A Highly Efficient System for Automatic Face Region Detection in MPEG Video

Hualu Wang and Shih-Fu Chang*

***Abstract*--Human faces provide a useful cue in indexing video content. In this paper, we present a highly efficient system that can rapidly detect human face regions in MPEG video sequences. The underlying algorithm takes the inverse quantized DCT coefficients of MPEG video as the input, and outputs the locations of the detected face regions. The algorithm consists of three stages, where chrominance, shape, and frequency information are used respectively. By detecting faces directly in the compressed domain, there is no need to carry out the inverse DCT transform, so that the algorithm can run faster than the real time. In our experiments, the algorithm detected 85-92% of the faces in three test sets, including both intra-frame and inter-frame coded image frames from news video. The average run time ranges from 13 to 33 milliseconds per frame. The algorithm can be applied to JPEG unconstrained images or motion JPEG video as well.**

***Index Terms*--Face detection, MPEG video, compressed-domain approach, video indexing.**

I. INTRODUCTION

Digital images and video play an important role in the multimedia information era. The huge amount of visual information is handled by image and video databases, which require effective and efficient mechanisms for indexing and search. In recent years, techniques have been proposed that allow users to search images by visual features, such as texture, color, shape, and sketch, besides traditional textual keywords [1]. Algorithms have also been proposed and summarized in [2] to explore automatic detection of low level visual features directly from the compressed domain, and the synergy between feature extraction and compression.

The human face is an important subject in image and video databases, because it is a unique feature of human beings, and is ubiquitous in photos, news video, and documentaries. The face can be used to index and search images and video, classify video scenes (e.g., anchorperson shots in news video), and segment human objects from the background. Therefore, research on face detection is critical in image and video database applications.

Although face detection is related to face recognition, the problem here is a little different from those in traditional face recognition scenarios. As summarized in [3], past work on face recognition has been focused on digital images taken in highly constrained environments. Strong assumptions are used to make the task more tractable. For example, there is usually just one front-view face in the center of the image, the head is upright, the background is clean, no occlusion of faces exists, no glasses are worn, and so on. The existence and locations of human faces in these images are known *a priori*, so there is little need to detect and locate faces. Face recognition has been an active research field for more than twenty years. Some recent techniques have been proposed using neural nets [4], deformable template matching [5], and Karhunen-Loeve expansion (i.e., eigenfaces) [6]. All these methods, as well as traditional ones, can be roughly classified into two broad categories, namely geometric-feature-based matching, and template matching [7].

In image and video databases, however, there is generally little or no constraint on the number, location, size, and orientation of human faces in the image or video scenes. The background of these images and video scenes is usually complex. Thus, face detection becomes important and challenging before the indexing, search, and recognition of the faces can be done.

* The authors are with the Department of Electrical Engineering and Center for Telecommunications Research, Columbia University, New York, NY 10027.

Recently, work has begun on face detection in unconstrained images. The phrases *face detection*, *face location*, and *face localization*, are used interchangeably in literature. For consistency we use the phrase *face detection* in this paper. Govindaraju *et al* [8] proposed a model-based approach where the face is defined as inter-connected arcs that represent chins and hairline. The arcs are extracted using low level computer vision algorithms, and are then grouped based on cost minimization to detect candidate face regions. Ten images are tested without any miss. However, false alarms are often generated. Burl *et al* [9] presented a face detection system in which local feature detectors are coupled with a statistical model of spatial arrangement of facial features. A set of facial features (e.g., eyes, nostrils, nose-mouth junction) is detected first, and is used to form constellations of feature sets. A ranking technique with a statistical model of spatial arrangements of these features is then applied to detect possible face regions in the constellations of feature sets. To facilitate the detection process, an intelligent search scheme in the constellations is used. Evaluation on a database of 150 images (quasi-frontal, under the same lighting condition) indicates a correct detection rate of around 84%. A low complexity algorithm to detect and track face regions was proposed in [10] for model-assisted coding of low-bit-rate teleconferencing video. It is a three-step hierarchical procedure utilizing the fact that the human face outline is roughly elliptical. Yang and Huang [11] proposed a hierarchical knowledge-based algorithm to detect human faces in a complex background. The algorithm consists of three levels. The higher two use mosaic images of different resolutions. The third one extracts edges of facial components. Domain knowledge and rules are applied at each level. A detection rate of 83% is reported (50 faces from 60 512x512 images) with 28 false alarms. The run time of face detection is 60-120 seconds on a SUN Sparc 2 station. A neural network-based face detection system was reported comprehensively in [12] by Rowley *et al*. A set of neural network-based filters is first applied to an image at several scales. An arbitrator is then used to combine the filter outputs. The algorithm is able to detect 90.5% of the faces in 130 images from three different sources, many of which contain multiple faces. Computational complexity is high because the neural networks have to process many small local windows in the images. Wavelet transform domain has been explored for face detection as well. Venkatraman and Govindaraju [13] used zero-crossings of a wavelet transform at different scales to extract local facial features. These features are then combined in a model matching stage to detect faces.

Although face detection has been investigated using various approaches, there is still much to explore. Current techniques usually carry out face detection in the uncompressed pixel domain. Face detection directly in the compressed-domain has not been investigated adequately. The test sets are usually gray-level still images. Video sequence and color information are seldom addressed. Computational complexity, which is an important factor in image/video database applications, is not addressed adequately in the aforementioned algorithms. The run time of the proposed algorithms is usually long, if reported at all. Performance evaluations, especially that of false alarm rates, are sometimes missing.

In this paper, we propose a fast algorithm that automatically detects faces in MPEG compressed video. The face detection algorithm consists of three stages, where chrominance, shape, and DCT frequency information are used respectively. The algorithm starts at the MPEG macroblock level, a lower resolution version of the video frames, so that the amount of data to be processed can be greatly reduced. Moreover, the algorithm uses DCT coefficients directly so that only minimal decoding of MPEG streams is necessary. Therefore, our algorithm is able to run in real time on regular workstations. It is important to point out that our algorithm focuses on speed, which is crucial to the indexing and search of large video databases, rather than on accuracy†. As a trade-off, locations of face regions are approximated by their bounding rectangles. Although not pixelwise face outlines, they provide important information for video indexing, scene classification, and face recognition. We argue that this level of outline resolution is suffi-

† Nevertheless, we will show later that our three-stage algorithm can still achieve a high detection rate.

cient for video indexing and search, in which detection of prominent objects is more important than detailed low-level features. Sophisticated pixel-domain analysis can be applied to the detected regions if needed. This also provides a promising direction for efficient and accurate face recognition.

This three-stage algorithm is applied directly to I frames in MPEG video. For P and B frames, only the first two stages are used. The third one, which is a verification stage, is omitted to avoid the expensive computation of DCT AC coefficients in inter-coded frames. This allows the algorithm to achieve real-time speed, at the cost of a higher false alarm rate.

The paper is organized as follows. Section II presents an overview of the proposed algorithm, along with system design principles. Section III, IV, and V describes in detail the three stages of our algorithm, respectively. Section VI explains special handlings of inter-coded frames. Section VII gives experimental results and discussions. The conclusion and discussion of future work are given in Section VIII.

II. Overview and Design Principles

We present a fast algorithm that automatically detects face regions in MPEG-compressed video. The algorithm takes as input the inverse-quantized DCT coefficients of the MPEG macroblocks, and generates bounding rectangles of detected face regions. By detecting faces using just the DCT coefficients, we avoid the expensive inverse DCT transform. Thus, only minimal decoding of MPEG video is necessary, and the algorithm is able to achieve high speed.

MPEG video consists of three different types of frames, namely I (intra-frame coded), P(one-way predictive coded), and B(bi-directional predictive coded) frames [14]. For the purposes of indexing and search, face detection in I frames is usually sufficient. This is because faces in video scenes usually stay much longer than the duration of an MPEG group of pictures (GOP), which usually consists of 12 to 15 frames (about 0.5 second duration). After minimal decoding of an MPEG stream, the DCT coefficients can be obtained easily for the luminance and chrominance blocks in I frames. If face detection in B or P frames is desired, one can apply transform-domain inverse motion compensation to obtain the corresponding DCT coefficients for blocks in B and P frames. The DCT coefficients of the translated block in the reference frame can be computed using the algorithm proposed by Chang and Messerschmitt [15]. The idea is that the DCT coefficients of a translated and non-aligned block can be obtained by summing weighted DCT coefficients from their four overlapping neighbor blocks. The expensive inverse DCT transform is not needed. The algorithm becomes very efficient if only part of the DCT coefficients (e.g., DC coefficients) are needed. As mentioned in Section I, after we obtain the DCT DC coefficients of P and B frames, we can apply the first two stages of our face detection algorithm to these video frames.

Figure 1 shows the block diagram of our face detection algorithm. In the diagram, rounded rectangles represent input data, intermediate and final results; rectangles represent operations in the algorithm. The algorithm has three stages, where average chrominance of macroblocks, shape constraints on human faces, and energy distribution of the DCT coefficients are used respectively. MPEG macroblock (16x16 pixels) is our processing unit, so that the bounding rectangles of the detected face regions have a resolution limited by the size of the macroblock. The result of the algorithm is a list of face regions and their locations in the video.

The algorithm also uses domain knowledge to help make decisions at each stage. Domain knowledge is shown as ellipses in the diagram. Statistics of human skin-tone colors in the chrominance plane is used in Stage 1. We use sample patches of face regions and non-face regions as training data to generate the statistics. Shape constraints on human faces are applied in Stage 2. One of them is the anatomical constraint

of human faces. For example, it is impossible for the outline of a human face to have an aspect ratio (height over width) of 3 to1, or 1 to 3, if we do not consider face regions in video created by special effects. Other constraints are from the attributes of MPEG video. For example, the size of the video frames sets the upper bound of the largest face regions that our algorithm can detect. In Stage 3, knowledge of the energy distribution over the DCT coefficients of face regions is used.

In the first stage of the algorithm, DCT DC values of Cb and Cr blocks are used to represent average chrominance of the corresponding macroblocks. According to its average Cb and Cr values, each macroblock is classified as a skin-tone macroblock or the opposite, based on the statistical distribution of skin-tone colors in the chrominance plane. After Stage 1, all macroblocks with skin-tone colors are considered as candidate face regions. Therefore, a binary mask image can be generated for each frame, in which a "one" means a candidate face macroblock, and a "zero" means the opposite. The binary mask image is post-processed by morphological operations to eliminate noise and fill up holes in it.

