ELSEVIER

Signal Processing: *Image Communication* 7 (1995) 435–455

SIGNAL PROCESSING: IMAGE COMMUNICATION

# Automatic face location detection for model-assisted rate control in H.261-compatible coding of video

Alexandros Eleftheriadis[a,*], Arnaud Jacquin[b]

[a]*Department of Electrical Engineering, Columbia University, New York, NY, USA*
[b]*Signal Processing Research Department, AT&T Bell Laboratories, Murray Hill, NJ, USA*

**Abstract**

We present a novel and practical way to integrate techniques from computer vision to low bit-rate coding systems for video teleconferencing applications. Our focus is to locate and track the faces and selected facial features of persons in typical head-and-shoulders video sequences, and to exploit the location information in a 'classical' video coding/decoding system. The motivation is to enable the system to encode selectively various image areas and to produce perceptually pleasing coded images where faces are sharper. We refer to this approach – a mix of classical waveform coding and model-based coding – as model-assisted coding. We propose two totally automatic algorithms which, respectively, perform the detection of a head outline, and identify an 'eyes–nose–mouth' region, both from downsampled binary thresholded edge images. The algorithms operate accurately and robustly, even in cases of significant head rotation or partial occlusion by moving objects. We show how the information about face and facial feature location can be advantageously exploited by low bit-rate waveform-based video coders. In particular, we describe a method of object-selective quantizer control in a standard coding system based on motion-compensated discrete cosine transform – CCITT's recommendation H.261. The approach is based on two novel algorithms, namely buffer rate modulation and buffer size modulation. By forcing the rate control algorithm to transfer a fraction of the total available bit-rate from the coding of the non-facial to that of the facial area, the coder produces images with better-rendered facial features, i.e. coding artefacts in the facial area are less pronounced and eye contact is preserved. The improvement was found to be perceptually significant on video sequences coded at the ISDN rate of 64 kbps, with 48 kbps for the input (color) video signal in QCIF format.

*Keywords*: Very low bit-rate coding; Facial feature detection; model-based coding; H. 261, Video compression

## 1. Introduction

In very low bit-rate video teleconferencing situations, state-of-the-art coding algorithms produce artifacts which are systematically present throughout the coded images; all the more as the image content in terms of motion and texture is high. These artifacts usually affect all areas of the image without discrimination. Viewers, however, will mostly find coding artifacts to be more noticeable in areas of particular interest to them. In particular, a user of a video teleconferencing system or video telephone will typically focus his or her attention to

* Corresponding author Address: Room 2D-337, AT&T Bell Laboratories, 600 Mountain Avenue, P.O. Box 636, Murray Hill, NJ 07974-0636, USA.

Elsevier Science B.V.
*SSDI* 0923-5965(95)00017-8

the *face(s)* of the person(s) on the screen, rather than to areas such as clothing or background. Besides, although fast motion is known to mask coding artifacts, the human visual system has the ability to *lock on* and *track* particular moving objects, such as a person's face. Communication between users of very low bit-rate video teleconferencing systems or video phones will be intelligible and pleasing only when facial features are not plagued with an excessive amount of coding artifacts.[1]

The motivation of this work was to investigate the possibility to detect specific moving objects known a priori to be present in a video sequence, and to enable a video coding system to use this information in order to encode discriminatively different areas in typical 'head-and-shoulders' video sequences – an idea which has been proposed in [30,41,32], in which, however, only parts of the problem are addressed. The coder would, for example:

- encode facial features (such as eyes, mouth, nose, etc.) very accurately;
- encode less accurately the rest of the picture, be it moving or still.

This requires that the coder first detects and models face locations, then exploits this information to achieve *model-assisted coding*. The location detection algorithm should be of fairly low complexity; in addition, if transmission of the model parameters is required, the overhead bit-rate should be minimized. In [41], Ueno et al. propose the use of face location information in a preprocessing stage which consists of low-pass filtering the image area outside the face location, without, however, disclosing how the face location detection can be automatically and reliably performed.

In this work, we show how to exploit and integrate in a novel way techniques derived from *computer vision* (scene analysis, geometric modeling, object recognition) for low bit-rate video coding. Previous work published by the authors in [14,15] proposed an automatic algorithm for face location detection, as well as the design of a model-assisted dynamic bit allocation strategy with object-selective quantization, in the context of a 3D-subband-based video coder operating at a bit-rate of 128 kbps. The coding system used in this work operates at the total audio-video ISDN rate of 64 kbps, with an input digital color video signal in YUV format, and with a coding rate of 48 kbps for the video signal. The video data consist of 'head-and-shoulders' sequences. We describe automatic algorithms for face location detection and for the detection of 'eyes–nose–mouth' regions. The former algorithm models face contours as ellipses. The latter exploits the symmetry with respect to a slanted facial axis, which is inherent to a human face appearing in a 2D projection provided that the rotation of the head is slight. We then describe two novel algorithms, *buffer rate modulation* and *buffer size modulation*, which allow exploitation of the face and facial feature location information to achieve model-assisted dynamic bit allocation. These techniques were incorporated for evaluation purposes in an 'intelligent' rate control module of an H.261 compatible codec. Their applicability, however, is not restricted to H.261, and they can be used with a variety of different coding schemes.

