Locating head and face boundaries for head–shoulder images

Jianming Hu, Hong Yan*, Mustafa Sakalli

Department of Electrical Engineering, University of Sydney, NSW 2006, Australia

Received 8 October 1997; received in revised form 7 July 1998; accepted 31 August 1998

Abstract

This paper presents a model-based approach to locate head and face boundaries in a head-shoulder image with plain background. Three models are constructed for the images, where the head boundary is divided into left/right sub-boundaries and the face boundary is divided into left/right and top/bottom sub-boundaries. The left/right head boundaries are located from two thresholded images and the final result is the combination of them. After the head boundary is located, the four face sub-boundaries are located from the grey edge image. The algorithm is carried out iteratively by detecting low-level edges and then organizing/verifying them using high-level knowledge of the general shape of a head. The experimental results using a database of 300 images show that this approach is promising.

Keywords: Face detection; Face boundary models; Shape analysis; Feature extraction

1. Introduction

Computer recognition of human faces has many applications, such as personal identification, security systems, and human computer interfacing [1,2]. Locating faces or facial features in an image is an important step for face image processing and face recognition. For model-based image coding, it is a key problem to locate the boundary of a head in a head–shoulder image [3]. For face recognition, it has been shown that the background and head size can significantly affect the recognition performance [4]. The changing of hairstyles may also negatively affect the recognition. In addition, locating facial features accurately can align an image and therefore improve the performance of a recognition system.

There are many approaches to locating faces and facial features. According to the types of the images, algorithms are developed to locate facial features at plain background or complex background [4,17–20]. The light condition and the view point (front view or side view) are also restricted for some algorithms. In most of the approaches [5–13], the eyes/mouth features are located first and the face area is determined approximately according to the eye–eye and eye–mouth distances. This method is popular because it is robust even if the image has a complex background. In other approaches, the head/face boundaries are located first and the eyes/mouth features are then extracted [14–16].

In this paper, we address the problem of locating head and face boundaries in a front-view ID-type image with plain background. Some typical methods for locating head/face boundaries of this kind of images can be found in [7,16]. In [7] Li and Roeder used a simplified adaptive Hough transform (AHT) technique to identify straight cheek lines and chin lines. The relevant subimages are determined by the locations of eye and mouth, which are

*Corresponding author. Tel.: + 61 2 351-4824; Fax: + 61 2 351-3847; e-mail: yan@ee.usyd.edu.au
detected by the algorithms in [6]. The major limitation of this technique is the dependence on the accuracy of the eye and mouth detection algorithm and the quality of edge detection. In [16], Lee et al. applied a knowledge-based boundary tracking method based on the binary edge map, which is obtained using four-directional Sobel edge operators along with locally adaptive threshold (LAT). However, the head boundaries may be broken in the binary edge images, and therefore reduce the reliability of the algorithm. We will discuss the edge detection algorithms in detail in the next section.

Another major approach to locating head/face boundaries is to use snakes [3,8,14,21]. Snakes are the active contour models introduced by Kass et al. [22] which aims at minimizing the energy function of the active contour model. The energy function is the weighted sum of three components: the internal energy of the contour due to bending or discontinuities, the image force, and the external constraint forces. Huang and Chen [8] used the greedy algorithm for the minimal energy process, and Lam and Yan [14] used a fast greedy algorithm to improve the computational cost. However, the success of this approach is heavily dependent on the position of the initial head boundary. Furthermore, the snake may be attracted to false edges, such as those caused by shadows, beard and moustache.

In this paper, we combine traditional edge detection and thresholding techniques and head/face models to locate the head and face boundaries. By organizing local edge evidence to high level interpretation, this approach can locate the head and face boundaries effectively. In addition, some more useful information about the hairstyles and wearing of clothes, three head/face models can be constructed for ID type images. Fig. 1 shows the models, where (a) corresponds to the typical short hairstyle, (b) corresponds to the typical long hairstyle, and (c) corresponds to the situation when the clothes and the background have a very low contrast (e.g., white clothes against white background).

In the following, the leftmost point at the topmost row of an image is chosen as the origin of the coordinate system. The frame of the head in the image is located by horizontal and vertical histogram analysis of the edge image. The coordinates of the frame are denoted as \( y_{\text{min}}, x_{\text{min}} \) and \( x_{\text{max}} \) (Fig. 1a).