It should be noted that the goal of the first stage is to detect *all* candidate face blocks. A moderate level of false alarms is tolerable at this stage. Additional constraints in later stages can be used to greatly reduce the false alarm rate.

In the second stage, our goal is to detect face regions in the mask images generated by Stage 1. We use a projection method to quickly find the target areas, then apply an iterative template matching procedure to locate the individual candidate face regions. To cope with various sizes and orientations of faces, shape constraints are used to eliminate false alarms of face regions.

In the final stage, for each face region detected in Stage 2, we calculate the energy distribution of the luminance DCT coefficients over different frequency bands in the DCT domain. This is based on the observation that human faces contain uneven frequency components in different orientations. The result is used as a final verification of face detection and helps eliminate anomalies in Stage 2. Note that this stage can be omitted for P and B frames in order to save computation of DCT AC coefficients in these types of frames.

Overall, our algorithm has a cascading structure with various kinds of domain knowledge applied. To speed up the algorithm, our principle is to push simpler stages up to the beginning, and leave the most complex ones to the end. Thus, more complex stages only have to work on a subset of the original data so that computation is reduced.

III. Stage One: Macroblock Classification Based on Chrominance

In the first stage of the algorithm, we check each macroblock of the video frame to see if it is a candidate face macroblock or not. The key to this classification is the uniqueness of human skin-tone colors. We use training data to generate skin-tone color statistics, then apply the Bayesian minimum risk decision rule to classify each macroblock.

A. Human Skin-Tone Characteristics

In video transmission and storage, colors are usually separated into luminance and chrominance components to exploit the fact that human eyes are less sensitive to chrominance variations. Psychophysical experiments indicate that perception of colors has three attributes: hue, saturation, and intensity [16]. Intensity corresponds to the luminance value (Y), while hue and saturation are kept in the chrominance components (such as Cr and Cb).

Human skin tones form a special category of colors, distinctive from the colors of most other natural objects. Although skin colors differ from person to person, and race to race, they are distributed over a very small area on the chrominance plane. This means that skin colors are relatively consistent in hue and saturation. The major difference between skin tones is intensity.

The above fact has been noticed and used by researchers in consumer electronics to design TV circuits that automatically detect and correct human skin-tone colors that are sensitive to human eyes [17], [18]. I and Q components of NTSC chrominance signals are used to estimate the hue and saturation of a color. Colors are classified as skin tones if their hue and saturation fall into certain ranges. By taking out the luminance component of colors, the difference between skin colors of different races and the effect of lighting conditions are reduced.

In our work, we make use of human skin-tone characteristics as well, but take a different approach. First, we use Cb and Cr, instead of I and Q, because the former are usually the chrominance components used in MPEG video. Second, we generate statistics of skin-tone color distribution directly on the Cb-Cr plane and use it in the classification process. Thus, we can get more accurate statistics and avoid the non-linear transform to get hue and saturation. Third, to enhance the robustness and flexibility of our algorithm, we classify the colors based on the Bayesian decision rules.

B. Generating Skin-Tone Color Statistics

Figure 2(a) shows the distribution of all displayable colors in the RGB color cube on the Cr-Cb chrominance plane. R, G, B values are normalize to [0, 1]. The conversion between (R, G, B) and (Y, Cb, Cr) is given as follows [14]:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.169 & -0.331 & 0.500 \\ 0.500 & -0.419 & -0.081 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} \qquad (1)$$

Note that Figure 2(a) is for illustration purposes only. One point in the hexagon actually corresponds to all colors that have the same chrominance value but different intensity levels. For example, all gray level colors from black to white are mapped to the origin (W) in Figure 2(a). Letters B, M, R, Y, G, and C in the figure correspond to the colors with hues of blue, magenta, red, yellow, green, and cyan, respectively.

To generate skin-tone color statistics on the Cb-Cr plane, we use as training data sample face patches of different races, which are cut from video frames. Figure 2(b) shows the distribution of skin-tone colors in the Cr-Cb plane, based on over forty sample face patches. The white area in Figure 2(b) corresponds to chrominance values that have a non-zero probability in the sample face patches. Compared with Figure 2(a), it is clear that skin-tone colors are distributed over a very small area in the chrominance plane.

C. Classification Based on the Minimum Cost Decision Rule

We use the *Bayesian decision rule for minimum cost* to classify a color into skin-tone class or non-skin-tone class. This technique is flexible because it allows us to use the statistics of skin-tone colors, and to take into consideration the different effects of false alarms and false dismissals. The Bayesian decision

rule for minimum cost [19] (for two classes) is described as follows.

$$R_0(X) = C_{00} \cdot p(\omega_0|X) + C_{10} \cdot p(\omega_1|X) \quad (2)$$

$$R_1(X) = C_{01} \cdot p(\omega_0|X) + C_{11} \cdot p(\omega_1|X) \quad (3)$$

$$R_0(X) < R_1(X) \Rightarrow X \in \omega_0 \quad (4)$$

$$R_0(X) > R_1(X) \Rightarrow X \in \omega_1 \quad (5)$$

ω_0 and ω_1 denote two classes, respectively. $p(\omega_i|X)$ denotes the *a posteriori* probability, i.e., the probability of being in class *i* given sample *X*. C_{00} and C_{11} denote the cost coefficients of correct classifications; C_{01} and C_{10} denote the cost coefficients of false classifications. Therefore, $R_i(X)$ is the "cost" of classifying an unknown sample into class *i*. The classification problem becomes finding the class which gives the minimal cost, considering different cost weightings on classification decisions.

In our context, the two classes are non-skin-tone class (ω_0) and skin-tone class (ω_1). We assign zero cost to correct classifications, so that C_{00} and C_{11} are both zero. Then the minimum cost decision rule reduces to the following.

$$C_{10}p(\omega_1|X) < C_{01}p(\omega_0|X) \Rightarrow X \in \omega_0 \quad (6)$$

$$C_{10}p(\omega_1|X) > C_{01}p(\omega_0|X) \Rightarrow X \in \omega_1 \quad (7)$$

Applying the *Bayesian Formula* (Eq. (8)) to the above equations, we obtain the decision rules used in our algorithm as shown by Eqs. (9) and (10).

$$p(\omega_i|X) = \frac{p(X|\omega_i)p(\omega_i)}{p(X)} \quad (8)$$

$$\frac{p(X|\omega_1)}{p(X|\omega_0)} < TH \Rightarrow X \in \omega_0 \quad (9)$$

$$\frac{p(X|\omega_1)}{p(X|\omega_0)} > TH \Rightarrow X \in \omega_1 \quad (10)$$

$$\text{where} \quad TH = \frac{C_{01}}{C_{10}} \cdot \frac{p(\omega_0)}{p(\omega_1)}$$

In the above equations, $p(\omega_i)$ is the corresponding *a priori* probability of class ω_i. $p(X|\omega_i)$ denotes the conditional probability density functions of skin or non-skin colors on the chrominance (Cb-Cr) plane. The conditional probabilities are generated by the method described in Section III(B) using sample face and non-face patches as training data.

TH is the adjustable decision threshold. The higher C_{10} (or the lower C_{01}) is, the more false alarms are allowed, and vice versa. By changing C_{10} and/or C_{01}, we can control the amount of false alarms and false dismissals allowed in Stage 1. In applications, TH set at 2.0 is a reasonable value, based on our experiments of the effect of TH on the classification.

Figures 3 and 4 demonstrate the effect of the decision threshold (TH) on the classification of color pixels. A video frame with complex background and multiple small faces is shown in Figure 3(a). As TH

decreases, more skin-color pixels are detected, which are shown as gray pixels in Figures 4(a), 4(b), and 4(c). For a complex scene like Figure 3(a), it is clear that a small TH value is needed to generate solid clusters of face pixels for the detection of face regions.

Figure 3(b) shows a video frame with a relatively clean background and a large human face. The result of pixel classification with TH equals 0.5 is shown in Figure 4(d). When TH is such a small value, we can detect almost all of the face pixels, with some noise in the background. Figure 5 illustrates the variations of false detection rate and false alarm rate when TH changes from 1 to 20. We use the classification result of Figure 4(d) as the basis for comparison. First, we manually segment Figure 3(b) into the face region (a bounding rectangle) and the background region. Then, in the result mask images, for each TH value, we count the number of "one" pixels in the face region, and the number of "one" pixels in the background. These numbers are compared with those for TH equals 0.5, to derive the relative false alarm rate, and relative false dismissal rate. Figure 5 shows that as we raise the classification threshold, the false alarm rate decreases. However, at the same time, we have more false dismissals. Because false dismissals are undesirable at the early stage of face detection, a TH value of 2.0 is reasonable for classification. Nevertheless, it can be changed in order to have a higher detection rate or a lower false alarm rate in applications.

Experiments show that our method based on the statistical decision rule gives more accurate results, compared with the method suggested in [18]. An example of comparison is given in Figure 6. Figures 6(a) and 6(b) are the classification results using the method in [18] and our method, respectively. Gray pixels correspond to skin-tone colors; black pixels the opposite. The original video frame is Figure 3(a). It can be seen that Figure 6(b) has more solid clusters of candidate face regions.

D. Macroblock Classification Based on Average Chrominance

We apply the above minimum cost decision rule to MPEG video streams, and classify each MPEG macroblock as a candidate face macroblock or a non-face one. We use only the DCT DC coefficients of the corresponding Cr and Cb blocks, which are equivalent to the average values (up to a scale) of the chrominance blocks in the pixel domain. Higher resolution in skin-tone detection can be achieved by taking more DCT coefficients (e.g., a few low-order coefficients) as input in the above classification process.

After the classification, we get a binary mask image for each video frame. Each value in the mask indicates the classification results of the corresponding macroblock. Then, a 3x3 (macroblocks) cross median filter is applied to the binary mask image to remove noise and smooth the image. The filtering helps because faces are connected regions, and are homogenous in chrominance. We use the video frame of Figure 3(b) as an example. Figure 7(a) shows the binary mask image after the classification by average chrominance. Figure 7(b) is the better result after applying the median filter to Figure 7(a).

IV. Stage 2: Detecting Face Regions in Macroblock Mask Images

As shown in Figure 1, after Stage 1 of the algorithm, we have a macroblock mask image for each video frame. In these macroblock mask images, a "one" pixel corresponds to a macroblock whose average color is a skin-tone color. Now our task is to scan through these mask images, and detect actual face regions in them.

