Triangle-based approach to the detection of human face

Chiunhsiun Lin, Kuo-Chin Fan*

Institute of Computer Science and Information Engineering, National Central University, Chung-Li, Taiwan 32054, ROC

Received 17 August 1999; received in revised form 9 March 2000; accepted 27 March 2000

Abstract

A robust and efficient human face detection system that can detect multiple faces in complex backgrounds is presented in this paper. The proposed system consists of two primary parts. The first part is to search for the potential face regions. The second part is to perform face verification. Our system can conquer different sizes, different lighting conditions, varying pose and expression, and noise and defocus problems. In addition to overcoming the problem of partial occlusion of mouth and sunglasses, the system can also detect faces from the side view. Experimental results demonstrate that an approximately 98% success rate is achieved. The experimental results reveal that the proposed method is better than traditional methods in terms of efficiency and accuracy. © 2001 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Face detection; Triangle-based segmentation; Weighting mask function

1. Introduction

Automatic recognition of human faces is one of the most difficult and important problems in the areas of pattern recognition and computer vision. It can be used as a security mechanism to replace metal key, plastic card, and password or PIN number. It can also be used in criminology to find out possible criminals. However, most face recognition systems require the input face to be free of background and the size to be nearly unchanged. These constraints greatly hinder the usefulness of the system. As we know, a successful face detection process is the prerequisite to facilitate later face recognition task. If we do not have a successful face detection method, we cannot develop a successful face recognition system. Therefore, we should not treat face detection process merely as a preprocessing of a face recognition system. On the other hand, we should deal with face detection problem with the same importance as the face recognition problem. Speedy and correct detection of faces in images is a goal to be pursued in any face detection system. Nevertheless, most of the previous face detection researches impose many restrictions, such as they do not allow varying pose, expression, and noise and defocus problems. In order to remove these restrictions, we develop a robust system that is suitable for real-time face detection.

Several researches have been conducted on human face detection. Some successful systems have been developed and reported in the literature, such as human face recognition [1-10], and human face detection [11-20]. Govindaraju et al. [11,12] presented a system which could search human faces in photographs of newspapers, but the approximate size and expected number of faces must be known in advance. Viennet [13] utilized a multi-resolution image analysis technique followed by a scanning of smoothed images using a time delay neural network architecture (TDNN) to detect faces in real-life home scenes. Yang et al. [14] utilized a three-level hierarchical knowledge-based method to locate human faces in complex backgrounds. In their method, the higher two levels are implemented based on mosaic images at different resolutions, whereas an improved edge detection
method is employed in the lower level. Sung et al. [15] utilized two distance metrics to measure the distance between the input image and the cluster center. Twelve clusters including six faces and six non-face clusters were trained by a modified k-mean clustering technique. A feature vector consisting of 12 values was inputted to a multi-layer perceptron network for the verification task. Leung et al. [16] coupled a set of local feature detectors via a statistical model to find the facial components for face location. Rowley et al. [17] presented a neural network-based face detection system. In their work, a retinal connected neural network was employed to check the small windows of an image, and judge whether each window contains a face. They adopted a small window (20 × 20) to slide over all portions of an image at various scales, and used an oval mask for ignoring background pixels. Their system arbitrates between multiple networks to improve the performance over a single network. However, the inefficient search is a time-consuming procedure. Moreover, it cannot detect the partially occluded faces. Lee et al. [18] extracted the face part from the homogeneous background by tracking the face boundary. In their work, they assume that the face part is located in the center of a captured image. Then they use knowledge-based feature extraction and neuro-fuzzy algorithm to carry out the face recognition task. However, they can only detect the front faces. Juell et al. [19] proposed a hierarchical neural network to detect human faces in gray-scale images. An edge-enhancing preprocessor and four backpropagation neural networks arranged in a hierarchical structure were devised to find multiple faces in the scene. Their approach was invariant with respect to translation, rotation, and scale. Han et al. [20] proposed a system using a morphology-based technique to perform eye-analog segmentation. Then, a trained backpropagation neural network performs the face verification task. Instead of gray-scale images, Sobottka et al. [21] located the poses of human faces and facial features from color images. They presented a system that used an ellipse in Hue-saturation value (HSV) color space to express the shape of a face. Wu et al. [22] proposed a skin color distribution function on an unchanged color space to detect the face-like region. The skin color regions in color images were modeled as several 2-D patterns and verified with the built face model by a fuzzy pattern matching approach. Recently, Dai et al. [23] presented a texture model incorporating color information utilization. Their system is based on the space gray-level dependence (SGLD) matrix. The facial texture model composed by a set of condition inequalities was derived on an experimental basis. Moreover, they converted the RGB color space to YIQ, then used the I component to highlight the skin color regions to improve the performances. However, they used a window to scan the full picture, and calculated the average intensity to decide whether the window contains a facial region or not. The wasteful search is a time-consuming procedure. Moreover, it cannot detect the partially occluded faces, wearing sunglasses, and faces in side view.