For simplicity, the head boundary is divided into left head boundary and right head boundary. The x coordinates of the left and right head boundary points at scanline \( i \) are denoted as \( h_l(i) = y_{\text{min}} \ldots, y_{\text{max}1} \) and \( h_r(i) = y_{\text{min}} \ldots, y_{\text{max}2} \) respectively (Fig. 1). To describe the face boundary, we use four sub-boundaries: \( f_l(i), f_t(i), f_r(i), f_b(i) \) for left, right, top, and bottom (chin) boundaries respectively.

2. Head/face boundary models

The head boundary refers to the outermost boundary of the head, while the face boundary refers to the face contour excluding the hair, shoulders, and the neck. According to the hairstyles and wearing of clothes, three head/face models can be constructed for ID type images. Fig. 1 shows the models, where (a) corresponds to the typical short hairstyle, (b) corresponds to the typical long hairstyle, and (c) corresponds to the situation when the clothes and the background have a very low contrast (e.g., white clothes against white background).

In the following, the leftmost point at the topmost row of an image is chosen as the origin of the coordinate system. The frame of the head in the image is located by horizontal and vertical histogram analysis of the edge image. The coordinates of the frame are denoted as \( y_{\text{min}}, x_{\text{min}} \) and \( x_{\text{max}} \) (Fig. 1a).

3. Locating head boundary

The flowchart of the algorithm for locating head boundary is shown in Fig. 2. The image processed here is front-view ID type image with grey level in the range \([0, 255]\). To smooth the image, a \(3 \times 3\) median filter can be applied [23]. The grey edge image is obtained by the templates described in Section 3.1, and the frame of the face can be located using horizontal and vertical histogram analysis. At this stage, we can also normalize

---

Fig. 1. The head and face boundary models.
the image by interpolation [24]. In our experiments, the width of the normalized image is set to 164 while its height is scaled proportionally. From the grey edge image, two thresholded binary images are obtained. The high thresholded image can reduce the effect of the shadows in the image and the background noise, while the low thresholded image can result in smoother boundaries. By analyzing and combining the boundaries of the two binary images, the head boundaries can be obtained.

3.1. Edge detection and thresholding

Edges characterize object boundaries and are therefore useful for segmentation, registration, and identification of objects in a scene. Edge detection is a fundamental problem in image analysis, and many edge detection methods are available [26,27]. The performance of an edge detection algorithm is related to the processed image and the application (how the edge information is used by the following algorithms). In practical use, the speed of the algorithm is also an important consideration.

In our application, we need thick and strong edges to reduce broken boundaries in the image. Since a fixed threshold is not sufficient to segment images with different contrast, we need to analyze the edge images at different resolution [26]. Therefore, we first obtain a grey edge image rather than a binary edge image. The Sobel operations [25] are simple and effective to generate grey edge images, but the edges for some low contrast images (especially at the checks) are very weak. We tried larger windows and a number of different patterns. The patterns shown in Fig. 3 give better results. Figs. 3a and b are used to detect vertical and horizontal edges respectively, while Figs. 3c and d are used to detect slant edges. Suppose the output of the four masks for pixel \( p(i, j) \) are \( S_1, S_2, S_3, \) and \( S_4 \) respectively, the edge at \( p(i, j) \) is chosen as

\[
e(i, j) = \max\{S_1, S_2, S_3, S_4\}
\]  

The Canny edge detection algorithm can also be used to produce similar strong edges, but the computational cost is high. Therefore, we use the four line detection masks in Fig. 3 and select the largest output as the edge magnitude. Fig. 4 shows some examples, where the leftmost column shows the original images, the second, third and last columns show the results of Sobel detection, the proposed detection and the Canny detection, respectively.

To find the head boundary, the simplest way is to segment the image to binary image and trace the left and right boundaries of the segmented image. However, a single threshold does not work well for all the images where the contrasts may vary significantly. In our approach, we use two adaptive thresholds (a high threshold and a low threshold) for each image to obtain two thresholded images. The high threshold is chosen such that the edges due to shadows can be eliminated. The low threshold is chosen such that the smoothness of the boundary can be maintained while the background noise can be reduced as much as possible.