Clearly, chrominance information alone is not enough to detect face regions. In a video sequence with complex scenes, besides human faces, there may be other exposed parts of the body with skin tones, and natural scenes with colors similar to skin tones (e.g., a desert scene). All these examples would produce positive yet false detections in Stage 1 of our algorithm. In Stage 2, we apply shape constraints of human

faces on the binary mask images generated by Stage 1, to eliminate these false alarms and detect candidate face regions.

A. Shape Constraints on Human Faces

Just like color, the shape of human faces is unique and consistent. The outline of a human face can be approximated by an ellipse [10], or more precisely, by connected arcs [8]. Furthermore, typical face outlines have been found to have aspect ratios in a narrow range between 1.4 and 1.6, and tilt in the range (-30^o, $+30^o$) [8].

Our face detection algorithm works on the macroblock level, so that it is difficult to use ellipse or arcs to describe face outlines. As mentioned in Section II, this is a trade-off for high speed. As an approximation, we use rectangles with certain aspect ratios as the boundary of face regions. We set the range of aspect ratios of these bounding rectangles to [1, 1.7]. It is a larger range, because we are using rectangles to estimate the aspect ratio of the actual face outline.

Besides certain aspect ratios, these rectangles are also bounded by size. The size of the video frames upper bounds the size of face regions. It is lower bounded as well, because it is generally believed in the face recognition field that 32x32 pixels is the lower limit for face detection [3]. Since we are working in the compressed domain, we set the lower limit of our face detection algorithm to 48x48 pixels, or, 3x3 macroblocks. Faces smaller than this size are not considered for detection.

In summary, the shape constraints we put on macroblock mask images are as follows: (1) faces are contiguous regions that fit well in their bounding rectangles, whether the face is front view or side view, or whether the head is upright or a little tilted; (2) the size of the bounding rectangles is lower bounded by the lower limit of face detection, and upper bounded by the size of the video frames; (3) the aspect ratios of the bounding rectangles should be in a certain range.

B. Detecting Face Regions by Binary Template Matching

Stage 1 of the proposed algorithm generates candidate face macroblocks after skin-tone classification. Compared with original video frames, the resolution of the mask images is 16 times lower in horizontal and vertical directions. Existing geometric analysis techniques, e.g., those detecting arcs corresponding to chin boundaries and hairlines [8], are not appropriate at this resolution. Note that we may detect all candidate regions back in the pixel domain and apply the above algorithms. But to keep our algorithm in the compressed domain, we take into account the resolution limit and modify the task objective. Now our task is to find contiguous regions in the macroblock mask images that can be bounded by a rectangle satisfying some size and shape constraints.

To accomplish this task, we propose a method called binary template matching. The idea is to use rectangles of possible sizes and aspect ratios as face templates to match against the binary mask images. A face template is shown in Figure 8(a), whose size is (M+1)x(N+1) macroblocks. Note that internally, the face templates are represented by only (M+1)x(N+1) pixels. For the sake of illustration, mask images are blown up in the figures by a factor of 16 in each direction. The template consists of two parts: the face region, which is the MxN shaded rectangle, and the background, which is the area between the inner and outer rectangles. The size of the face region (MxN) should be bounded by size and aspect ratio, according to our shape constraints. The reason we consider the background as part of the template is that the color of the background adjacent to the face region is usually distinctive from skin tone, so that there should be few “ones” in this region. The macroblocks adjacent to the bottom of the face region are not considered in

the template, because they can be either the exposed neck or clothes and have no definitive color characteristics.

The matching criterion is two-fold. As we slide this two-frame template over a macroblock mask image, we count both the number of ones covered in the shaded region, and the number of ones in the background region. The intuition is that for a match, the first number should be high, and the second should be low. Since we are processing binary mask images, no multiplication is involved in the matching. Therefore this procedure is of low complexity. We count the number of ones inside the face (shaded) rectangle, as well as the numbers of ones in the top, left, and right parts of the background region. Denote these numbers as N_0, N_1, N_2, and N_3, respectively. Only if N_0 is above a threshold, and N_1, N_2, N_3 are below certain thresholds, do we declare a match.

This is illustrated in Figure 8. Figure 8(b) is a match, because the face region is almost covered by ones, and there are few ones in the background region. Figure 8(c) is not a match, because the face rectangle is not covered enough by ones. Figure 8(d) is not a match either, because there are too many ones in the background region.

C. Flow Chart of Stage 2

The flow chart of Stage 2 of our algorithm is shown in Figure 9. The core of this stage is the binary template matching technique described in Section IV(B). To further speed up the processing of this stage, extra work is done to limit the search area for template matching.

For each macroblock mask image, we first segment it into non-overlapping rectangular regions that contain either no ones or a contiguous one region. This segmentation is done by projecting the mask image onto the x and y axes. Given a binary mask image B(x,y) (x=0,1,...,M-1; y = 0,1,...,N-1), the projections are defined as follows, where PX denotes the projection onto the X axis; PY denotes the projection onto the Y axis.

$$PX(x)= \sum_{y=0}^{N-1} B(x,y) \qquad \text{x=0, 1, ..., M-1} \qquad (11)$$

$$PY(y)= \sum_{x=0}^{M-1} B(x,y) \qquad \text{y=0, 1, ..., N-1} \qquad (12)$$

Based on the zero-runs and non-zero-runs in PX and PY, we are able to segment the binary mask image. As an example, Figure 10(a) is the binary mask image corresponding to a video frame with two faces in it. After the projections, we have PX=[0, 0, 0, 0, 0, 0, 5, 6, 4, 2, 0, 0, 0, 2, 3, 3], and PY=[0, 0, 0, 0, 4, 6, 6, 4, 3, 2, 0, 0, 0, 0, 0]. Using the above projection results, the binary mask image is segmented as shown in Figure 10(b). The way we segment the image guarantees that in each of the segments, there is either a contiguous one region, or no ones at all.

For each of the segments, we first count the number of ones in the segment. If the number is zero, or less than the minimum value for a possible face region, there is no need to perform binary template matching in it, so we simply proceed to the next segment, as shown in the flow chart in Figure 9. Therefore, for all the rectangular segments shown in Figure 10(b), only the two regions in Figure10(c) need to be searched for faces.

Then, in each of these segments, we apply the binary template matching method. Because the size of the face region is unknown, we start from the largest possible rectangle for each segment, then gradually reduce the template size. Therefore, all sizes of the face regions can be detected.

Finally, overlapping face bounding rectangles are resolved. If only a small area is overlapped, this may be a situation where two faces are very close to each other. Therefore, we keep both regions as valid face regions. If one of the regions is small and the overlapping area is large compared with its size, we discard the smaller rectangle.

An example is shown in Figure 11. The original video frame is in Figure 11(a). The binary mask image is in Figure 11(b), along with search regions (bounded by white rectangular frames). Figure 11(c) shows the detected face regions before Stage 3. The final result after Stage 3 is overlaid on Figure 11(a). In some cases, there might be non-face regions that have similar colors to skin tones, and have a roughly rectangular shape. This will cause false alarms, such as the rectangle in the lower-left corner of Figure 11(c). This problem can be solved in Stage 3 of our algorithm, as will be described in the following section.

V. Stage 3: Verification Based on Energy Distribution of DCT Coefficients

The main purpose of the last stage of our algorithm is to verify the face detection result generated by the first two stages, and remove false alarms caused by objects with colors similar to skin tones. Because of the existence of eyes, nose-mouth junction, and lips in face regions, there are many discontinuities of intensity level in the vertical direction of the image in face regions. These discontinuities correspond to the DCT coefficients in the high vertical frequency area. Therefore, we expect some energy in the luminance DCT coefficients (Y-component in MPEG video) in that frequency band. This is the rule we use in this stage for verification. Before calculating the energy, we have to group the 64 DCT coefficients into different frequency bands.

A. Grouping the DCT Coefficients

Various methods have been proposed in the signal processing field to classify DCT coefficients into groups of different spatial frequencies. In our algorithm, we follow the DCT coefficient grouping scheme proposed by Ho and Gersho [20] for vector quantization of transformed images, where the DCT coefficients in a 8x8 transform block are partitioned into groups corresponding to the directional features (horizontal, vertical, and diagonal edges) of the block in the spatial domain, along with the DC coefficient. Figure 12 shows the grouping of the DCT coefficients, where groups H, V, and D correspond to vertical, horizontal, and diagonal edges, respectively. In these matrices, one means the DCT coefficient at that position belongs to the group, and zero means the opposite. In our algorithm, we use only the H and V matrices, along with the DCT DC coefficients of luminance blocks.

B. Verification

Given a candidate face region of size MxN macroblocks, we compute the energy of the corresponding

luminance blocks in the DC and H, V areas as follows:

$$E = \sum_{i=0}^{MxNx4-1} \left(\sum_{m=0}^{7} \sum_{n=0}^{7} \left| DCT_i(m,n) \right|^2 \right) \tag{13}$$

$$E_{DC} = \left(\sum_{i=0}^{MxNx4-1} \left| DCT_i(0,0) \right|^2 \right) / E \tag{14}$$

$$E_H = \left(\sum_{i=0}^{MxNx4-1} \left(\sum_{m=0}^{7} \sum_{n=0}^{7} \left| DCT_i(m,n) \cdot H(m,n) \right|^2 \right) \right) / E \tag{15}$$

$$E_V = \left(\sum_{i=0}^{MxNx4-1} \left(\sum_{m=0}^{7} \sum_{n=0}^{7} \left| DCT_i(m,n) \cdot V(m,n) \right|^2 \right) \right) / E \tag{16}$$

Eq. (13) shows that E is the summation of the energy of all the DCT coefficients in the candidate face region. It equals the energy of the pixel values of this face region, because of the DCT transform's energy-conserving property. E_{DC}, E_H, E_V are the normalized energies of all the DCT coefficients in the candidate region, of groups DC, H, and V, respectively. Note that we assume 4:2:0 macroblock structure for the MPEG video, so that in a region of MxN macroblocks, there are MxNx4 DCT luminance blocks. Matrices H and V are given as in Figure 12.

We set up two thresholds T_{DC} and $T_{V/H}$. If either $E_{DC} > T_{DC}$, or $E_V/E_H < T_{V/H}$, we declare the face region is a false alarm. The reason is that a face region should not have near 100% energy in DC value, because face regions contain details and edges. Also the energy corresponding to the horizontal edges (E_V) should be large enough. We use E_V/E_H instead of absolute Ev, because face regions may have different resolutions in an unconstrained video stream, so that E_V may have a large dynamic range. Therefore, the ratio of E_V/E_H is more consistent and reliable than the absolute value of E_V.