Most of the aforementioned methods limit themselves to dealing with human faces in frontal view. There are some drawbacks in these approaches. (1) They cannot detect a face which is smaller than 50 * 50 pixels. (2) They cannot detect multiple faces (more than 3 faces) in complex backgrounds. (3) They cannot handle the defocus and noise problems. (4) They cannot conquer the problem of partial occlusion of mouth or wearing sunglasses. (5) They cannot cope with the problem of detection of faces in side view. Although there are some researches that can solve two or three problems as pointed out above, there is still no system that can solve all the aforementioned problems.

In this paper, a robust and efficient human face detection system is proposed to solve all the aforementioned problems. The designed system is composed of two principal parts as shown in Fig. 1. The first part of the designed system is to search the potential face regions. The second part is to perform the task of face verification for each potential face region. There are four steps in the first part of the designed system. Firstly, read in an image, then convert this input image to a binary image. Secondly, label all 4-connected components in the image to form blocks and find out the center of each block. Thirdly, detect any 3 centers of 3 different blocks that form an isosceles triangle (frontal view) or a right triangle (side view). Fourthly, clip the blocks that satisfy the triangle criteria as the potential face region. The second part of the designed system is to perform the task of face verification. We propose an efficient weighting mask function that is applied to decide whether a potential face region contains a face. There are three steps in this part. The first step is to normalize the size of all potential facial regions. The second step is to feed every normalized potential facial region into the weighting mask function and calculate the weight. The third step is to perform the verification task by thresholding the weight obtained in the previous step.

The proposed face detection system knows how to locate multiple faces oriented in complicated background automatically. Furthermore, it can handle different sizes, dissimilar lighting conditions, varying pose and expression, and noise and defocus problems. In addition to coping with the problem of partial occlusion of mouth and sunglasses, the system can also detect faces presented in side view.

1 Refs. [17,23] can handle (1) and (2). However, it is too slow because the inefficient search is really a time-consuming procedure.
The rest of the paper is organized as follows. In Section 2, segmentation of the potential face regions based on some geometric rules is described. In Section 3, each of the normalized potential face regions is fed to the weighting mask function to verify whether the potential face region really contains a face. Experimental results are demonstrated in Section 4 to verify the validity of the proposed face detection system. Finally, conclusions are given in Section 5.