Since the frame of the image has been located, the high threshold \( T_h \) can be calculated in this area adaptively by taking all the high intensity pixels which occupy 30% of the image area. Suppose that the image height and width are \( hg \) and \( wd \) respectively, and that the histogram for the image is \( \text{hist}(i) \ (i = 0, 1, \ldots, 255) \). \( T_h \) is then determined as the smallest value which satisfies the following equation:

\[
\sum_{i=T_h}^{255} \text{hist}(i) \geq 0.3 hg \times wd.
\]
Fig. 3. The masks used to detect greylevel edges.

The low threshold $T_l$ can be determined similarly by taking more pixels (45%) as the edge pixels:

$$\sum_{i=T_h}^{255} \text{hist}(i) \geq 0.45hg \times wd.$$  (3)

If $T_l$ is very low ($T_l \leq 25$), the minimum value of $T_l$ is set to 25 to ensure the background noise is excluded. The thresholded edge images are then smoothed using morphological operations. The smoothing is carried out by an opening followed by a closing, where a $3 \times 3$ square structure element is used [25].

Fig. 5 shows some examples, where Fig. 5(1) and (5) is the original images, 5(2) and (6) is the grey edge images, Fig. 5(3) and (7) is the high thresholded images, and Fig. 5(4) and (8) are the low thresholded images. For some images with good contrast and smooth background, the differences between the head boundaries of the high and low thresholded images are small (Fig. 5(3) and (4)). In Fig. 5(7) the left head boundary is broken while in Fig. 5(8) the edges due to the shadows are detected in the right head boundary.

3.2. Boundary tracing

From an initial boundary point, we trace along the boundary by analyzing the connectivity of the boundary pixels.

Denote a left boundary pixel at scanline $i$ as $p_l^0(x_i, y_i)$. The adjacent pixel at scanline $i + 1$ is then $p_l^{i+1}(x_{i+1}, y_{i+1})$, where $y_{i+1} = y_i + 1$. Pixel $p_l^i(x_i, y_i)$ is connected to $p_l^{i+1}(x_{i+1}, y_{i+1})$ if the following conditions are true:

$$p_l^i(x_i + k, y_i) = 255 \ (k = 1, \ldots, x_{i+1} - x_i - 1)$$

if $x_{i+1} \geq x_{i}$;  \hspace{1cm} (a)

$$p_l^i(x_{i+1} + k, y_{i+1}) = 255 \ (k = 1, \ldots, x_i - x_{i+1} - 1)$$

if $x_{i+1} < x_{i}$.  \hspace{1cm} (b)

The connectivity for right boundary pixels can be determined similarly. Fig. 7 shows all the situations, where (a) and (b) are for left and right boundaries, respectively. According to the connectivity of boundary pixels, a boundary can be divided into segments where the boundary points in each segment are connected. For the convenience of later processing, the boundary points in one segment are given the same label. In the tracing procedure, the tracing for a boundary segment is stopped when the scanning point reaches the border of the image frame ($y_{\text{max}1}$ and $y_{\text{max}2}$ can then be determined), or the next boundary point cannot be found. A complete boundary segment is the boundary segment where the starting point begins at $y_{\text{min}}$ and ends at $y_{\text{max}1}$ or $y_{\text{max}2}$. If the first traced segment is a complete boundary segment, the tracing procedure is stopped. Otherwise, all
the boundary segments are traced. To eliminate the background noise, a boundary segment less than \((h_g - y_{min})/20\) is discarded.

Figs. 6(3) and (5) shows the traced boundaries for the high and low thresholded images in Fig. 6(2) and (4), respectively. We can see that the background noise was excluded in the traced boundaries in Fig. 6(4).

### 3.3. Boundary analysis

The traced left/right boundary consists of a complete boundary segment or several boundary segments. If the boundary consists of several boundary segments, we first process the gaps between the two boundary segments. If there is a large gap between two boundary segments, we can segment a small area around the gap in the grey edge image according to Eq. (3). For example, the area marked with a square in Fig. 5(7) is segmented again, and the result is shown in Fig. 5(9). In this case, the boundary is relocated. If the gap between two segments is small, we draw a line segment to connect the two end points and locate a boundary pixel nearby the line segment from the grey edge image (details are described in Step 7 in the following).
The located boundary is then smoothed. The boundary data is first smoothed by a $3 \times 3$ median filter, and it is further smoothed by morphological smoothing [25] where the length of the structure element is set to 7. The smoothing scheme can smooth the boundary data effectively, and it is also used in later boundary smoothing (Fig. 2). After data smoothing, the smoothness analysis of a boundary is carried out.