The organization of this paper is the following. In Section 2, we briefly review the concept of model-based video coding, and define our model-assisted coding approach. In Section 3, we describe the models adopted for the representation of face location information, describe fully automated algorithms for the detection of head outlines as well as eyes–nose–mouth regions, and show sample detection results. In Section 4 we describe a region-adaptive rate control algorithm, and discuss its application to a codec conforming to the H.261 standard. We also briefly describe the structure of the Reference Model 8 (RM8) implementation of an H.261-compliant encoder [11], which was used for subjective, comparative quality evaluation purposes of the results produced with the model-assisted encoder. Finally, Section 5 presents a summary and discussion of the results.

## 2. Model-based and model-assisted video coding

It is widely agreed upon that 'classical' (i.e. purely waveform-based) coding techniques alone may not

[1] In some situations, a very good rendition of facial features is paramount to intelligibility. The case of hearing-impaired viewers who would mostly rely on lip reading is one such example.

be sufficient for high-quality coding of digital signals at very low bit-rates – e.g. 64 kbps and below for a color video signal [43, 33, 25]. Thus, *model-based* approaches to very low bit-rate coding of video, also referred to as *knowledge-based coding*, *semantic coding*, or *analysis-synthesis coding*, have been receiving a great deal of attention [28, 19, 7, 1, 2, 35, 36, 8, 42, 23, 34, 22, 12], and are considered to have great potential [26]. For a detailed overview of state-of-the-art model-based video coding techniques, the reader is referred to [7, 1, 16].

In a generic model-based coding system, each input video frame to the encoder is analyzed, and a geometric model of the data is constructed. The model can be either two- or three-dimensional, and can be obtained either through fitting[2] or segmentation [2, 36, 23, 34, 22, 12]. The parameters of the model are transmitted on the channel along with an appropriately coded error signal. The latter is necessary in order to mitigate quality loss in regions of the image – typically complex, highly detailed ones – where the model does not give a sufficiently good fit, or simply 'fails'.

The signal is reconstructed (synthesized) at the receiving end from the model parameters and the decoded error signal. Since the bit-rate required to transmit the model parameters can be extremely low, very low coding rates can be achieved for very specific scenes, usually fairly low in texture and motion content. This approach, however, apart from its inherently very high complexity, suffers also from a lack of flexibility: the models are usually tailored to a specific sequence content. Whenever the input video data differs substantially from what can be modeled by the encoder, a model breakdown will occur with serious consequences for the coded signal.

Rather than relying on ubiquitous data models for head-and-shoulders video sequences, our approach has been to only *partially model* the data, i.e. model the *location* of specific objects known a priori to be present in the scene, and integrate this partial model to a 'classical' video coding system. For the purposes of very low bit-rate coding of video teleconferencing scenes, where typically one person is shown from the waist up moving in front of a still background, we propose to model the locations of the face[3] and selected facial features of the person present in the scene, rather than model the face itself.

This location model, which is obtained automatically and reliably, is used to *improve* in an area-selective fashion the image quality given by a classical video coder. In effect, the coder is assigned to transfer a fraction of the available bit-rate from the coding of the non-facial area to that of the facial area, thereby providing images with sharper facial features. Note that in cases where the a priori assumptions with respect to the source content are not satisfied (model breakdown), the classical video coder can be used as a 'fall-back' coding mode. We refer to this approach as *model-assisted* video coding, in order to distinguish it from *model-based* coding which relies more heavily on the data model. The benefits of our approach are that: (i) it guarantees an acceptable lower bound in coding quality since it relies on a good fall-back mode, (ii) it preserves the 'natural look' of images (i.e. no cartoon-like, textureless faces), (iii) it is compatible with existing decoders, and (iv) its requirements in terms of model-fitting accuracy are significantly reduced. In what follows, we concentrate on a specific type of video data (head-and-shoulders sequences), partial models (face locations), and fall-back video coders ($p \times 64$), with a global coding rate of 48 kbps for an input video signal in QCIF format. Despite the specificity of this framework, however, the concept is quite general. It can be used in the context of other video coders working at other rates, and the object tracking algorithms can also be redesigned for different applications where objects other than faces are of interest.

[2] The *automatic* fitting (i.e. ot involving any human interaction) of models – such as the wireframe models of Harashima et al. [2, 36], in real-time. to video data is far from being a solved problem.

[3] Throughout this paper, the term 'face location' is slightly abused for the sake of simplicity. This term is meant to encompass the case of peope turning their head to their left or right – thereby appearing in a profile, or even turning their back to the camera, where 'face location' should read 'location of the head outline'.