2. Segmentation of potential face regions

The main purpose of this process is to find the regions in an input image that might potentially contain faces. There are four steps in the first part of the designed system. Firstly, read in an image, it could be either a color or gray-level image. Then, convert this input image to a binary image. For example, when we read in an RGB color image, we will preprocess the input image to a gray-level image by eliminating the hue and saturation information while retaining the luminance. Then, binarize the gray-level image to a "binary image" by simple global thresholding with threshold $T$ because the objects of interest in our case are darker than the background. Pixels with gray level $\leq T$ are labeled black, and any pixel with gray level $> T$ is labeled white. Hence, the output binary image has values 1 (black) for all pixels in the input image with luminance less than threshold $T$ and 0 (white) for all other pixels. Before proceeding to the next step, we perform the opening operation (erosion first, then dilation) to remove noise, and then the closing operation (dilation first, then erosion) to eliminate holes. The details of opening and closing operations can be found in Ref. [24]. In our work, we assume that the input image would not be too dark or too light to obtain a satisfactory binarization result.

2.1. Preprocess the inputted image to a binary image

The input image could be a color or gray-scale image. In this step, we first convert the inputted image to a binary image. For example, when we read in an RGB color image, we will preprocess the input image to a gray-level image by eliminating the hue and saturation information while retaining the luminance. Then, binarize the gray-level image to a "binary image" by simple global thresholding with threshold $T$ because the objects of interest in our case are darker than the background. Pixels with gray level $\leq T$ are labeled black, and any pixel with gray level $> T$ is labeled white. Hence, the output binary image has values 1 (black) for all pixels in the input image with luminance less than threshold $T$ and 0 (white) for all other pixels. Before proceeding to the next step, we perform the opening operation (erosion first, then dilation) to remove noise, and then the closing operation (dilation first, then erosion) to eliminate holes. The details of opening and closing operations can be found in Ref. [24]. In our work, we assume that the input image would not be too dark or too light to obtain a satisfactory binarization result.

2.2. Label all 4-connected components and find the center of each block

After binarization, the next task is to get 4-connected components, label them, and then find the center of each
block. The details of connected component finding can be found in Ref. [24].

2.3. Find any 3 centers of 3 different blocks that form an isosceles triangle (frontal view) or a right triangle (side view)

From careful observation, we discover that two eyes and one mouth in the frontal view will form an isosceles triangle, and a right triangle will be formed by one eye, one mouth, and one ear hole in the side view. This is the rationale on which the triangle, and a right triangle will be formed by one eye, one mouth in the frontal view will form an isosceles triangle, obtained from the criteria of based. We could search the potential face regions that are obtained from the criteria of “the combination of two eyes and one mouth (isosceles triangle)” and “the combination of one eye, one ear hole, and one mouth (right triangle)”.

2.3.1. The matching rules for finding an isosceles triangle

If the triangle $ijk$ is an isosceles triangle as shown in Fig. 2(a), then it should possess the characteristic of “the distance of line $ij$ = the distance of line $jk$”. From observation, we discover that the Euclidean distance between two eyes (line $ik$) is about 90–110% of the Euclidean distance between the center of the right/left eye and the mouth. Due to the imaging effect and imperfect binarization result, a 25% deviation is given to absorb the tolerance. The first matching rule can thereby be stated as $(\text{abs}(D(i,j) - D(j,k)) < 0.25 \max (D(i,j), D(j,k)))$, and the second matching rule is $(\text{abs}(D(i,j) - D(i,k)) < 0.25 \max (D(i,j), D(j,k)))$. Since the labeling process is operated from left to right then from top to bottom, we can get the third matching rule as “$i < j < k$”. Here, “abs” means the absolute value, “$D(i,j)$” denotes the Euclidean distance between the centers of block $i$ (right eye) and block $j$ (mouth), “$D(j,k)$” denotes the Euclidean distance between the centers of block $k$ (left eye) and block $j$ (mouth). “$D(i,k)$” represents the Euclidean distance between the centers of block $i$ (right eye) and block $k$ (left eye). For example, as shown in Fig. 2(b), if three points $(i, j, k)$ satisfy the matching rules, then we think that they form an isosceles triangle.

Fig. 2. (a) The isosceles triangle $ijk$. (b) Three points $(i, j, k)$ satisfy the matching rules, which will form an isosceles triangle.