To measure the smoothness of a boundary, the first derivative of the data is used. The first derivative $d_1(i)$ for the left boundary is determined by

$$d_1(i) = h_l(i) - h_l(i-1) \quad (i = y_{\min} + 1, \ldots, y_{\max}) \quad (4)$$

where $h_l(i)$ is the coordinate of the boundary pixel (Section 2). Based on the first derivative $d_1(i)$, we can find the critical points of the boundary. Four kinds of critical
points are defined as follows:

(a) $A_1$ point: a turning point where the boundary goes left and then goes right (Fig. 8a);
(b) $A_2$ point: a turning point where the boundary goes right and then goes left (Fig. 8b);
(c) $B_1$ point: a point where the boundary goes right at a leap or at discontinuous (Fig. 8c);
(d) $B_2$ point: a point where the boundary goes left at a leap or at discontinuous (Fig. 8d).

The critical point detection algorithm is crucial in the algorithm. There are many methods such as the algorithms in [28,29] for this purpose. To simplify the problem, we defined the above four kinds of critical points. We can see that $A_1$ and $A_2$ points have unique definitions, while $B_1$ and $B_2$ are determined experimentally. In our experiment, $B_1$ point is located if one of the following conditions is satisfied:

(a) $d_1(i) > T_1$ (large leap);
(b) $d_1(i) > T_2$ and $\text{label}(i) = \text{label}(i - 1)$ (small leap but discontinuous).

$B_2$ point can be determined in a similar way. $T_1$ and $T_2$ can be determined experimentally according to the size of the image frame. We set $T_1 = 8$ and $T_2 = 3$ in our experiments. After the critical boundary points are
located, we can relocate (repair) the boundary segment which is not smooth. The whole procedure for left boundary is described as follows:

1. Trace the left boundary of the thresholded image.
2. Determine the connectivity of the boundary pixels as described in Section 3.2.
3. Smooth the boundary data.
4. Find the critical points of the boundary.
5. Find the first $B1$ point as the starting point $s$. From the $s$ point, find a $A1$ or $B2$ point as the end point $t$ which satisfies the condition

$$|h_i(t) - h_i(s)| < t - s - 1.$$  \hspace{1cm} (5)

If $s$ point is found but $t$ point is not found, then the first boundary point from $s + 1$ which satisfies Eq. (5) is chosen as the $t$ point (Fig. 9a).

6. If $s$ point is not found, then find a $B2$ point as the $t$ point. Search back for the first boundary point as the $s$ point which satisfies Eq. (5) (Fig. 9b).

7. For the boundary segment between $s$ and $t$, a reference point is first calculated for each scanline which is the point on the line connecting the $s$ point and $t$ point. If the position of the reference point is $x_0$, then the relocated edge pixel is the point with the largest grey-level value in the grey edge image among the points $[x_0 - 4, x_0 + 4]$.

8. Repeat Steps 3–7 until no more $s$ point or $t$ point can be found.

Fig. 6(1)–(5) show the original image, high/low thresholded images, and their traced boundaries. Both the left and right head boundaries for the high thresholded image shown in Fig. 6(2) are complete and smooth. The boundaries in Fig. 6(5) are complete, but the top segment of the right boundary is not smooth. Combining the boundaries of the two thresholded images, we can obtain the final head boundary.
3.4. Boundary combination

The traced boundaries from the high thresholded image are more robust to shadows and background noise than that from the low thresholded image. However, more boundary pixels may be missing in the high thresholded image. On the contrary, boundaries in the low thresholded image tend to be smoother than that in the high thresholded image, but edges formed by the shadows may be detected in the head boundary. Therefore, we need to combine the two results to obtain a smoother and reliable boundary.