## 3. Automatic tracking of head outline and facial features

The detection of head outlines as well as outlines of persons (silhouettes) in still or moving images has been the object of active and recent research in computer vision [18, 9, 20, 21]. Only very recently was it realized that such tasks, when performed totally automatically, could be helpful in low bit-rate coding environments [30, 32]. The task of detecting face locations in a sequence of images is facilitated by both the fact that people's head outlines are consistently roughly elliptical, even when the persons appear in a profile, and by the temporal correlation from frame to frame. The task of detecting the eye–nose–mouth region is facilitated by the axial symmetry inherently present in a human face for a person facing the camera, or looking slightly to the side, and appearing in projections in successive frames of a video signal. In this section, we describe totally automatic low-complexity algorithms which were designed to perform the detection and tracking task in head-and-shoulders video sequences. The algorithms belong to a broad class of pattern-matching algorithms used for object detection [40, 38].

### 3.1. Face and feature location models

The model we adopted in order to represent the location of a face was simply that of an ellipse $\delta$, as shown in Fig. 1, characterized by a center $(x_0, y_0)$, the lengths of its minor and major axes $A$, $B$, and a 'tilt' angle $\theta_0$. Although the upper (hair) and lower (chin) areas in actual face outlines can have quite different curvatures, ellipses provide a good trade-off between model accuracy and parametric simplicity. Moreover, due to the fact that this information is not actually used to regenerate the face outline, a small lack of model-fitting accuracy does not have any significant impact in the overall performance of the coding process.

Since an elliptical head outline can in some cases provide only a rough estimate of the face location, we have chosen to refine this elliptical location model by identifying a rectangular region $\mathcal{W}$ inside the ellipse, which tightly captures the eyes, nose, and mouth of the person in the scene. This location model of an 'eyes–nose–mouth' region is similar to the one proposed by Lavagetto et al. [29, 10], with an extra degree of freedom which we introduce by allowing a slant of its vertical axis as shown in Fig. 2. This additional degree of freedom ensures that the detection will be robust in the (very frequent) case of slight head rotation (see Sections 3.4 and 3.5). The upper third of the window, denoted by $\mathcal{W}_u$, is further identified to contain the eyes and eyebrows – two most reliably symmetric features in a human face. The window $\mathcal{W}$ is entirely characterized by a center $(x_1, y_1)$, width $w$, height $h$, and slant angle $\theta_1$.

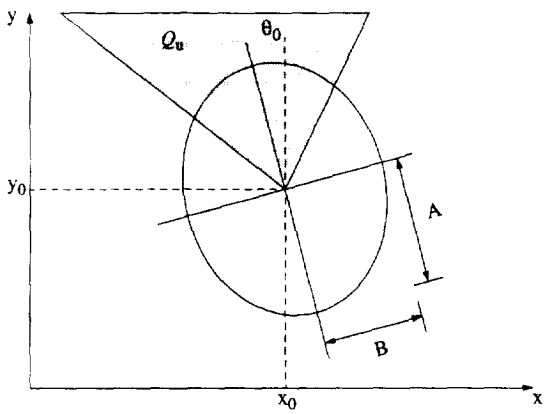


Fig. 1. Elliptical face location model.

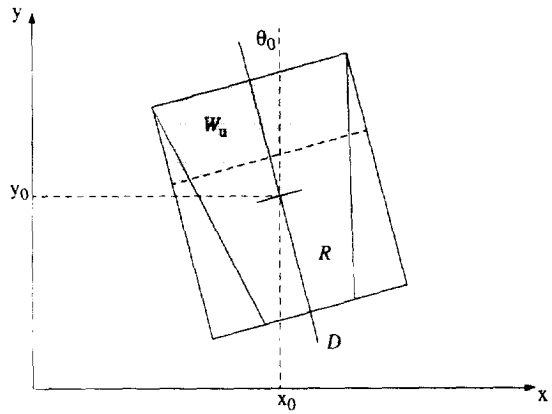


Fig. 2. Rectangular window as eyes–nose–mouth location model.

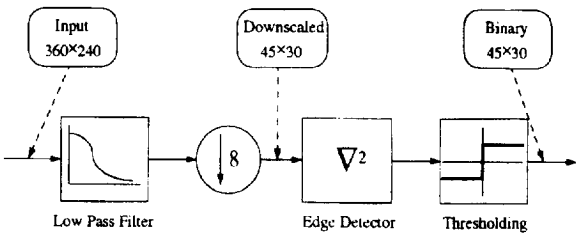


Fig. 3. Block diagram of edge extraction system for face location detection.