Using these thresholds, we verify each candidate face region declared by Stage 2 of our algorithm. This helps remove some false alarms from Stage 2. For example, the false detection in Figure 11(c) is removed after the verification, and does not appear in the final detection result (Figure 11(a)). The effect of the T_{DC} and $T_{V/H}$ on the performance of our algorithm will be discussed in Section VI.

VI. Face Region Detection in Inter-Coded Frames

In MPEG video streams, I frames are intra-coded. After minimal parsing of an MPEG stream, all the DCT coefficients of an I frame are available for its luminance and chrominance blocks. Thus, our algorithm can be applied directly to MPEG I frames, using the DCT DCs of chrominance blocks and the DCT coefficients of luminance blocks.

P frames consist of motion compensated (MC) macroblocks and intra-coded macroblocks. For an intra-coded macroblock, all its DCT DCs can be obtained directly. An MC macroblock is coded using a motion vector and the DCT transformed residue errors. Each macroblock consists of four luminance blocks and two chrominance blocks (for 4:2:0 chrominance format), namely Cb and Cr blocks. Using partial inverse motion compensation, the DCT DC values of P frames can be computed efficiently, without the inverse DCT transform. Here we use the efficient method proposed in [21].

To apply our algorithm, we need to compute the DCT DC coefficients of Cb, Cr blocks in P frames. In

practical videos, variance within each chrominance block is small. Thus, the DCT DC of a MC block in P frame can be approximated by taking the area weighted average of the four blocks in the previous reference frame pointed to by the motion vector [21]. This is shown by Eq. (17) and Figure 13, where *x* and *y*

$$b = [b_0*(8-x)*(8-y) + b_1*x*(8-y) + b_2*(8-x)*y+b_3*x*y] / 64 + b_{error} \quad (17)$$

are the horizontal and vertical components of the motion vector, modulo block size 8; b_0, b_1, b_2, and b_3 are the DCT DC coefficients of the four neighboring blocks pointed to by the motion vector; b_{error} is the DCT DC value of the MC residue error of the block to be computed; *b* is the inverse motion compensated DCT DC value. The layout of the blocks and the motion vector are illustrated in Figure 13.

Note that motion estimation is usually done on luminance macroblocks. The motion vectors for the chrominance blocks has to be adjusted according to the encoding chrominance format. For example, in a 4:2:0 chrominance scheme, the motion vector has to be reduced by half before using Eq. (17).

Inverse motion compensation can be used to reconstruct DCT AC coefficients in P frames as well. However, this is more expensive than just computing the DCT DCs. Therefore, for P frames, we use only the first two stages of our algorithm to detect face regions. This will not affect the detection rate. However, more false alarms are expected because of the omission of the last verification stage.

B frames in MPEG streams are similar to P frames, except that they are bi-directionally motion compensated. For each MC macroblock, either forward MC, or backward MC, or both are used. The technique above can still be applied.

VII. Experimental Results and Discussion

A. The Test Sets

We use 100 I-frames from a 10-minute MPEG-compressed CNN news video as our first test set. These frames include anchorperson scenes, news stories, interviews, and commercials which cover video scenes with various complexities. The number of each category of frames is shown in Table 1 to briefly summarize the content of this test set. The size of each frame is 352x240 pixels. 4:2:0 macroblock format is used, which means that chrominance signals are subsampled by two, both horizontally and vertically.

There are 91 faces in these frames, including frontal views, semi-frontal views, side views, and tilted faces. These faces are often in complex scenes. Some of the frames have multiple faces. A detailed description of the faces is shown in Table 2.

We also use 50 P frames from the same news video as test set 2, to test our face detection algorithm in inter-coded frames. There are 44 faces in these P frames.

Finally, we use another 100 I frames from other CNN news clips as test set 3. It includes anchorperson scenes (with different anchors from those in test set 1), two news stories, and one out-door interview. There are 46 faces in these frames.

B. Speed

The run time of our algorithm varies depending on the content of the video frames. Stage 1 of the algo-

rithm treats all incoming video frames equally, and is run on each macroblock. So run time for Stage 1 is fixed. The difference is in Stages 2 and 3.

The run time of Stage 2 depends on the output of Stage 1. If after Stage 1, very few ones are detected in the macroblock mask image, then little binary template matching is involved, so less time is needed. On the contrary, if the video scene contains many skin-tone color regions, and is complex, Stage 2 will spend a longer time.

The run time of Stage 3 is proportional to the area of face regions detected in Stage 2, because, for each macroblock, we have to calculate its energy distribution in the DCT domain.

Overall, based on experiments, the speed of our algorithm is as follows. On a SPARC 5 workstation, the average run time is 32.6 milliseconds for the 100 I frames in test set 1. On a SGI Indigo 2 workstation, the average run time is 15.6 milliseconds. We measure the run time of the algorithm using C library functions of timing with the precision of microseconds. No inverse quantization is involved. Figure 14 shows the histogram of run time of our algorithm on the 100 frame test set 1. The run times on both SPARC and SGI workstations are given.

For P frames, since we only carry out the first two stages of the algorithm, the run time is shorter. Based on the experiments on the 50 P frames in test set 2, the average run time is 13.4 milliseconds on a SPARC 5 workstation; 7.0 milliseconds on a SGI Indigo2 workstation.

C. Accuracy

For test set 1, our algorithm detects 84 of the faces (92%), including faces of different sizes, frontal and side-view faces, etc. Detected face regions are marked by white rectangular frames overlaid on the original video frames. There are 8 false alarms in our experiment on test set 1.

For test set 2 (P frames), the algorithm detects 39 of the faces (88%), with 15 false alarms. The false alarm rate is higher because Stage 3 of the algorithm is skipped for inter-coded frames. However, as most face regions last longer than the duration of a GOP, and continue to appear in successive frames, analysis of continuation over time can be used to improve the accuracy of face detection in P and B frames. For test set 3, the algorithm detects 39 of the faces (85%), with 6 false alarms.

Figure 15 shows some examples of the result of our face detection algorithm, from all of the test sets. These examples include frames with one or more faces of different sizes, frontal and side-view faces, and a frame without a face (Figure 15(g)). No face region is detected for Figure 15(g), although there are exposed arm and hand areas that have skin-tone colors.

As mentioned in Section V, the thresholds used in Stage 3 affect the performance of the algorithm. In our experiment, we set T_{DC}=90%, $T_{B/H}$=0.5. Without Stage 3, our detection rate would be the same, but the number of false alarms would increase to 18 in the case of test set 1. Figure 16 shows the effect of T_{DC} and $T_{V/H}$ on the number of false alarms and false dismissals in test set 1, where the thresholds are applied separately. In the real algorithm, the two thresholds are combined to achieve a better performance.

Figure 16 shows that, as we raise the T_{DC}, the number of false alarms will increase, and the number of false dismissals will decrease. If we raise $T_{V/H}$, the opposite will happen. Note that in Figure 16, the number of false alarms corresponds to that of the algorithm, while the number of false dismissals corresponds

to that in Stage 3 only. The false dismissals in the first two stages of the algorithm cannot be recovered by Stage 3. From Figure 16 we also see that the thresholds we use for the algorithm are reasonable.

D. Restrictions and Discussions

The algorithm is restricted in several aspects. It can only be applied to color images and videos, because of the use of chrominance information in Stage 1 of the algorithm. The smallest faces that are detectable by this algorithm are about 48 by 48 pixels (3 by 3 macroblocks), bounded by the lower limit of machine detection of faces, and the fact that the algorithm stays in the compressed domain. The algorithm is relatively independent of lighting conditions, but very poor lighting conditions still cause false dismissals.

False dismissals cannot be totally avoided, especially in very cluttered scenes with many small faces (e.g., a scene in a football game). This situation would be alleviated if we use video sequences with larger frame sizes. False alarms usually happen because of the existence of regions that have skin-tone colors, but which are not human faces, for example: desert/soil scene, yellowish and reddish light, etc. There are fewer false alarms after we apply the shape and energy constraints. By adjusting the thresholds in Stages 1 and 3 of our algorithm, we can make face detection more or less conservative, i.e., to have higher detection rate with higher false alarm rate, or the opposite.

Despite its restrictions, our compressed-domain approach is efficient and can be applied to large video databases for indexing and recognition. It helps focus our attention to only a small portion of the entire video sequences and frames. Once we detect these target regions, we can decode them back to the pixel domain, in which more sophisticated techniques can be applied, to either enhance or verify our face detection results, or apply video indexing or face recognition techniques.

The algorithm does not give the exact outlines of the faces, because we avoid inverse DCT transform and work at the macroblock resolution. The positions of the faces detected are sometimes not perfectly aligned, because the face rectangles we detect lie on the borders of 16x16 macroblocks. This can be improved if the compressed video sequence has a format with more chrominance information, e.g., the 4:2:2 format, so that we can work on 8x8 blocks and the face detection result will be improved. Also, we can use more DCT coefficients rather than just the DC coefficient of the Cb and Cr blocks, to get more accurate results.

VIII. Conclusion and Future Work

We propose and demonstrate a fast algorithm to detect face regions directly in MPEG video streams. The speed of the algorithm on a SPARC 5 workstation is faster than real time (less than 33 ms for 352x240 frames, or 30 frames per second). Experiments also show a high accuracy (85-92% detection rate) for video streams without constraints on the number, size, and orientation of faces.

In future work we will consider other information derived from MPEG video (such as object motion, scene changes, etc.), as well as domain knowledge. Some decoding will be performed for more accurate face detection, video indexing, or sophisticated face recognition.

IX. Acknowledgements

We thank Dr. Harold Stone of NEC for his valuable comments. We also thank CNN for the permission to use the test video in our experiments. This work was supported in part by NEC Research Institute, the National Science Foundation under a CAREER award (IRI-9501266), and Columbia's ADVENT indus-

trial partnership project.

X. REFERENCES

[1] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, P. Yanker, C. Faloutsos, and G. Taubin, "The QBIC project: querying images by content using color, texture and shape", *SPIE 1993 International Symposium on Electronic Imaging: Science and Technology, Conference 1908, Storage and Retrieval for Image and Video databases*, February 1993.

[2] S.-F. Chang and J. R. Smith, "Extracting multi-dimensional signal features for content-based visual query," *SPIE Symposium on Visual Communications and Signal Processing*, May 1995.

