. Chang and X.Z. Han, 2013. Fast face detection algorithm based on improved skin-color model. Arab J. Sci. Eng., 38: 629-635. DOI: 10.1007/s13369-012-0376-1

Gondaliya, D.S., P.P. Kamothi, V.N. Fudnawala, K.P. Patel and H.S. Patel *et al*., 2015. Review on human face detection based on skin color and edge information. Int. J. Eng. Sci. Technol., 7: 12-20.

Goswami, J.P., P.K. Chourasiya and N.S. Chauhan, 2013. Face detection using color based segmentation and edge detection. Int. J. Comput. Applic., 72: 49-54. DOI: 10.5120/12582-9328

http://fei.edu.br/~cet/facedatabase.html

http://robotics.csie.ncku.edu.tw/Databases/FaceDetect_PoseEstimate.htm#Our_Database_

https://goo.gl/qSmJOU

https://goo.gl/knpygn

Ikeda, O., 2003. Segmentation of faces in video footage using HSV color for face detection and image retrieval. Proceedings of the International Conference on Image Processing, Sept. 14-17, IEEE Xplore Press, pp: 913-916. DOI: 10.1109/ICIP.2003.1247394

Jee, H., K. Lee and S. Pan, 2004. Eye and face detection using SVM. Proceedings of the Intelligent Sensors, Sensor Networks and Information Processing Conference, Dec. 14-17, IEEE Xplore Press, pp: 577-580. DOI: 10.1109/ISSNIP.2004.1417525

Jizeng, W. and Y. Hongmei, 2008. Face detection based on template matching and 2DPCA algorithm. Proceedings of the Congress on Image and Signal Processing, May 27-30, IEEE Xplore Press, Sanya, China, pp: 575-579. DOI: 10.1109/CISP.2008.270

Juang, C.F. and S.J. Shiu, 2008. Using self-organizing fuzzy network with support vector learning for face detection in color images, Neurocomputing, 71: 3409-3420.

Kalbkhani, H., M.G. Shayesteh and S.M. Mousavi, 2013. Efficient algorithms for detection of face, eye and eye state. IET Comput. Visio, 7: 184-200. DOI: 10.1049/iet-cvi.2011.0091

Kukharev, G. and A. Novosielski, 2004. Visitor identification-elaborating real time face recognition system. Proceedings of the 12th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision, (VCV' 04), Czech Republic, pp: 157-164.

Li, Y., H. Yin and X. Gong, 2013. A beach image detection method based on erotic image detection. Applied Mechan. Mater., 278-280: 1282-1286. DOI: 10.4028/www.scientific.net/AMM.278-280.1282

Lin, C., 2007. Face detection in complicated backgrounds and different illumination conditions by using YCbCr color space and neural network. Patt. Recognit. Lett., 28: 2190-2200. DOI: 10.1016/j.patrec.2007.07.003

Liu, X., T. Chen and B.V. Kumar, 2003. Face authentication for multiple subjects using eigenflow. Patt. Recognit., 36: 313-328.

Ma, S.Y. and T.C. Du, 2010. Improved adaboost face detection. Proceedings of the International Conference on Measuring Technology and Mechatronics Automation, Mar. 13-14, IEEE Xplore Press, Changsha City, pp: 434-437. DOI: 10.1109/ICMTMA.2010.184

MWCIT, 2003. Background image dataset. Markus Weber at California Institute of Technology.

Nasrollahi, K., M. Rahmati and T.B. Moeslund, 2008. A NNT based cascaded classifier for face detection in color images with complex background.

Osman, G., M.S. Hitam and M.N. Ismail, 2012. enhanced skin colour classifier using RGB ratio model. Int. J. Soft Compu., 3: 1-14.

Patilkulkarni, S. and H.C.V. Lakshmi, 2013. Vanishing moments of a wavelet system and feature set in face detection problem for color images. Int. J. Comput. Applic., 66: 36-42. DOI: 10.5120/11171-6369

Perumal, K. and N.S. Perumal, 2013. Image enhancement for face recognition using color segmentation and edge detection algorithm. Int. J. Comput. Sci. Commun. Netw., 3: 187-193.

Rewar, E. and S.K. Lenka, 2013. Comparative analysis of skin color based models for face detection. Signal Image Process. Int. J., 4: 69-75. DOI: 10.5121/sipij.2013.4206

Siddiqui, K.T.A. and A. Wasif, 2015. Skin detection of animation characters. Int. J. Soft Comput.

Sultana, S., M.S. Islam and M.G. Moazzam, 2014. Face detection using fuzzy skin color segmentation. Int. J. Comput. Applic., 85: 29-34.

Timo, A., H. Abdenour and P. Matti, 2004. Face recognition with local binary patterns. Proceedings of the 8th European Conference on Computer Vision, May 11-14, Prague, Czech Republic, pp: 469-481. DOI: 10.1007/978-3-540-24670-1_36

Timo, A., H. Abdenour and P. Matti, 2006. Face description with local binary patterns: Application to face recognition. IEEE Trans. Patt. Anal. Mach. Intell., 28: 2037-2041. DOI: 10.1109/TPAMI.2006.244

Tiwari, N. and M. Ahmed, 2013. Detection of facial features of frontal face color images. Int. J. Adv. Comput. Res., 3: 54-58.

Tsao, W.K., A.J.T. Lee, Y.H. Liu, T.W. Chang and H.H. Lin, 2010. A data mining approach to face detection. Patt. Recognit., 43: 1039-1049. DOI: 10.1016/j.patcog.2009.09.005

Waring, C.A. and X. Liu, 2005. Face detection using spectral histograms and SVMs. IEEE Trans. Syst. Man Cybernet. Part B., 35: 467-476. DOI: 10.1109/TSMCB.2005.846655

Yu, F. and N. Zhu, 2013. Algorithm for face detection combining geometry constraints and face-mask. Inform. J., 12: 1406-1411.

Zali-Vargahan, B., H. Kalbkhani and M.G. Shayesteh, 2013. An efficient algorithm for lip detection in color face images. Proceedings of the 21st Iranian Conference on Electrical Engineering, May 14-16, IEEE Xplore Press, Mashhad, 1-4. DOI: 10.1109/IranianCEE.2013.6599705.