Denote the $x$ coordinates of the left boundaries of the high and low thresholded images as $h_{l1}(i)$ and $h_{l2}(i)$ ($i = y_{\min}, \ldots, y_{\max}$), respectively. Obviously, $h_{l1}(i) \geq h_{l2}(i)$. The combination result $h_{l}(i)$ of $h_{l1}(i)$ and $h_{l2}(i)$ is determined as follows:

(a) If the left boundary for the high thresholded image is complete and smooth, then it is taken as the final left head boundary: $h_{l}(i) = h_{l1}$;
(b) If the left boundary for the high thresholded image is not complete or smooth, but if the left boundary for the low thresholded image is complete and smooth, then we have $h_{l}(i) = h_{l2}$;
(c) In all other cases, the final left boundary is taken as the left boundary for the low thresholded image and the boundary is analyzed and repaired according to the procedure described in Section 3.3.

The combination scheme for the right head boundary is similar to that for the left boundary. Fig. 6 shows several examples of the combination results. For the images in Fig. 6(1)–(5), the final left and right head boundaries are taken as that for the high thresholded image (Fig. 6(3)). Another example is given in Fig. 6(6)–(8), where the boundaries for the high and low thresholded images are shown in Fig. 6(6) and (7), respectively. The right boundary of the high thresholded image is complete and smooth, but the left boundary is broken. Both the left and right head boundaries for the low thresholded image are complete and smooth. Therefore, the final head boundary in Fig. 6(8) consists of the left boundary of Fig. 6(7) and the right boundary of Fig. 6(6). Fig. 6(9)–(12) shows one more example, where 6(9) shows the boundaries of high thresholded image, 6(10) shows the boundaries of low thresholded image, 6(11) is the combination result, and 6(12) is the final result after boundary repairing.

4. Locating face boundary

The face boundary consists of four sub-boundaries: $f_{l}(i), f_{t}(i), f_{r}(i)$, and $f_{b}(i)$ (Section 2). Since the methods for locating left, right, and top face boundaries are similar, we will only describe the algorithm for locating left face boundary. The bottom face boundary, however, is more difficult to locate. In our approach, we use a model-based method to approximate the bottom (chin) boundary.

4.1. Left face boundary

The procedure for locating the left face boundary is shown in Fig. 10. The working images are the original image $I(x, y)$ and the grey edge image $I_{e}(x, y)$. In order to find the border pixel between the hair and face, the average intensity of the hair nearby the left boundary is computed. The initial average hair intensity $g_{a}$ is estimated by

$$g_{a} = \sum_{i = y_{\min} + 60}^{y_{\min} + 40} \sum_{j = 5}^{10} g(h_{l}(i) + j, i)/100,$$

Fig. 10. Diagram for locating face boundary.
where $g(x, y)$ is the intensity of pixel $I(x, y)$ and $h_l(i)$ is the left head boundary. In case of the hair area being very small, $g_a$ is refined by excluding very high and very low intensity pixels ($|I(x, y) - g_a| > 80$) in this area.

The next step is to find a face border pixel at each scanline. The searching ranges are $[h_l(i) + 1, w/3]$ for $y_{\text{min}} < i \leq (y_{\text{max}1} - y_{\text{min}})/2$ and $[h_l(i) + 1, w/2]$ for $(y_{\text{max}1} - y_{\text{min}})/2 < i < y_{\text{max}1}$, where $w$ is the image width.

Fig. 11. Face boundary location: (1) initial left face boundary; (2) the final left face boundary after smoothing between the starting and end points; (3) final results of the left/right/top face boundaries; (4) another example of the face boundaries; (5)-(8) illustration of the bottom face boundary location.
and \( y_{\text{min}} \) and \( y_{\text{max}} \) are the starting and end points of the left head boundary (Fig. 1). Let \( a = g(j + 3, i) + g(j + 4, i) + g(j + 5, i) \), \( b = g(j - 3, i) + g(j - 4, i) + g(j - 5, i) \) and \( c = |a - b| \) at scanline \( i \). The border pixel \( I(j, i) \) is the point which has the maximum \( c \) value in the searching range and satisfies the following conditions:

(a) \( c > 40 \) and \( |b - g_d| < 60 \), if \( g_a(j, i) \geq 200 \);
(b) \( c > 30 \) and \( |b - g_d| < 50 \), if \( 60 < g_a(j, i) < 200 \);

where \( g_a \) is the average intensity of the hair and \( g_e(x, y) \) is the intensity of the grey edge pixel \( I_e(x, y) \).

Fig. 11(1) shows the detected left face border pixels of a face image. If a border pixel is found at \( (j, i) \), then we have \( f_0(j) = j \). If no border pixel is found, we set \( f_0(j) = 0 \).