### 3.2. Generation of binary edge data

The binary input data to the automatic face location algorithm of the next section are obtained at the encoder through a preprocessing stage depicted in Fig. 3, consisting of the following cascade of operations:

1. Temporal downsampling of the input luminance video signal from 30 to 5 fps, consistently with the frame rate of the input signal to the video codec (See Section 4).
2. Low-pass filtering of input video frames of size $360 \times 240$ with a separable filter with cut-off frequency at $\pi/8$, followed by decimation by a factor 8 in both horizontal and vertical dimensions, producing low-pass images of size $45 \times 30$.
3. Edge detection on these images. The Sobel operator [37] is used to compute the two components of an image gradient. A gradient magnitude image is then obtained by computing the magnitude of the gradient at each pixel.
4. Thresholding of the gradient magnitude images to generate binary edge data.

This preprocessing stage is illustrated on a single frame of the sequence 'jim' in Fig. 4.

The binary input data to the automatic location algorithm for the eyes–nose–mouth region is generated in a similar way, albeit on images which are only downsampled by a factor of two, and therefore of size $180 \times 120$, in order not to lose the features of interest in the downsampling, namely eye, nose, and mouth edge data.

### 3.3. Detection of head outline

The algorithm detects the outline of a face location geometrically modeled as an ellipse, using as preprocessed input data binary thresholded gradient magnitude images of size $45 \times 30$. Our face

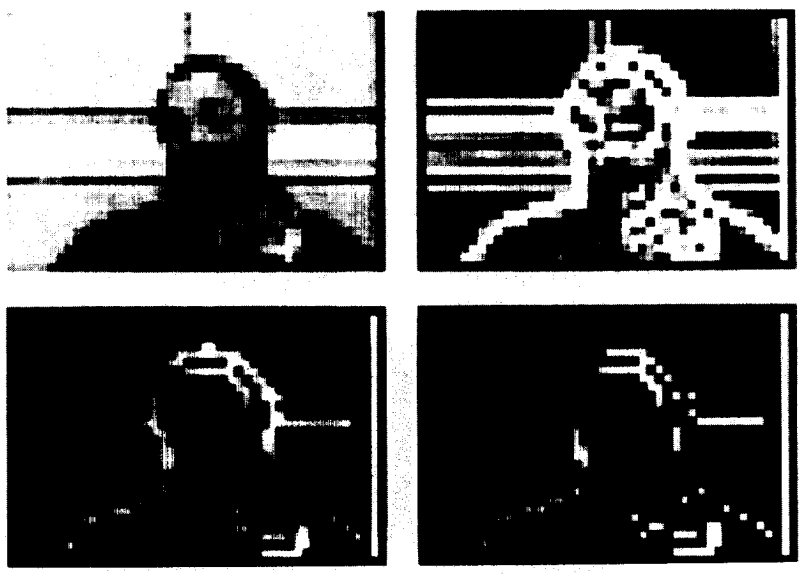

Fig. 4. Edge extraction for 'jim' image: (a) low-pass filtered downsampled ($\times 8$) image (magnified by four); (b) output of Sobel operator; (c) thresholded edge image; (d) thresholded edge image with head outline overlay.

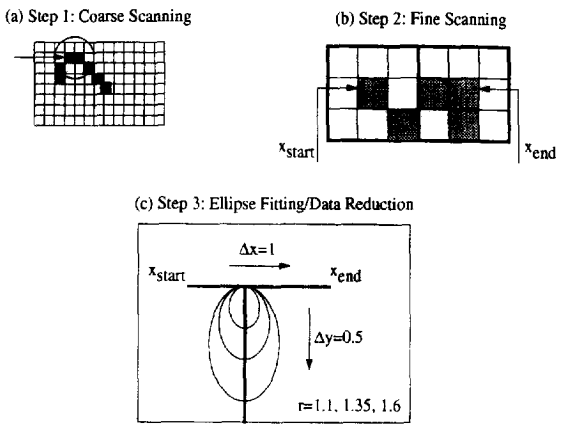


Fig. 5. Algorithm for automatic face detection and tracking in video sequences.

location detection algorithm was designed to locate both oval shapes (i.e. 'filled') as well as oval contours partially occluded by data. The algorithm is organized in a hierarchical three-step procedure: coarse scanning, fine scanning, and ellipse fitting. A final step consists of selecting the most likely among multiple candidates. This decomposition of the recognition and detection task in three steps, along with the small input image size, make the algorithm attractive for its low computational complexity. The different steps are described below, and are illustrated in Fig. 5.

*Step 1: Coarse scanning.* The input signal is segmented into blocks of size $B \times B$ (typically $5 \times 5$). Each block is *marked* if at least one of the pixels it contains is non-zero. The block array is then scanned in a left-to-right, top-to-bottom fashion, searching for contiguous runs of marked blocks. One such run is shown in the small circle, in Fig. 5(a). For each such run the following two steps are performed.