[3] A. Samal and P. A. Iyengar, "Automatic recognition and analysis of human faces and facial expressions: A survey," *Pattern Recognition*, Vol. 25, No. 1, pp. 65-77, 1992.

[4] G. Cottrell and M. Fleming, "Face recognition using unsupervised feature extraction," *Proceedings of International Neural Network Conference*, 1990.

[5] A. L. Yuille, "Deformable templates for face recognition," *Journal of Cognitive Neuroscience*, Vol. 3, No. 1, pp. 59-70, 1991.

[6] M. Turk and A. Pentland, "Eigenfaces for recognition," *Journal of Cognitive Neuroscience*, Vol. 3, No. 1, pp. 71-86, 1991.

[7] R. Brunelli and T. Poggio, "Face recognition: features versus templates," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 15, No. 10, pp. 1042-1052, October 1993.

[8] V. Govindaraju, R. K. Srihari, and D. B. Sher, "A computational Model for Face Location," *Proceedings of the Third International Conference on Computer Vision*, pp. 718-721, 1990.

[9] M. C. Burl, T. K. Leung, and P. Perona, "Face localization via shape statistics," *International Workshop on Automatic Face and Gesture Recognition*, June 1995.

[10] A. Eleftheriadis and A. Jacquin, "Model-assisted coding of video teleconferencing sequences at low bit rates," *Proceedings of IEEE International Symposium on Circuits and Systems*, pp. 3.177-3.180, May-June 1994.

[11] G. Yang and T. S. Huang, "Human face detection in a complex background," *Pattern Recognition*, Vol. 27, No. 1, pp. 53-63, 1994.

[12] H. Rowley, S. Baluja, and T. Kanade, "Human face detection in visual scenes," Carnegie Mellon University Computer Science Technical Report, CMU-CS-95-158R.

[13] M. Venkatraman and V. Govindaraju, "Zero-crossings of a nonorthogonal wavelet transform for complex object location," *Proceedings of the IEEE International Conference on Image Processing*, October 1995.

[14] ISO/IEC 13818 - 2 Committee Draft (MPEG-2).

[15] S.-F. Chang and D. G. Messerschmitt, "Manipulation and compositing of MC-DCT compressed video," *IEEE Journal of Selected Areas in Communications, Special Issue on Intelligent Signal Processing*, January 1995, pp. 1-11.

[16] A. N. Netravali and B. G. Haskell, *Digital Pictures: Representation, Compression, and Standards*, 2nd Ed., Plenum Press, 1995.

[17] L. A. Harwood, "A chrominance demodulator IC with dynamic flesh correction," *IEEE Transactions on Consumer Electronics*, Vol. CE-22, pp. 111-117, February 1976.

[18] T. Rzeszewski, "A novel automatic hue control system," *IEEE Transactions on Consumer Electronics*, Vol. CE-21, pp. 155-162, May 1975.

[19] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd Ed., Academic Press, Inc., 1990.

[20] Y. S. Ho and A. Gersho, "Classified transform coding of images using vector quantization," *IEEE International Conference on ASSP*, pp. 1890-1893, May 1989.

[21] J. Meng, Y. Juan, and S.-F. Chang, "Scene change detection in a MPEG compressed video sequence," *Proceeding of SPIE*, Vol. 2419, pp. 14-25, 1995.

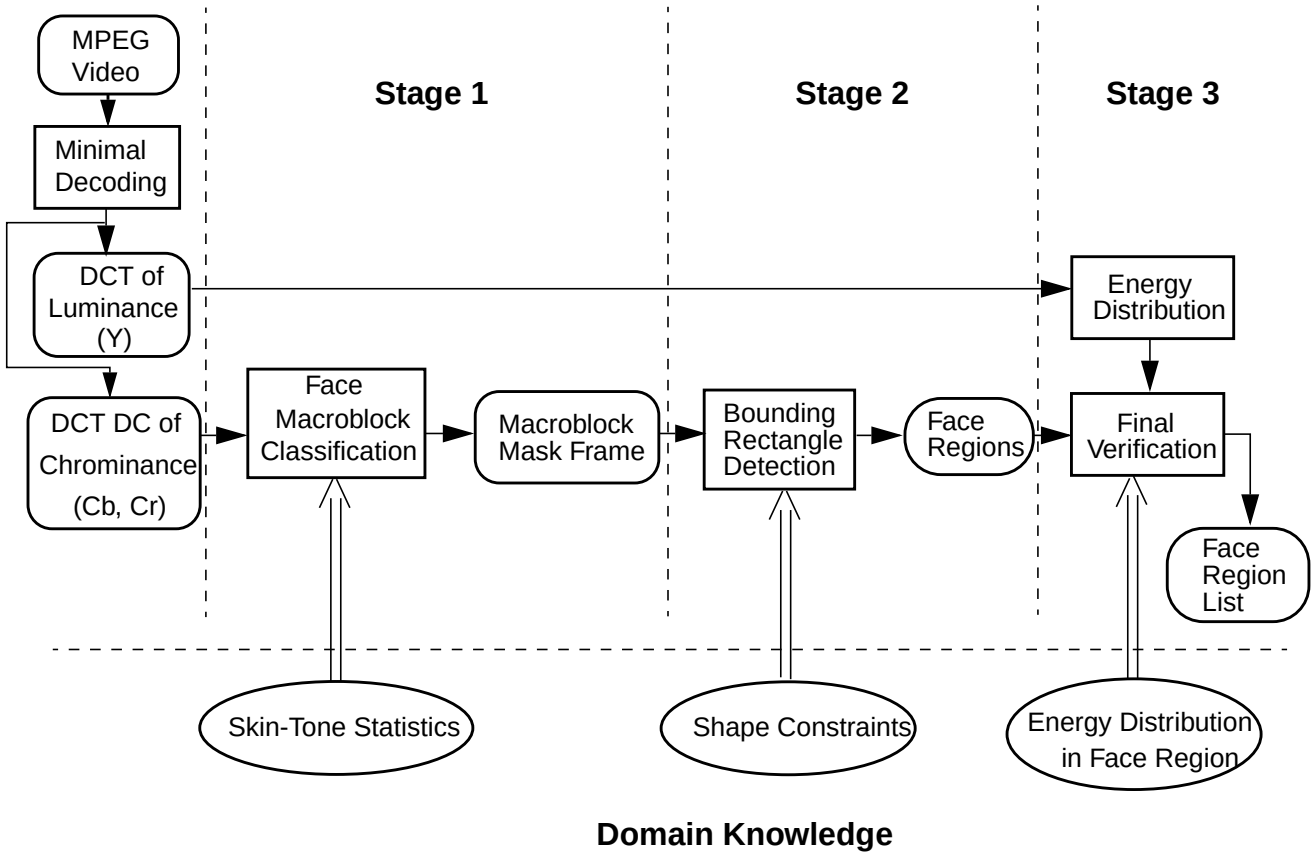


Figure 1: Block diagram of the proposed face detection algorithm.

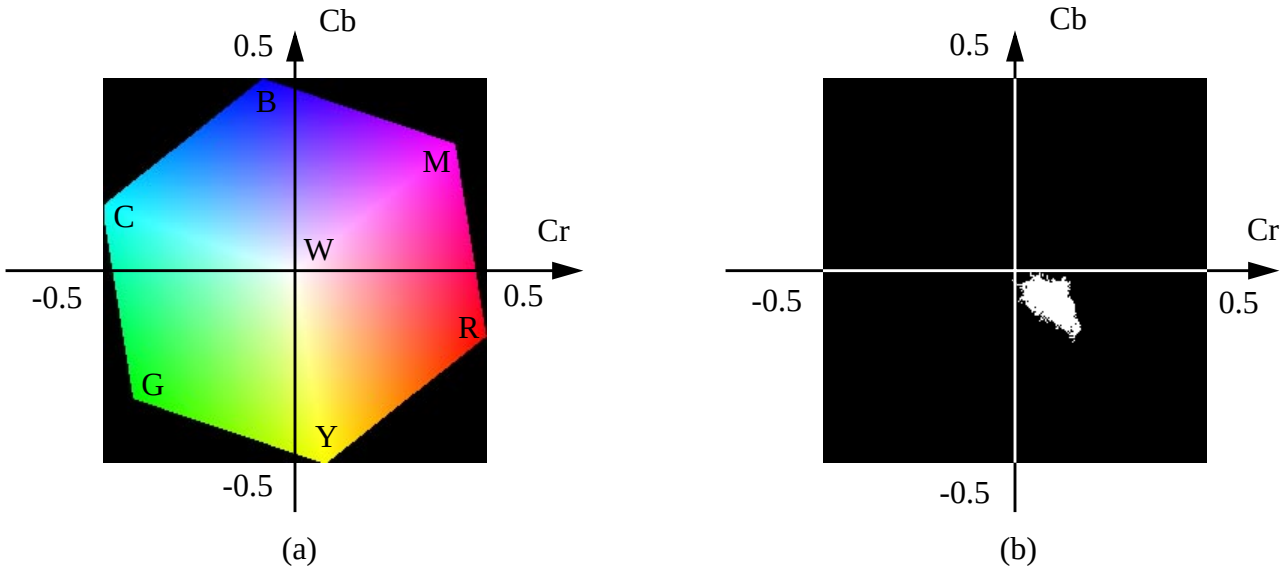
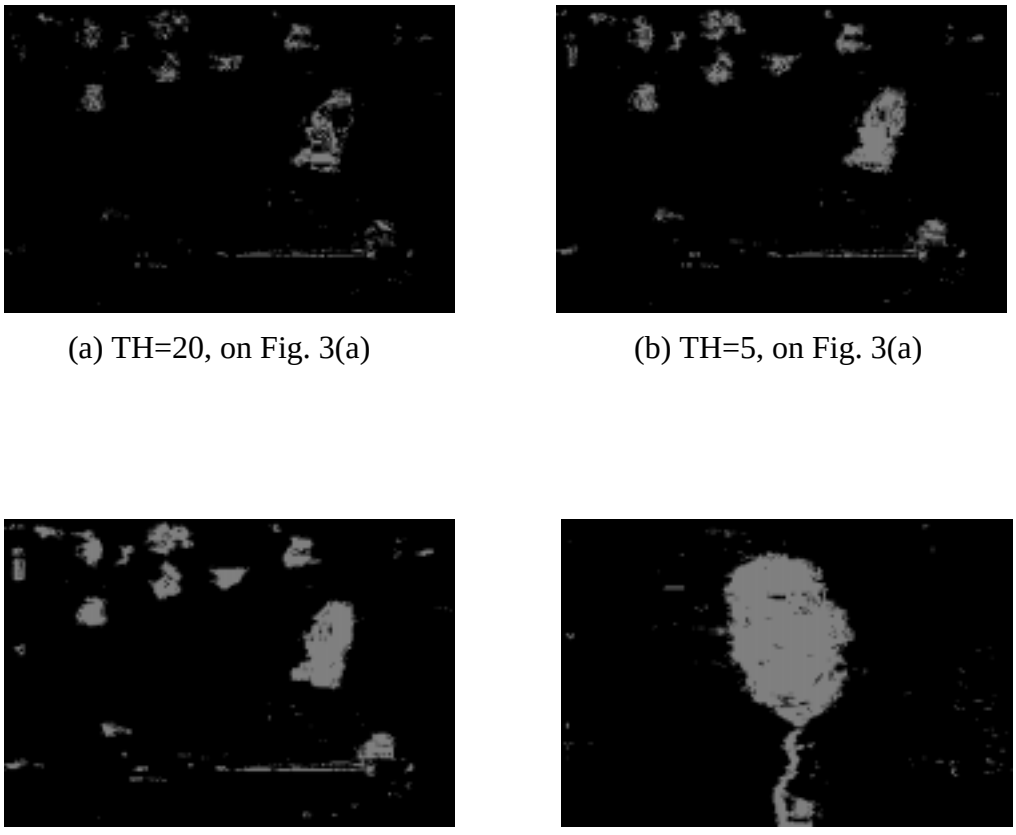