The face border data is then smoothed using the same smoothing procedure described in Section 3.3. The initial starting point \( s \) of the face border is the first point from \( y_{\text{min}} \) which satisfies

(a) \( f_0(s) > 0, f_0(s + 1) > 0, f_0(s + 2) > 0 \), and
(b) \( |f_0(s) - f_0(s + 1)| < 4, |f_0(s + 2) - f_0(s + 1)| < 4 \).

Similarly, we can search back from \( y_{\text{max}} \) to find the initial end point \( t \). For the face border between \( s \) and \( t \), it is divided into several segments by detecting any two adjacent points \( f_0(i) \) and \( f_0(i + 1) \) which satisfy \( |f_0(i) - f_0(i + 1)| < 4 \). The points in a segment are labeled with the same symbol, which is convenient for later smoothing. Denoting the starting and end points of the previous segment, the current segment and the next segment as \((s_1, t_1), (s_2, t_2)\), and \((s_3, t_3)\) respectively, the current segment can be repaired (it is substituted by the straight line connecting \( t_1 \) and \( t_3 \)) if the following conditions are true:

(a) \( f_0(t_1) - f_0(s_2) > 3 \) and \( f_0(s_3) - f_0(t_2) > 3 \); or
(b) \( t_1 - s_1 > t_2 - s_2 \) and \( t_3 - s_3 > t_2 - s_2 \).

After repairing, we find the starting point \( s \) and the end point \( t \) of the left face boundary again. The starting point \( s \) is taken as the first point of the first segment which are more than 4 pixels in length. From the starting point \( s \), we find the end point \( t \) when a zero value \( f_0(i) = 0 \) is met or very short scattered segments are met. Fig. 11(2) shows the smoothed left face border between the final starting point and end point.

Similar procedure can be applied to locate the right face boundary \( f_0(i) \) and the top face boundary \( f_0(i) \). Fig. 11(3) shows the result for the face image in Fig. 11(1) while Fig. 11(4) shows the result for another face image.

If no smooth face boundary segment can be found (no hair is present, or the color of the hair is very similar to the skin of the face), the face boundary is the same as the head boundary.

### 4.2. Bottom face boundary

The bottom face boundary is the border of the chin. Since the beard and the shadow below the chin can produce "wrong" edges, and since the contrast at the chin may be very low (very weak edges), we cannot rely on the result of edge detection only (refer to Section 4 in [7]). In our approach, we use a segment of ellipse to approximate the chin shape. The initial position is determined according to the head and face boundaries. Edges are then detected near the segment of ellipse, and the position of the ellipse in turn is adjusted according to the detected edges.

The ellipse models are shown in Fig. 12, where (b) and (c) are the rotated cases of (a). To draw the ellipse arc, we need to know the lengths of the two axes of the ellipse and the two points \( p_1(x_1, y_1) \) and \( p_2(x_2, y_2) \). If the minimum \( y \) coordinate of the top face boundary is \( y_0 \), then the length of the long axis of the ellipse is approximated by \( a = (y_1 + y_2)/2 - y_0 \). Denoting the minimum \( x \) coordinate of the left face segment above \( y_1 \) as \( x_0 \) and the maximum \( x \) coordinate of the right face segment above \( y_2 \) as \( x_0 \), the length of the short axis can be estimated as \( b = x_0 - x_0 \). Therefore, the main problem is to determine the coordinates of \( p_1 \) and \( p_2 \).

Since the head boundary and the left and right face boundaries have been located, we can determine the type of the face according to Fig. 1. First, shape analysis is carried out for the head boundary. The shape of the boundary is characterized by its convex/concave changes. We find out all the feature points as shown in Figs. 8a and b, and analyze the position of each feature point as well as their relative positions. For the first and the third types of head boundaries in Fig. 1, the feature points \( p_1 \) and \( p_2 \) are determined at one of the head boundaries. The left and right face boundaries are then extended to join the feature points \( p_1 \) and \( p_2 \). Fig. 11(5) shows the located feature points \( p_1 \) and \( p_2 \) and the initial ellipse arc. For the second type in Fig. 1 (long hair), the feature points \( p_1 \) and \( p_2 \) are determined from the face boundaries only. Fig. 11(7) shows the located feature points \( p_1 \) and \( p_2 \) and the initial ellipse arc.