*Step 2: Fine scanning.* Fig. 5(b) shows the two circled blocks of the run of Fig. 5(a), appropriately magnified. The algorithm scans the pixels contained in the blocks of a run, again in a left-to-right, top-to-bottom fashion. Here, however, the algorithm is not interested in contiguous runs of pixels, but rather in the first line that contains non-zero pixels. The first and last non-zero pixels of that line, with coordinates $(X_{start}, Y)$, $(X_{end}, Y)$, define a *horizontal scanning region.*

The first two steps of the algorithm act as a horizontal edge-merging filter. The size of the block directly relates to the maximum allowable distance between merged edges. It also has a direct effect on the speed of the algorithm, which is favored by large block sizes. The purpose of these two steps is to identify candidate positions for the top of the head, where the edge data corresponding to the head outline is generally unencumbered by data corresponding to other objects. At the end of the second step, the algorithm has identified a horizontal segment which potentially contains the top of the head.

*Step 3: Ellipse fitting/data reduction.* In this third step, illustrated in Fig. 5(c), the algorithm scans the line segment defined by $(X_{start}, Y)$, $(X_{end}, Y)$. At each point of the segment ellipses of various sizes and aspect ratios are tried-out for fitness, with the topmost point of the ellipse always located on the horizontal scanning segment. Good matches are entered as entries in a list. After the search is completed on the segment, the algorithm continues at the point where it left off in Step 1. Only ellipses with 'zero tilt' ($\theta_0 = 0$) were considered here. The primary reason for imposing this restriction is that we could trade-off an extra degree of freedom (and hence algorithm simplicity) by extending the search range for the aspect ratio.[4] Another reason is that the orientation of the facial axis (corresponding to the major axis of the ellipse model) can be much more reliably obtained by exploiting considerations of facial symmetry, as described in Section 3.4.

The fitness of any given ellipse to the data is determined by computing normalized weighted average intensities $I_i$ and $I_e$ of the binary pixel data on the ellipse *contour* ($\mathscr{C}_i$) and *border* ($\mathscr{C}_e$), respectively. Although the contour of an ellipse is well-defined by its non-parametric form, the rasterization (spatial sampling) of image data necessitates

[4] Typical face outlines have been found to have aspect ratios in the range of (1.4, 1.6). Moreover, the face tilt has been found to be in the range $(-30°, +30°)$; a significant constraint due to the human anatomy. Within these ranges for $\theta$ and $r$, a tilted ellipse can be reasonably covered by a non-tilted one, albeit with a smaller aspect ratio (in the range (1.0, 1.4)).

the mapping of the continuous curve to actual image pixels. This is also true for the ellipse border. These discretized curves are defined as follows. Let $I_{\mathscr{E}}(i,j)$ be the index function for the set of points that are inside or on the ellipse $\mathscr{E}$. In other words,

$$I_{\mathscr{E}}(i,j) = \begin{cases} 1 & \text{if } (i,j) \text{ is inside or on } \mathscr{E}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

A pixel is classified as being on the ellipse *contour* ($\mathscr{C}_i$) if it is inside (or on) the ellipse, and at least one of the pixels in its $(2L+1)\times(2L+1)$ neighborhood is not, i.e.,

$$(i,j) \in \mathscr{C}_i \Leftrightarrow I_{\mathscr{E}}(i,j) = 1 \quad \text{and}$$

$$\sum_{k=i-L}^{i+L} \sum_{l=j-L}^{j+L} I_{\mathscr{E}}(k,l) < (2L+1)^2. \quad (2)$$

Similarly, a pixel is classified as being on the ellipse *border* ($\mathscr{C}_e$) if it is outside the ellipse, and at least one of the pixels in its $(2L+1)\times(2L+1)$ neighborhood is on or inside the ellipse, i.e.,

$$(i,j) \in \mathscr{C}_e \Leftrightarrow I_{\mathscr{E}}(i,j) = 0 \quad \text{and}$$

$$\sum_{k=i-L}^{i+L} \sum_{l=j-L}^{j+L} I_{\mathscr{E}}(k,l) > 0. \quad (3)$$

The parameter $L$ defines the desired *thickness* of the ellipse contour and border, and is a tunable design parameter.

Given the above definitions for contour and border pixels, the normalized weighted average intensities $I_e$ and $I_i$ are defined as follows:

$$I_i = \frac{1}{|\mathscr{C}_i|} \sum_{(m,n)\in\mathscr{C}_i} w_{m,n}\, p(m,n), \quad (4)$$

where $p(m,n)$ are the (binary) image data, $|\mathscr{C}_i|$ is the cardinality of $\mathscr{C}_i$, and $w_{m,n}$ are weighting factors introduced to enhance the contribution of the data in the upper quarter of the ellipse (see Fig. 2) – the most reliable region for fitting, i.e.,

$$w_{m,n} = \begin{cases} w > 1 & \text{if } (i,j) \in \mathscr{Q}_u, \\ 1 & \text{if } (i,j) \text{ not in } \mathscr{Q}_u. \end{cases}$$

In our experiments, a weight $w = 1.5$ was used, for which consistently reliable results were obtained.