Figure 2: Color distributions on Cr-Cb chrominance plane: (a) all RGB colors; (b) skin-tone colors.

(a) (b)

Figure 3: Two video frames. (a) Multiple faces, complex background. (b) Single face, clean background.



(a) TH=20, on Fig. 3(a) (b) TH=5, on Fig. 3(a)

(c) TH=2, on Fig. 3(a) (d) TH=0.5, on Fig. 3(b)

Figure 4: The effect of threshold (TH in Stage 1) on color classification in the video frames in Fig. 3.

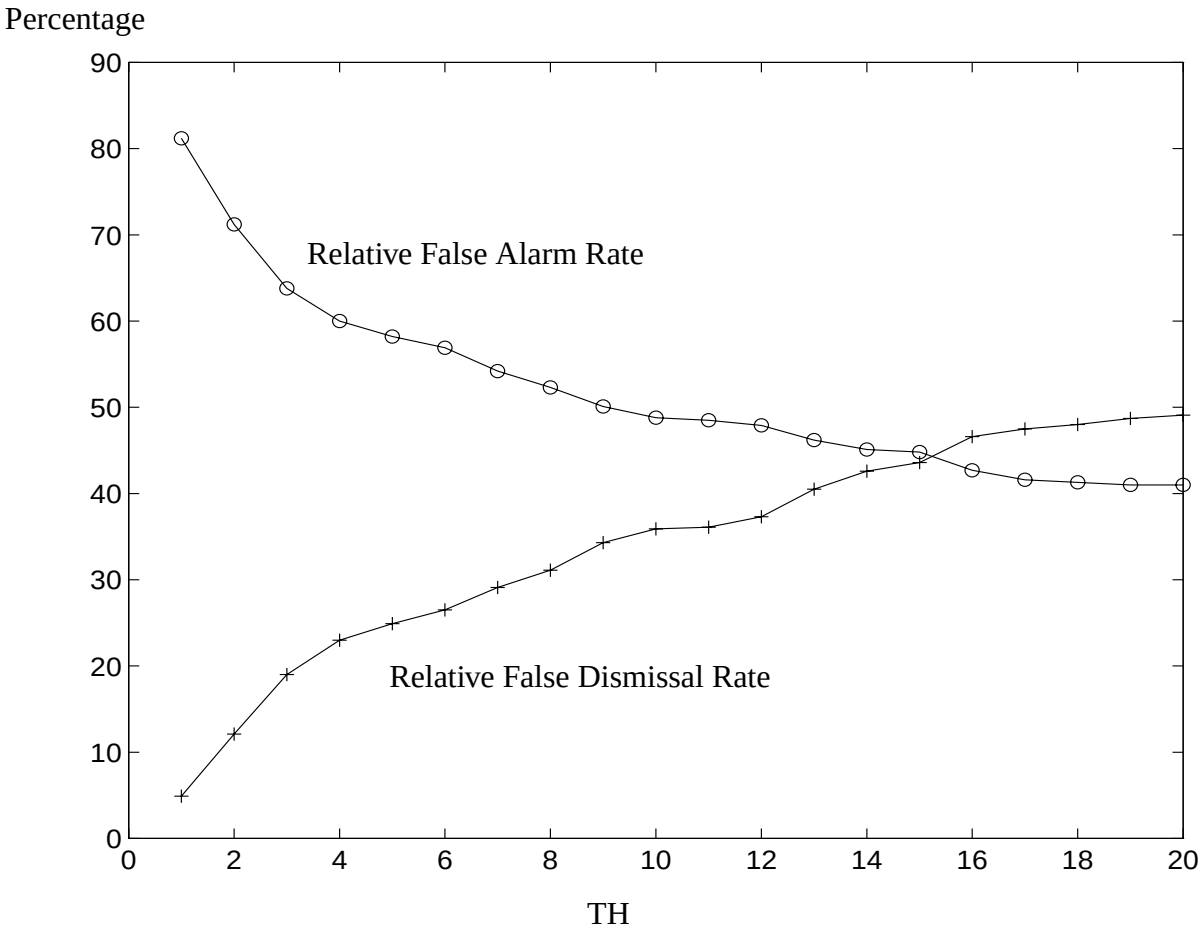
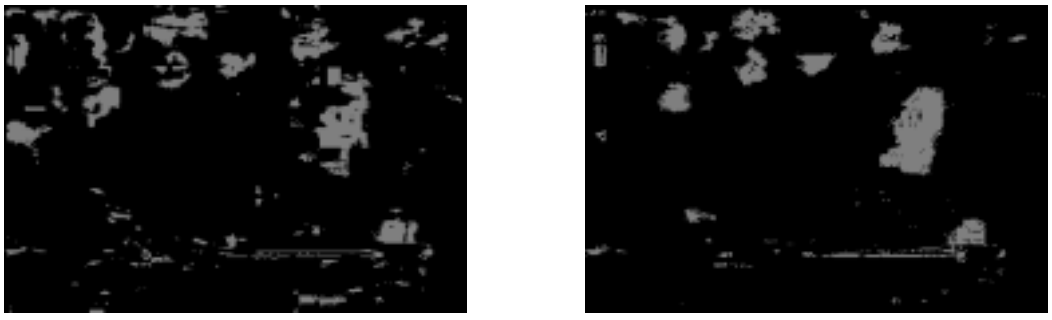
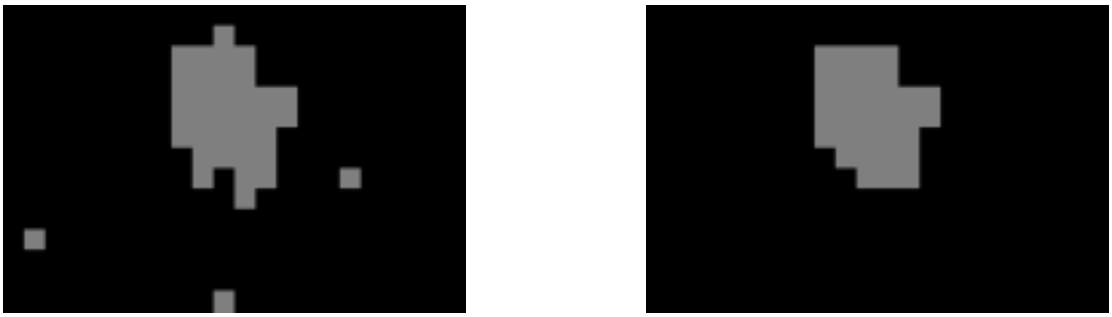


Figure 5: Performance of color classification in Fig. 3(b) over different thresholds.



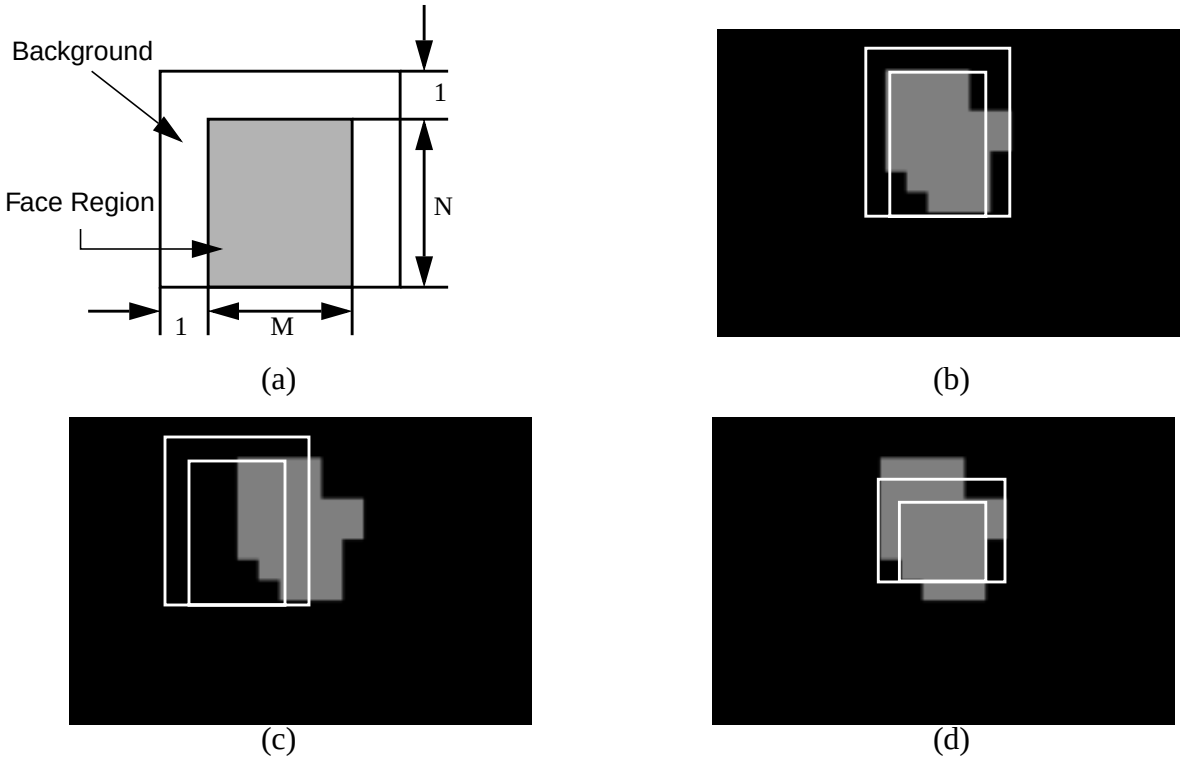
(a) (b)

Figure 6: Comparison of skin-tone classification methods: (a) using the method in [18]; (b) using our method.