After we have found the isosceles triangle, it is easy to get the coordinates of the four corner points that form the potential facial region. Since we think the real facial region should cover the eyebrows, two eyes, mouth and some area below the mouth [9], the coordinates can be calculated as follows: Assume that $(Xi, Yi), (Xj, Yj)$ and $(Xk, Yk)$ are the three center points of blocks $i, j,$ and $k$, that form an isosceles triangle. $(X1, Y1), (X2, Y2), (X3, Y3),$ and $(X4, Y4)$ are the four corner points of the face region as shown in Fig. 3. $X1$ and $X4$ locate at the same coordinate of $(Xi - 1/3D(i,k))$; $X2$ and $X3$ locate at the same coordinate of $(Xk + 1/3D(i,k))$; $Y1$ and $Y2$ locate at the same coordinate of $(Yi + 1/3D(i,k))$; $Y3$ and $Y4$ locate at the same coordinate of $(Yj - 1/3D(i,k))$; where $D(i,k)$ is the Euclidean distance between the centers of block $i$ (right eye) and block $k$ (left eye).

\[
\begin{align*}
X1 &= X4 = Xi - 1/3D(i,k), \\
X2 &= X3 = Xk + 1/3D(i,k), \\
Y1 &= Y2 = Yi + 1/3D(i,k), \\
Y3 &= Y4 = Yj - 1/3D(i,k).
\end{align*}
\]

Fig. 3. Assume that $(Xi, Yi), (Xj, Yj)$ and $(Xk, Yk)$ are the three center points of blocks $i, j,$ and $k$, respectively. The four corner points of the face region will be $(X1, Y1), (X2, Y2), (X3, Y3),$ and $(X4, Y4)$.
2.3.2. The matching rules for finding a right triangle (the right/left side view)

If the triangle $ijk$ is a right triangle ($30^\circ, 60^\circ, 90^\circ$) as shown in Fig. 4, then it should possess the characteristic of “the Euclidean distance of line $ik = 2$ times of the Euclidean distance of line $jk$”, and “the Euclidean distance of line $ij = 1.732$ times of the Euclidean distance of line $jk$”.

The 4 rules to get the right triangle for right side view: As mentioned previously, due to the imaging effect and imperfect binarization result, a 25% deviation is also given to tolerate the deviation. Therefore, the first matching rule is $\text{abs}(D(i, j) - D(j, k)) < 0.60D(i, k)$ and $\text{abs}(D(i, k) - D(j, k)) > 0.40D(i, k)$, the second matching rule is $\text{abs}(D(i, k) - D(i, j)) > 0.13D(i, k)$ and $\text{abs}(D(i, k) - D(i, j)) < 0.19D(i, k)$, and the third matching rule is $\text{abs}(D(i, j) - D(j, k)) < 0.44D(i, k)$ and $\text{abs}(D(i, j) - D(j, k)) > 0.29D(i, k)$). Since the labeling process is from left to right then from top to bottom, we can get the fourth matching rule as “$i < j < k$”. Here, “abs” means the absolute value, “$D(i, j)$” means the Euclidean distance between the centers of block $i$ (ear hole) and block $j$ (right eye), “$D(i, k)$” means the Euclidean distance between the centers of block $j$ (right eye) and block $k$ (mouth), and “$D(i, k)$” means the Euclidean distance between the centers of block $i$ (ear hole) and block $k$ (mouth).

For example, as shown in Fig. 5, if three points ($i, j, k$) satisfy the matching rules, then they form the right triangle ($30^\circ, 60^\circ, 90^\circ$) that we are looking for.