Similarly, we define

$$I_e = \frac{1}{|\mathscr{C}_e|} \sum_{(m,n)\in\mathscr{C}_e} p(m,n). \quad (5)$$

The normalization with respect to the 'length' of the ellipse contour and border is necessary, in order to accommodate ellipses of different sizes.

An ellipse will fit ellipse-shaped data well whenever the value of $I_i$ is high (close to the maximum value $I_{max} = (3+w)/4$), and that of $I_e$ is low (close to zero). In order to translate this joint maximization–minimization problem to the maximization of a single quantity, we define a model-fitting ratio $R$ as

$$R = \frac{1 + I_i}{1 + I_e}. \quad (6)$$

The higher the value of $R$, the better the fit of the candidate ellipse to the head outline.[5]

In order to filter out false candidates, only ellipses which satisfy

$$I_i > I_{i_{min}} \quad \text{and} \quad I_e < I_{e_{max}}, \quad (7)$$

are considered, where $I_{i_{min}}$ and $I_{e_{max}}$ are tunable design parameters. Their use is necessitated by the fact that $R$ is mostly sensitive to the relative values of $I_i$ and $I_e$, and much less to their absolute values.

This fitness criterion attempts to capitalize on specific properties observed on actual video data. In most cases, only an arc of the ellipse is clearly distinguishable, due to partial occlusion and to motion in the area surrounding the face (e.g. the shoulders). Using the above thresholds and the metric $R$, the algorithm is able to 'lock on' to such arcs, and hence yield very good results even in cases of severely occluded faces.

The above three-step procedure will in general yield more than one ellipse with a good fit, as is illustrated in Fig. 6 for the sequence 'jim'. If there is a need to select a *single* final one (e.g. when it is known that the sequence only includes one person), then an elimination process has to be performed.

[5] In the hypothetical situation of perfectly ellipse-shaped data, the best-fitting ellipse aligned with the data would correspond to $I_i = I_{max}$, $I_e = 0$ and $R = 1 + I_{max}$.

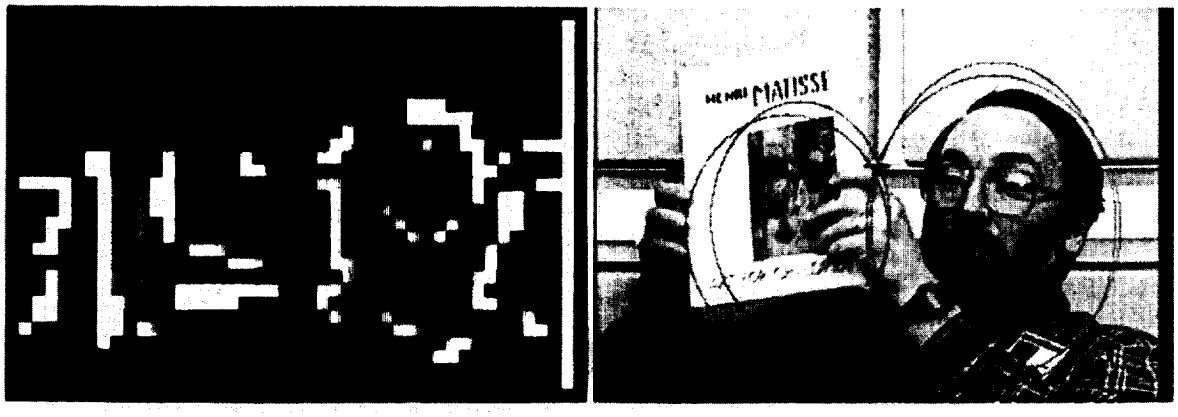


Fig. 6. Automatically detected candidate face locations in sequence 'jim' before multiple candidate elimination.

This process uses two 'confidence thresholds' $\Delta R_{\min}$ and $\Delta I_{e_{\min}}$. If the value of $R$ for the best-fitting ellipse is higher from the second best by more than $\Delta R_{\min}$, then the first ellipse is selected. If not, then if the border intensity difference between the two ellipses is higher than $\Delta I_{e_{\min}}$, the ellipse with the smallest $I_e$ is selected. If the border intensity difference is smaller than that (which rarely occurs in practice), then the original best candidate (the one with the maximum $R$) is selected. Selection of appropriate values for these thresholds was based on experimentation. Since the source to the algorithm is binary images, these thresholds tend to be to a large extent independent of the image content. In our experiments we consistently obtained good results for all image sequences examined.