(a) (b)

Figure 7: Macroblock classification: (a) the initial classification result; (b) the result after median filtering.



(a) (b)

(c) (d)

Figure 8: Binary template matching: (a) the face template; (b) a match; (c) and (d) non-match.

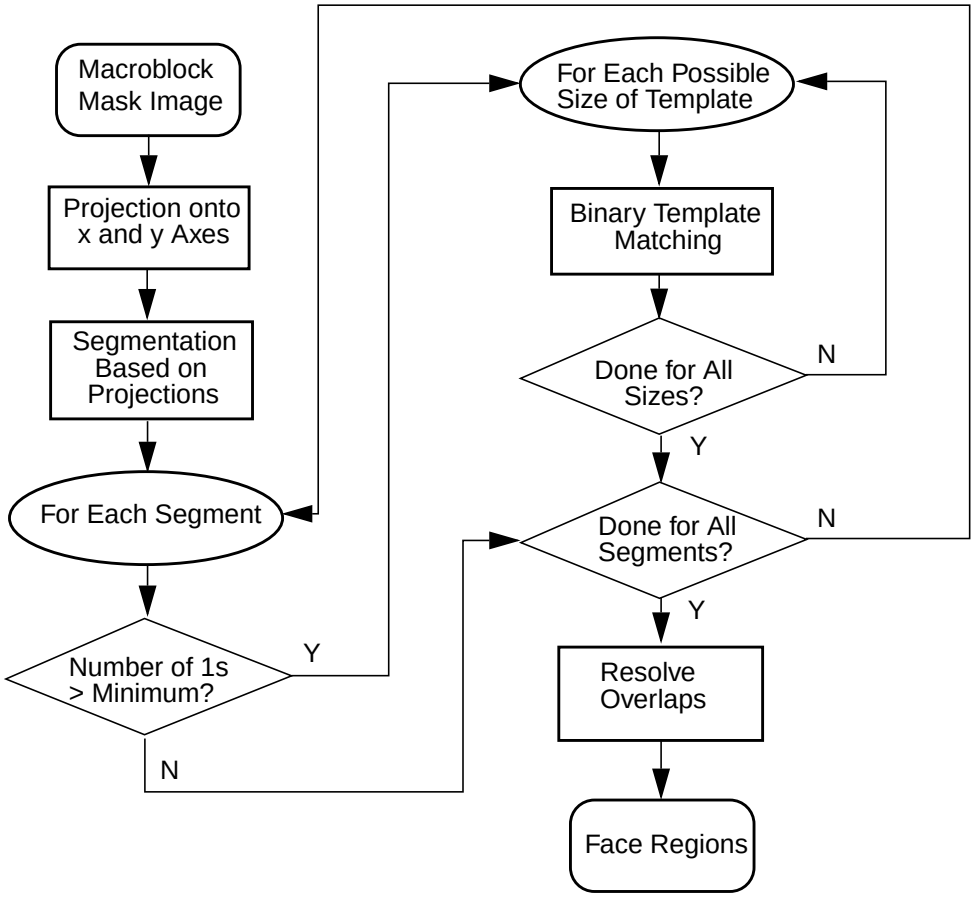
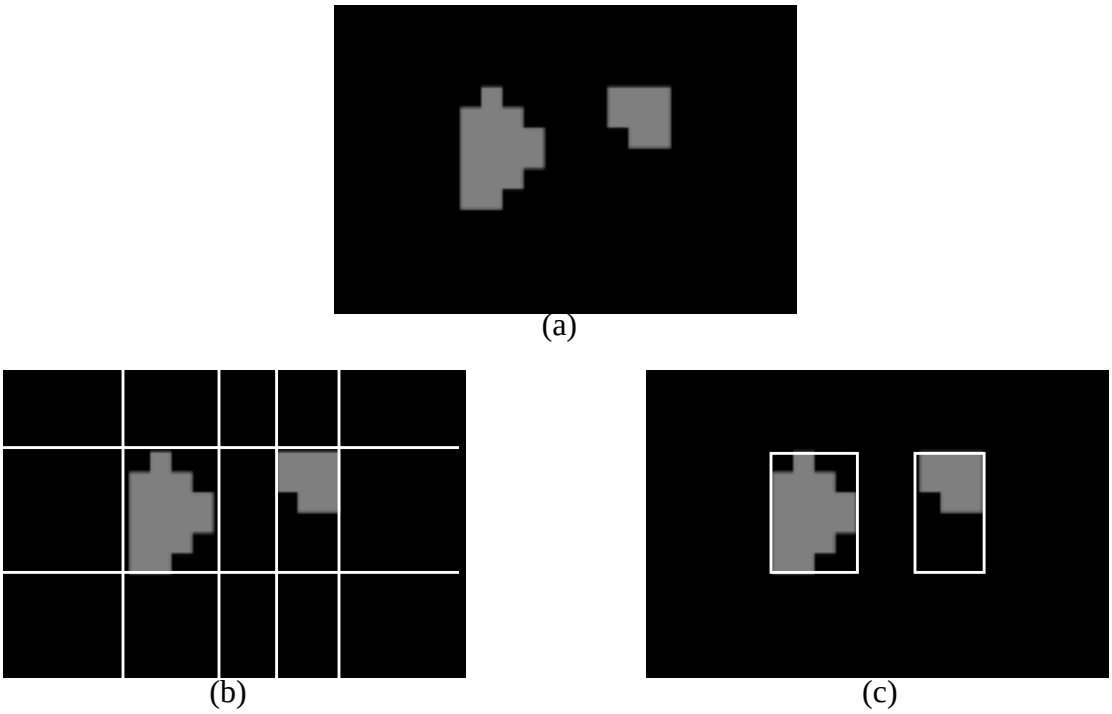
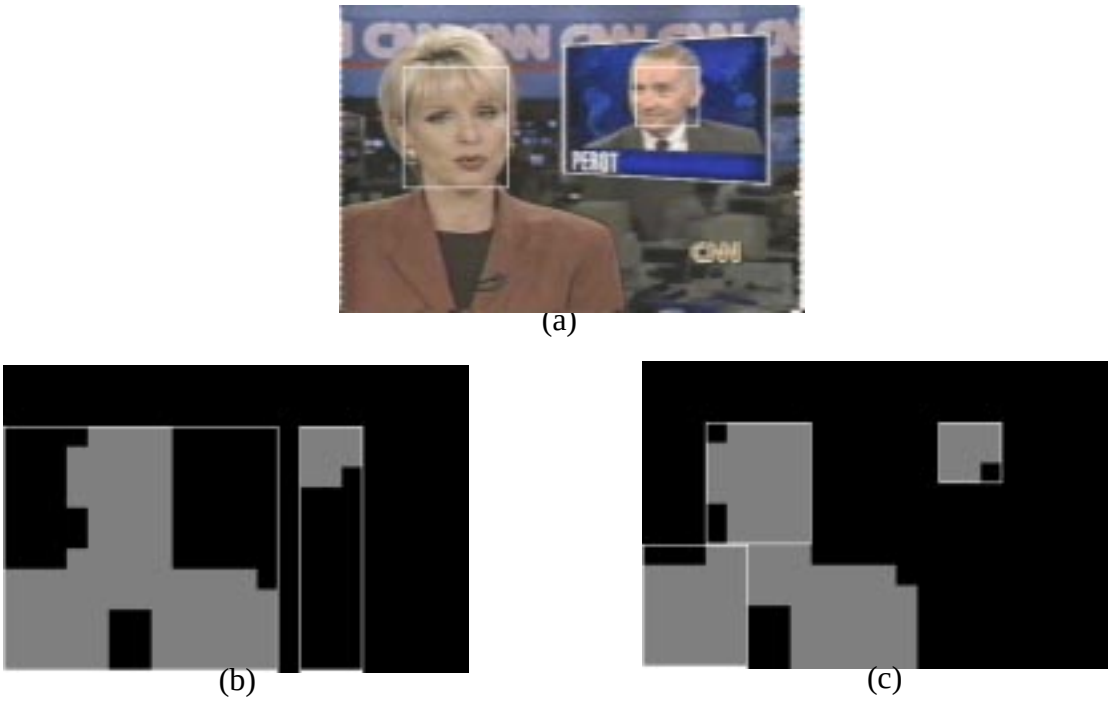


Figure 9: Flow chart of Stage 2 of the proposed algorithm.



(a) (b) (c)

Figure 10: Reduction of search area: (a) the mask image; (b) segmentation; (c) the final search area for matching.



(a) (b) (c)

Figure 11: Face region detection: (a) the video frame; (b) the mask image, with search region; (c) face regions.

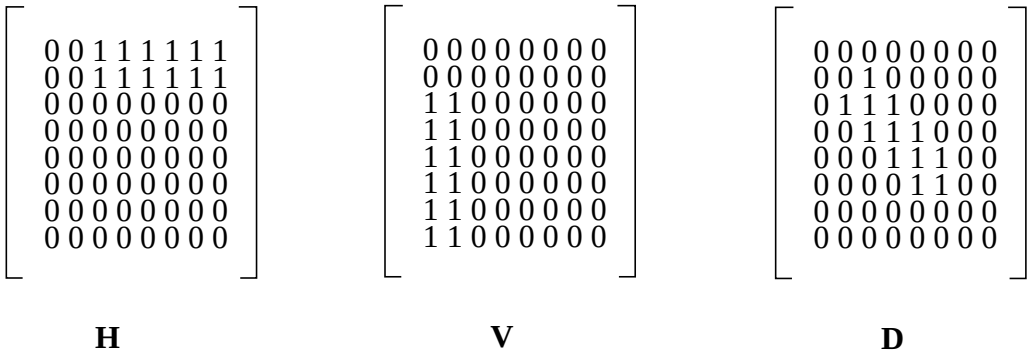


Figure 12: The grouping of the DCT coefficients.