After we have found the triangle, it is easy to get the coordinates of the four corner points that form the potential facial region. Since we think the real side view region should cover one eye, one ear hole, mouth and some area below the mouth, the coordinates can be calculated as follows: Assume that ($Xi, Yi$), ($Xj, Yj$) and ($Xk, Yk$) are the three center points of blocks $i, j, k$ respectively. ($X1, Y1$), ($X2, Y2$), ($X3, Y3$), and ($X4, Y4$) are the four corner points of the side view region.

\[ X1 = X4 = Xi - 1.2D(i, j), \]
\[ X2 = X3 = Xi + 1.2D(i, j), \]
\[ Y1 = Y2 = Yi + 1.2D(i, j), \]
\[ Y3 = Y4 = Yi - 1.0D(i, j). \]

The 4 rules to get the right triangle for left side view: The definition and matching rules for finding a right triangle of the left side view are similar to those for the right side view as described previously. For example, as shown in Fig. 7, if three points ($i, j, k$) satisfy the 4 matching rules, then they could form the right triangle ($30^\circ, 60^\circ, 90^\circ$) that we are searching for.

As shown in Fig. 8, assume that ($Xi, Yi$), ($Xj, Yj$) and ($Xk, Yk$) are the three center points of 3 blocks $i, j, k$, respectively. ($X1, Y1$), ($X2, Y2$), ($X3, Y3$), and ($X4, Y4$) are the four corner points of the side view region.

Fig. 4. The right triangle $ijk$.

Fig. 5. Three points ($i, j, k$) form a right triangle ($30^\circ, 60^\circ, 90^\circ$) because they satisfy rules 1, 2, 3 and 4.

Fig. 6. Assume that ($Xi, Yi$), ($Xj, Yj$) and ($Xk, Yk$) are the three center points of blocks $i, j, k$, respectively. ($X1, Y1$), ($X2, Y2$), ($X3, Y3$), and ($X4, Y4$) are the four corner points of the side view region.
respectively. \((X1, Y1), (X2, Y2), (X3, Y3), \) and \((X4, Y4)\) are the four corner points of the side view region.

The coordinates of the four corner points as shown in Fig. 9 can be calculated as follows: \(X1\) and \(X4\) locate at the same coordinate of \((Xj - 1/6D(j, k))\), \(X2\) and \(X3\) locate at the same coordinate of \((Xj + 1.2D(j, k))\); \(Y1\) and \(Y2\) locate at the same coordinate of \((Yj + 1/4D(j, k))\), and \(Y3\) and \(Y4\) locate at the same coordinate of \((Yj - 1.0D(j, k))\) where \(|D(j, k)|\) means the Euclidean distance between the centers of block \(i\) (left eye) and block \(j\) (left eye):

\[
X1 = X4 = Xj - 1/6D(j, k), \quad (9) \\
X2 = X3 = Xj + 1.2D(j, k), \quad (10) \\
Y1 = Y2 = Yj + 1/4D(j, k), \quad (11) \\
Y3 = Y4 = Yj - 1.0D(j, k). \quad (12)
\]

2.3.3. The speedup of searching for an isosceles triangle or a right triangle

When we are looking for any three centers to form an isosceles triangle or a right triangle by the matching rules as mentioned previously. The first matching rule is \((\text{abs}(D(i, j) - D(j, k)) < 0.25 \max(D(i, j), D(j, k)))\), the second matching rule is \((\text{abs}(D(i, j) - D(i, k)) < 0.25 \max(D(i, j), D(j, k)))\), and the third matching rule is \(“i < j < k”\) (because the labeling process is operated from left to right then from top to bottom). If the Euclidean distance between the centers of block \(i\) (right eye) and block \(j\) (mouth) is already known, then block center \(k\) (left eye) should locate in the area of 75–125% of Euclidean distance between the centers of block \(i\) (right eye) and block \(j\) (mouth), which will form a circle. Furthermore, since the third matching rule is \(“i < j < k”\), the third block center \(k\) is only limited to the up-right part of the circle which is formed by rules 1 and 2. In other words, the search area is only limited in the dark area instead of the whole area of the image as shown in Fig. 9. As a result, it is not really a selection of \(C_{ij}\)(select any 3 combinations from \(n\) blocks). In this way, the triangle-based segmentation process can reduce the background part of a cluttered image up to 97%. This process significantly speeds up the subsequent face detection procedure because only 3–9% regions of the original image are left for further processing.