### 3.4. Detection of eyes–nose–mouth region

The elliptical face location model described above can be refined by including a segmentation of the elliptical region into a rectangular window and its complement. The window needs to be positioned so that it tightly captures the region of the face corresponding to eyes and mouth; the features of interest. Our identification of eyes/mouth regions follows the procedure described by Lavagetto et al. [29, 10], and first proposed by Badiqué [5], and extends it to include detection in cases where (i) the subject does not directly face the camera, (ii) the subject has facial hair, and (iii) the subject is not white. The algorithm is based on exploiting the typical symmetry of facial features with respect to a longitudinal axis going through the nose and across the mouth. Our algorithm allows this symmetry axis to be *slanted* with respect to the vertical axis of the image, thereby ensuring more robustness in the detection of an eyes–nose–mouth region. In particular, detection of this region is still possible when the subject does not look directly at the camera (a very common occurrence in a video teleconferencing situation). The algorithm operates in two steps.

*Step 1: Definition of search region.* The center $(x_0, y_0)$ of the elliptical face location model is used to get estimates for the positioning of the eyes–nose–mouth window. The search region for the center of this window is defined as a square region of size $S \times S$ ($S$ was equal to 12 in our experiments). The window itself was chosen to be of fixed size $w \times h$, defined relatively to the minor and major axes of the face location model.

*Step 2: Scanning of search region.* For each candidate position $(x_k, y_k)$ of the window center in the search region, a *symmetry functional* with respect to the facial axis is computed, where this axis can be rotated by discrete angle values around the center of the window. In our experiments, the slant angle $\theta_k$ could take any of the discrete values $-10°$, $-5°$, $0°$, $5°$, $10°$. Let $S(m, n)$ denote the point which is symmetric to $(m, n)$ with respect to the axis

$\mathscr{L}((x_k, y_k), \theta_k)$. The symmetry functional is computed as follows:

$$S(x_k, y_k, \theta_k) = \frac{1}{A(\mathscr{R})}\left(\sum_{(m,n)\in \mathscr{R}\cap \mathscr{U}_u} w a_{m,n} + \sum_{(m,n)\in \mathscr{R}\setminus \mathscr{U}_u} a_{m,n}\right), \quad (8)$$

where $A(\mathscr{R})$ denotes the cardinality (area in pixels) of the trapezoid region $\mathscr{R}$ depicted in Fig. 2, $\mathscr{R}\setminus\mathscr{U}_u$ denotes the set difference of $\mathscr{R}$ and $\mathscr{U}_u$, $a_{m,n}$ is the function defined by

$$a_{m,n} = \begin{cases} 1 & \text{if } p(m,n) = p(S(m,n)) = 1, \\ \frac{1}{2} & \text{if } p(m,n) = p(S(m,n)) = 0, \\ 0 & \text{otherwise,} \end{cases}$$

and $w$ is a weighting factor greater than one. This weighting factor $w$ ensures that the edge data in $\mathscr{U}_u$ which is symmetric with respect to the axis, significantly contributes to the functional. The segmentation of the rectangular window into the regions $\mathscr{U}_u$ and $\mathscr{R}$ ensures that only data corresponding roughly to the eyes, nose and mouth be taken into consideration in the positioning of the window, and that this positioning is mostly enforced by 'eye data', which we have found to be the most reliably symmetric region (in a quantitative sense).

As in the ellipse detection case, false candidates defined as windows for which the density of data points in the upper third rectangle is below a minimum density $D_{\min}$, are filtered out.

### 3.5. Experimental detection results

The output of sample test runs of the automatic face location detection algorithm on sequences referred to as 'jim', 'jelena', 'roberto', is shown in