From the initial ellipse arc, we search for the edge pixels of the chin. If the \( y \) coordinate of the ellipse at column \( i \) is \( j \), we search for the strongest edge pixel in the range \([j - 10, j + 10]\) at column \( i \). Figs. 11(6) and (8) show the detected edge pixels corresponding to Fig. 11(5) and 11(7), respectively, where the initial ellipse arcs are also drawn in the images. We then calculate the average difference \( d_0 \) (in \( y \) direction) of the initial ellipse arc and the detected edges. The final bottom (chin) boundary is the initial ellipse adjusted by \( d_0 \) in \( y \) direction (Fig. 13(1) and (10)).
5. Experiments

We use the database (FullFaces) produced by the University of Bern in the experiments. The database contains front view images of 30 people. For each person there are 10 greyscale images with slight variations of the head positions (1, 2 right to the camera, 3, 4 looking to the right, 5, 6 looking to the left, 7, 8 upwards, 9, 10 downwards). Each image has a size of 512 × 342 pixels with 256 greylevels. Since the original image has a large area of background, we reduce the size of the image by shifting the face frame in the middle of the image (this can be done by horizontal and vertical histograms analysis). The width of each image is then normalized to 164. The images of the first 10 subjects and 10 scanned photographs were used in the training. The remaining images are used in the testing.

Since the algorithms are designed to deal with passport type of images, we focus on the first two images (1, 2 right to the camera) of each person. Fig. 13 shows the results for the first image of each person in the database (the last two are scanned photographs). For images (16) and (17), the bottom face boundaries are not located since the program fails to determine the feature points $p_1$ and $p_2$.

Although the algorithms are designed to deal with passport type of images, they can be applied to other images except for the procedure for locating the bottom face boundary. Table 1 shows the testing results for all the images in the database, where the result for each boundary is evaluated as “G” (good), “M” (moderate), “B” (bad), and “F” (fail) manually. In the case of “F”, the corresponding boundary is not located. Fig. 14 shows some examples: (1)−(4) good, (5)−(8) moderate, (9)−(12) bad, and (13)−(16) failed (bottom boundary).

From Table 1, we can see that the algorithms for locating head boundary, left/right/top face boundaries can be applied to all the images with success for most cases. The bottom face boundary is most difficult to locate. In the experiment, we only designed the procedure to deal with passport type of images (images 1 and 2). For other situations, more comprehensive analysis of the head boundary and the left/right boundaries are needed.

As discussed in [30], two of the most critical requirement in support of producing reliable face-recognition systems are a large database of facial images and a testing procedure to evaluate systems. We used the Bern face database and a number of scanned images in our experiments. The images in MIT face database have a complex background and the images in the Olivetti database have no background (the face has been located already). It should be pointed out that the head/face boundary location algorithms are suitable for passport type of images, in which the light conditions are generally good and the background are generally clean. If the background is not clean (a natural scene), or if there is a significant shadow (e.g., the light is on one side of the face), it is suggested the
eyes and mouth features are located directly (see Section 1). The bottom face boundary (chin) detection is the most difficult problem in face boundary detection since the edge information is generally not sufficient (see also Section 4 in [7]). In our current study, we deal with ID type front view images. More research works need to be carried out for other situations (as shown in Fig. 14).

6. Discussion and conclusion

Locating head and face boundaries is an important task in automatic face recognition. Many approaches are based on “global” methods, for example, snakes and Hough transform. It is believed that it is generally very difficult to organise low-level (edge) information into a
sensible contour for an object. However, for face recognition, we have a priori knowledge of the general shape of a head, which has been classified into three models in our experiments. For images with plain background, the frame of the head-shoulder image can easily be located by histogram analysis. Therefore, the face image can be assumed at the center of the frame approximately. Thus, the searching area for head and face boundary can be specified. The division of head/face boundaries into several sub-boundaries has the advantage of easy handling.
Table 1
Evaluation of the head and face boundary location by the proposed algorithms