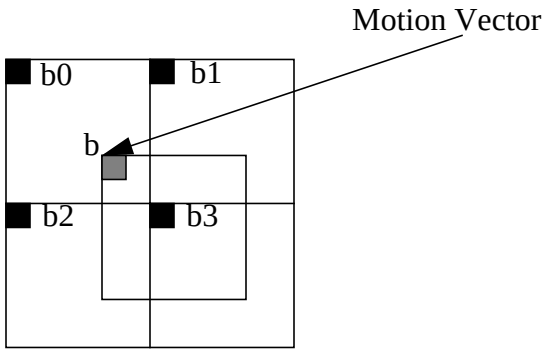
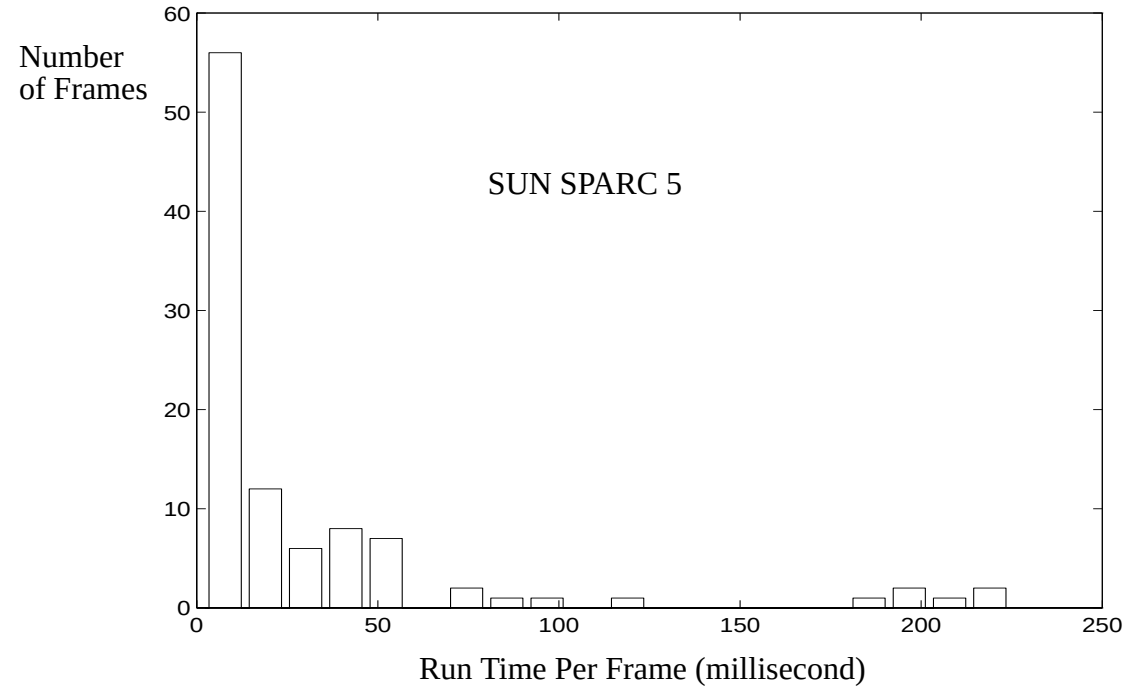


Figure 13: Inverse motion compensation of DCT DC.



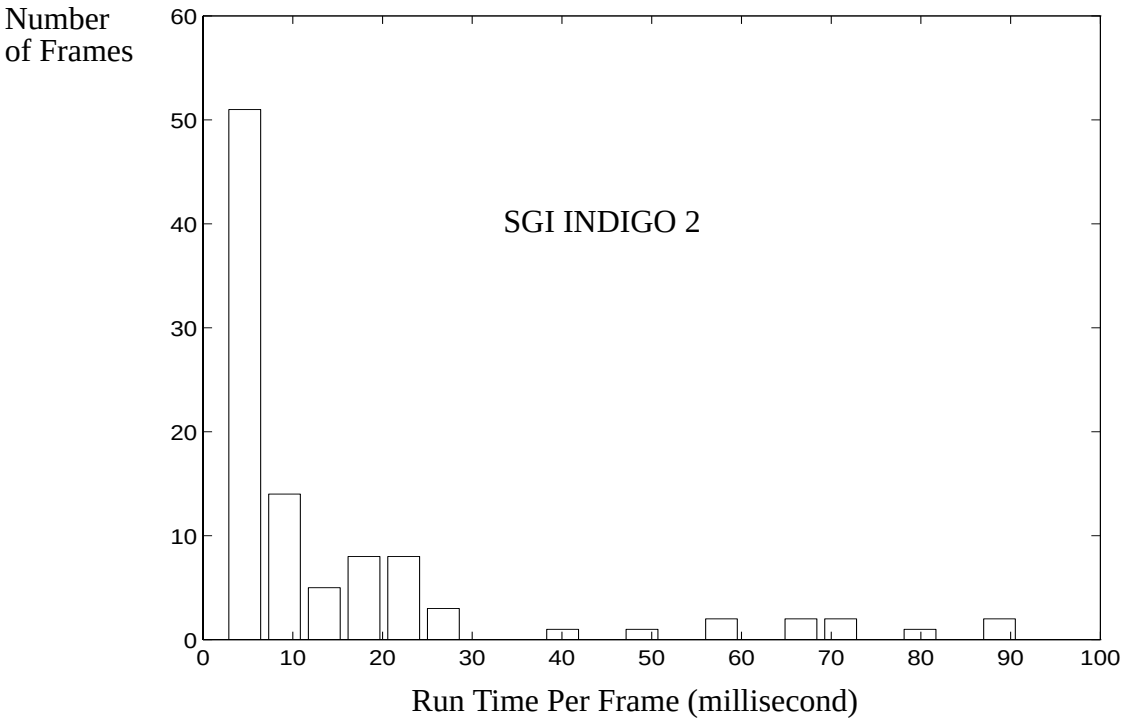


Figure 14: Run time distribution of the algorithm on the 100 frames in test set 1.

(a) (b) (c) (d) (e) (f) (g) (h)

Figure 15: Examples of the results of the proposed face detection algorithm.

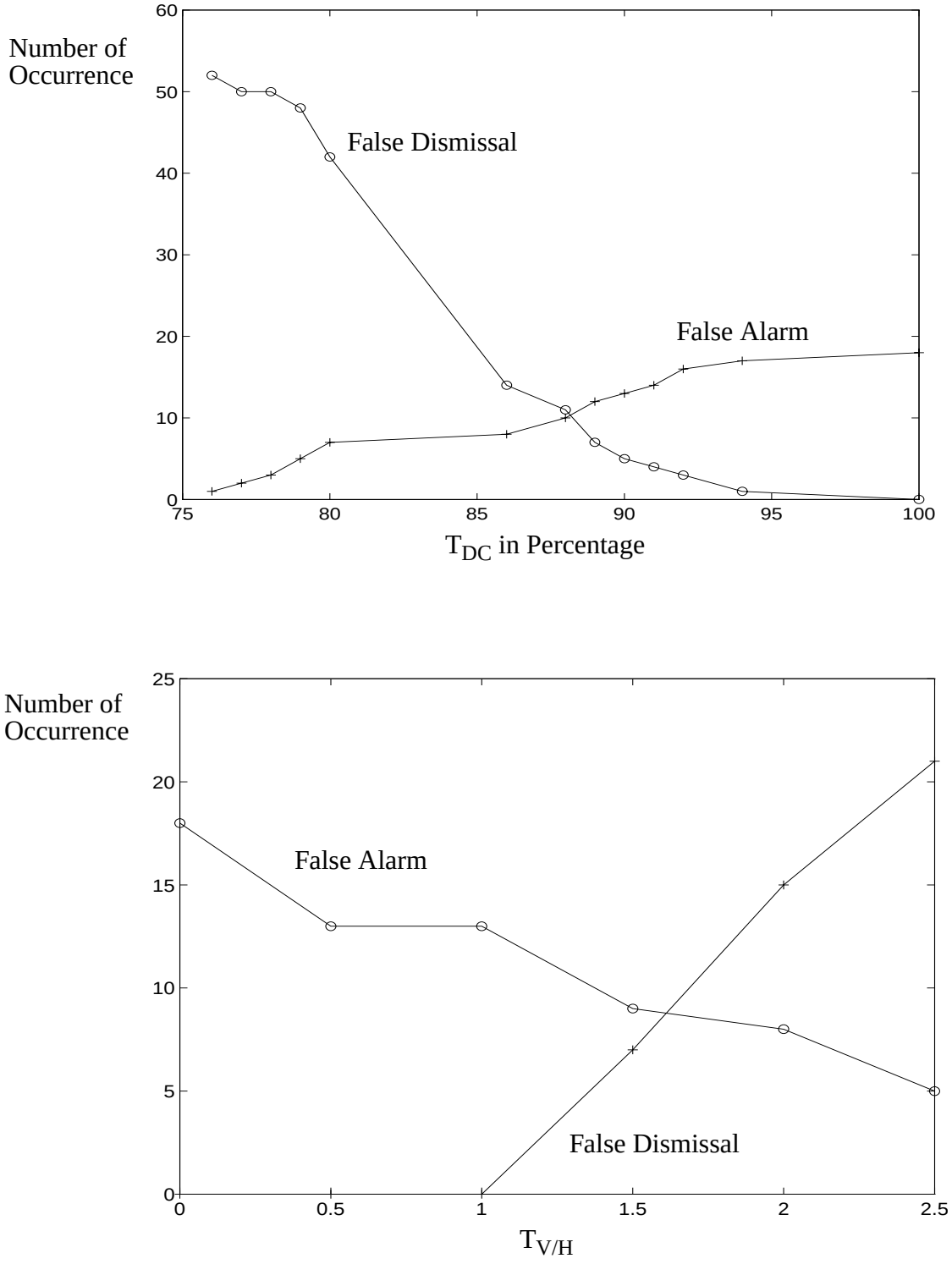


Figure 16: The effect of thresholds in Stage 3, using test set 1.

Category	Number
Anchor	15
Commercial	10
Interview	15
News Story	60
Total	100

Table 1: Number of frames in different categories, in test set 1.

Category	Number
Frontal	61
Semi-Frontal	13
Side	11
Tilted	6
In Multiple Faces	37
Number of Persons	15
Min Face Size	50x50
Max Face Size	240x240

Table 2: Statistics of the faces in test set 1.