3. Face verification

The second part of the designed system is to perform the task of face verification. In the previous section, we have selected a set of potential face regions in an image. In this section, we propose an efficient weighting mask function that is applied to decide whether a potential face region contains a face. There are three steps in this part. The first step is to normalize the size of all potential facial regions. The second step is to feed every normalized potential facial region into the weighting mask function and calculate the weight. The third step is to perform the verification task by thresholding the weight obtained in the previous step.

![Fig. 7. Three points \((i, j, k)\) that satisfy rules 1, 2, 3 and 4 will form a right triangle \((30°, 60°, 90°)\).](image)

![Fig. 8. Assume that \((Xi, Yi),(Xj, Yj)\) and \((Xk, Yk)\) are the three center points of blocks \(i, j, \) and \(k, \) respectively. \((X1, Y1),(X2, Y2),(X3, Y3), \) and \((X4, Y4)\) are the four corner points of the side view region.](image)

![Fig. 9. The search area of the third block center \(k\) is only limited to the dark area instead of the whole area of the image.](image)
3.1. Normalization of potential facial regions

Normalization of a potential facial region can reduce the effects of variation in the distance and location. Since all potential faces will be normalized to a standard size (e.g., 60 * 60 pixels) in this step, the potential facial regions that we have selected in the previous section are allowed to have different sizes. Here we resize the potential facial region by using “bicubic” interpolation technique.

3.2. Weighting mask function and weight calculation

If the normalized potential facial region really contains a face, it should have high similarity to the mask that is formed by 10 binary training faces. The idea is similar to neural network, but does not need so many training samples.

Every normalized potential facial region is fed into the weighting mask function that is used to compute the similarity between the normalized potential facial region and the mask. The computed value can be utilized for deciding whether a potential facial region contains a face or not.

The method for generating a mask is to read in 10 binary training masks that are cut manually from the facial regions of images, then add the corresponding entries in the 10 training masks to form an added mask. Next, binarize the added mask by thresholding each entry. For example, we have 10 masks and the size of each mask is 3 * 3. The first mask is formed by 9 “zero” (“zero” represents white pixel), so the first mask is a white 3 * 3 block. The 10th mask is produced by 9 “one” (“one” represents black pixel), so the 10th mask is a black 3 * 3 block. We add these 10 masks together and get an added mask.

Fig. 10. Example illustrating the generation of mask.

Fig. 11. Example illustrating the process of how to get the verified face from the face image with simple background (frontal view).
mask with the value of each entry as [7 8 9 4 5 6 3 2 1]. If we select the threshold value 5 (if the value of each entry is larger than 4, then we assign its value as 1; otherwise we assign its value as 0.), we get the final mask that has the values of [1 1 0 1 1 0 0 0]. The illustrating procedure is depicted in Fig. 10.

In our system, the mask is created by 60 * 60 pixels. Since the ratio of “the area of two eyes, the nose, and the mouth” and “the facial area excepting two eyes, the nose, and the mouth” is about 1/3, we design the algorithm for getting the weight of the potential facial region as follows.

For all pixels of the potential facial region and the mask: If both the potential facial region and the mask contain these pixels at the same location of the parts of two eyes, the nose, and the mouth (both of them are black pixels.), then weight = weight + 6. If both the potential facial region and the mask contain these pixels at the same location of the parts of the skin of the face (both of them are white pixels.), then weight = weight + 2. If the pixel of the potential facial region is black and the pixel of the mask is white, then weight = weight − 4. If the pixel of potential facial region is white and the pixel of the mask is black, then weight = weight − 2. The experimental results show that the values of +6, +2, −4, and −2, as given above, can obtain the best match of the facial region. In other words, the range of threshold values is the narrowest. The algorithm for obtaining the weight of the potential facial region is stated as follows:

**Input:** the potential facial region and the mask of the training samples

**Output:** the weight of the potential facial region

**Begin**

For all pixels of the potential facial region and the mask of the training samples

**Step (1) If** the pixel of the potential facial region is black and the pixel of the mask is also black, then weight = weight + 6.