<table>
<thead>
<tr>
<th>Image</th>
<th>Boundary</th>
<th>Head</th>
<th>Left face</th>
<th>Right face</th>
<th>Top face</th>
<th>Bottom face</th>
</tr>
</thead>
<tbody>
<tr>
<td>G</td>
<td>34</td>
<td>39</td>
<td>39</td>
<td>37</td>
<td>31</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td></td>
<td>30</td>
<td>33</td>
<td>35</td>
<td>34</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>8</td>
<td>6</td>
<td>5</td>
<td>6</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>26</td>
<td></td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>38</td>
<td>37</td>
<td>38</td>
<td>11</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>22</td>
<td></td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>37</td>
<td>36</td>
<td>36</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>7</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>34</td>
<td>38</td>
<td>38</td>
<td>35</td>
<td>18</td>
<td></td>
</tr>
<tr>
<td>M</td>
<td>5</td>
<td>2</td>
<td>2</td>
<td>4</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>13</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>165 (82.5%)</td>
<td>185 (92.5%)</td>
<td>185 (92.5%)</td>
<td>180 (90%)</td>
<td>78 (39%)</td>
<td></td>
</tr>
</tbody>
</table>

The boundary smoothing and boundary shape analysis are crucial in the algorithm. With boundary smoothing, the redundant critical feature points in the boundary can be reduced significantly. The algorithm is carried out iteratively by detecting low-level edges and then organising/verifying them using high-level knowledge of the general shape of a head. The experimental results show that this approach is promising.

7. Summary

Locating head and face boundaries is an important task in model based image coding and automatic face recognition. Many approaches are based on global methods such as snakes and the Hough transform. In general, it is very difficult to integrate low-level (edge) information into a contour of an object. However, for face recognition, we can make use of a priori knowledge of the general shape of a head, which has been classified into three models in our experiments. For images with plain background, we could detect the head and face edges and organize them into meaningful boundaries based on the three models. First, the frame of the head/shoulder image can easily be located by histogram analysis. Thus, the searching area for head boundary can be specified. Second, a grey edge image is obtained by extended Sobel operations. From the grey edge image, two thresholded images are obtained. The high-thresholded image is used to obtain strong edges while excluding the edges caused by shadows, and the low-thresholded image is used to provide more edges when the contrast of the image is low. The head boundary is divided into left/right sub-boundaries where each of the initial sub-boundary can be obtained from the edges directly. The boundary is then smoothed and analyzed against the three models, where a broken boundary can be detected and repaired. The final result is the combination of the boundaries from the two thresholded images. The algorithm is carried out iteratively by detecting low-level edges and then organizing/verifying them using high-level knowledge of the general shape of a head. The face boundary is located, the four face sub-boundaries are located from the insides of the head boundaries. The division of head/face boundaries into several sub-boundaries has the advantage of efficient processing. The boundary smoothing and boundary shape analyses are crucial steps in this algorithm. The experimental results on a database of 300 images show that this approach is promising.
Fig. 14. Examples of the results when the proposed algorithms were applied to other images in the database.

References


---

**About the Author**—JIANNMING HU received his MSE degree in Communication and Electronic Systems from Xidian University in 1990, and his Ph.D. degree in Electrical Engineering from the University of Sydney in 1997. He is now a senior research assistant in the Department of Electrical Engineering, University of Sydney. His research interests are in the areas of information theory, coding theory, image processing and pattern recognition.

**About the Author**—HONG YAN received his B.E. degree from Nanking Institute of Posts and Telecommunications in 1982, M.S.E. degree from the University of Michigan in 1984, and Ph.D. degree from Yale University in 1989, all in electrical engineering. From 1986 to 1989 he was a research scientist at General Network Corporation, New Haven, CT, USA, where he worked on developing a CAD system for optimizing telecommunication systems. Since 1989 he has been with the University of Sydney where he is currently a Professor in Electrical Engineering. His research interests include medical imaging, signal and image processing, neural networks and pattern recognition. He is an author or co-author of one book, and more than 200 technical papers in these areas. Dr. Yan is a fellow of the Institution of Engineers, Australia (IEAust), a senior member of the IEEE, and a member of the SPIE, the International Neural Network Society, the Pattern Recognition Society, and the International Society for Magnetic Resonance in Medicine.

**About the Author**—MUSTAFA SAKALLI received his BSc degree from the Faculty of Electrical Engineering, Istanbul Technical University, in 1980 and MSc degree from the Institute of Biomedical Engineering, Bogazici University, in 1988. He is currently a research assistant and is working toward a Ph.D. degree in the Image Processing Research Group, Department of Electrical Engineering, University of Sydney.