**Step (2) If** the pixel of the potential facial region is white and the pixel of the mask is also white, then weight = weight + 2.

**Step (3) If** the pixel of the potential facial region is black and the pixel of the mask is white, then weight = weight − 4.

**Step (4) If** the pixel of the potential facial region is white and the pixel of the mask is black, then weight = weight − 2.

**Step (5) Calculate** the weight of the potential facial region for all pixels.

**End**

3.3. Verification

After we have calculated the weight of each potential facial region, then a threshold value is given for decision making. Once a facial region has been confirmed, the last step is to eliminate those regions that overlap with the chosen facial region, then exhibit the result.
Verification of the frontal view — A set of experimental results demonstrates that the threshold values of the frontal view should be set between 4000 and 5500. Verification of the side view — A set of experimental results shows that the threshold values of the side view should be set between 2300 and 2600.

3.4. Display the result

The processing of how to get the verified faces of one face image with simple background (frontal view) is shown in Fig. 11, where (a) is the original image, (b) is the binary image, (c) is the isosceles triangle formed by the 3 centers of 3 blocks, (d) is the best potential facial region cropped from the binary image that covers the isosceles triangle, (e) is the best potential facial region cropped from the original image that covers the isosceles triangle, (f) is the best potential facial region which is normalized to a standard size (60 * 60 pixels). Finally, (g) is the result of the facial detection.

The processing of how to get the verified faces from the face image with simple background (side view) is shown
in Fig. 12, where (a) is the original image, (b) is the binary image, (c) is the right triangle formed by the 3 centers of 3 blocks, (d) is the best potential face region cropped from the binary image that covers the right triangle, (e) is the best potential facial region cropped from the original image that covers the right triangle, (f) is the best potential facial region which is normalized to the standard size (60 × 60 pixels). Finally, (g) is the result of the facial detection.

4. Experimental results and discussion

In this section, a set of experimental results is demonstrated to verify the effectiveness and efficiency of the proposed system. There are 500 test images (including 450 different persons) containing totally 600 faces that are used to verify the validity of our system. Among them, only 11 faces cannot be found correctly. Experimental results demonstrate that an approximately 98%
Fig. 15. Verification of face images with different sizes.

Fig. 16. Verification of face images with altered lighting conditions.

Fig. 17. Verification of face images with distinct positions.

Fig. 18. Verification of face images with changed expressions.
(589/600 = 98.17%) success rate is achieved and the relative false rate is below 2% (11/600 = 1.83%). Some test images are taken using a digital camera, some from scanner, and some from videotape. The sizes of the test images range from 10 * 10 to 640 * 480 pixels. In these test images, human faces were presented in various environments. In this research, the minimum size of a face that could be detected is 5 * 5 pixels.

The execution time required to locate the precise locations of the faces in the test image set is dependent upon the size, resolution, and complexity of images. For example, the gray-level image which is 200 * 307 pixels as shown in Fig. 13(a) needs less than 2.5 s to locate the correct face position using PII 233 PC. The gray-level image which is 200 * 145 pixels as shown in Fig. 13(b) needs about 28 s to locate the correct face position because the background is more complex than Fig. 13(a).

The environments of the experimental setup are described as follows: (Fig. 13): experimental results of gray-level images with simple/complex backgrounds. (Fig. 14): experimental results of color images with simple/complex backgrounds. (Fig. 15): verification of face images with different sizes. (Fig. 16): verification of face images with altered lighting conditions. (Fig. 17): verification of face images with distinct positions. (Fig. 18): verification of
face images with changed expressions. (Fig. 19): verification of defocus face images. (Fig. 20): verification of face images (a) with noise; (b) with partial occlusion of mouth; (c) wearing sunglasses. However, our system cannot distinguish the real human faces from the virtual faces. For example, the virtual faces that are shown in Fig. 21(a) and (b) are regarded as real human faces, where Fig. 21(a) is the face of a cartoon doll (the weight is 4160) and Fig. 21(b) is the face of a Chinese doll (the weight is 4284). Shown in Figs. 22 and 23 are two examples where the faces cannot be detected. In Fig. 22, the image is too dark. As to the image in Fig. 23, the right eye is occluded by the black hair.

5. Conclusions

In this paper, a robust and effective face identification system is presented to extract face in various kinds of face images. The triangle-based segmentation process can reduce the background part of a cluttered image up to 97%. This process significantly speeds up the subsequent face detection procedure because only 3–9% regions of the original image are left for further processing. Moreover, it can manage different sizes, changed lighting conditions, varying pose and expression, and noise and defocus problem. In addition, the presented system can also cope with the problem of partial occlusion of mouth and wearing sunglasses. The system can also detect faces presented in the side view. The experimental results reveal that the proposed method is better than traditional methods in terms of altered circumstance, efficiency, and accuracy. In the future, we plan to use this face detection system as a preprocessing for solving face recognition problem.

6. Summary

In this paper, a triangle-based approach is presented to extract human faces embedded in photographs or images. The proposed system consists of two main parts. The first part is to search the potential face regions that are obtained from the triangles based on the rules of “the combination of two eyes and one mouth” (frontal view) or “the combination of one eye, one ear hole, and one mouth” (side view). The second part of the proposed system is to perform the face verification task. In the second part, each face candidate obtained from the previous process is normalized to a standard size. Then, each of these normalized potential face regions is fed to the weighting mask function to verify whether the potential face region really contains a face or not.

The proposed face detection system can locate multiple faces embedded in complicated backgrounds. Moreover, it is able to handle different sizes, different lighting conditions, varying pose and expression, and noise and defocus problems. In addition to overcoming the problem of partial occlusion of mouth and sunglasses, the system can also detect faces from the side view. Furthermore, the triangle-based segmentation process can reduce the background of a cluttered image up to 97%. This process significantly speeds up the subsequent face detection task because only 3–9% regions of the original image are left for further processing.

Experimental results demonstrate that an approximately 98% success rate is achieved and the relative false detection rate is very low.

References


About the Author—KUO-CHIN FAN was born in Hsinchu, Taiwan, on 21 June 1959. He received his B.S. degree in Electrical Engineering from National Tsing-Hua University, Taiwan, in 1981. In 1983 he worked for the Electronic Research and Service Organization (ERSO), Taiwan, as a Computer Engineer. He started his graduate studies in Electrical Engineering at the University of Florida in 1984 and received the M.S. and Ph.D. degrees in 1985 and 1989, respectively. From 1984 to 1989 he was a Research Assistant in the Center for Information Research at University of Florida. In 1989, he joined the Institute of Computer Science and Information Engineering at National Central University where he became professor in 1994. He was the chairman of the department during 1994–1997. Currently, he is the director of the Software Research Center at National Central University. Professor Fan is a member of IEEE, and a member of SPIE. His current research interests include image analysis, pattern recognition, and computer vision.

About the Author—CHIUNHSIUN LIN was born on 5 December 1961 in Chia-yi, Taiwan, Republic of China. He received his M.S. degrees in Computer Science from DePaul University, Chicago in 1993. He has been an assistant researcher in Committee for Planning and Organizing the National Taipei University since 1995. In 1996, he entered the Institute of Computer Science and Information Engineering at National Central University working toward his Ph.D. degree. His current research interests include pattern recognition, face detection, face recognition, image processing, and image analysis.