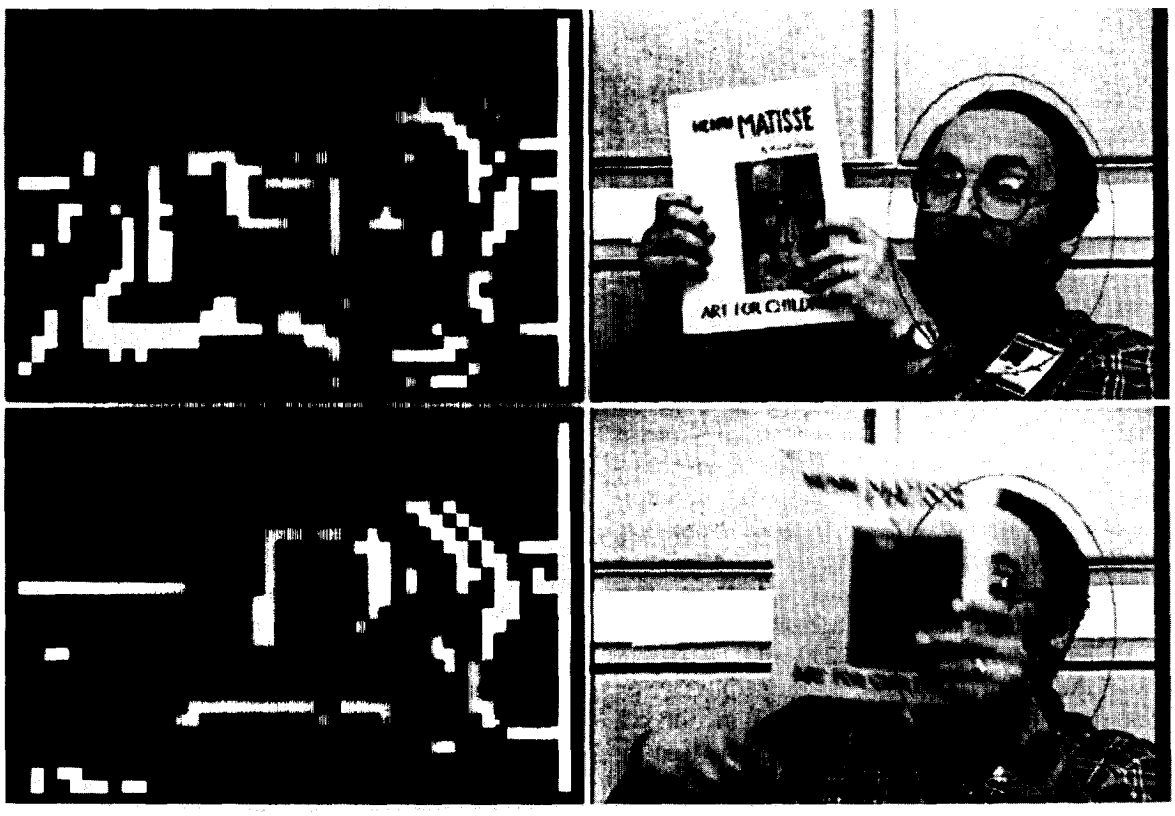


Fig. 7. Automatically detected face locations in sequence 'jim'.

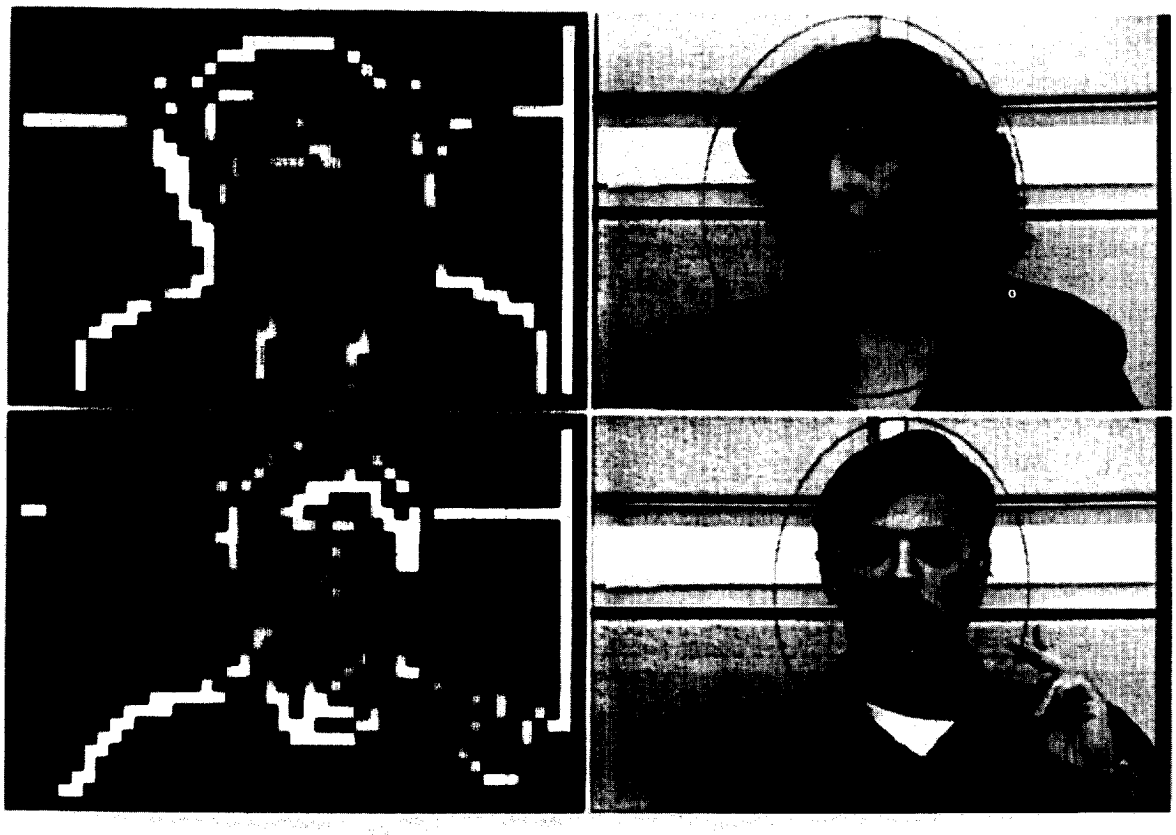

Fig. 8. Automatically detected face locations in sequences 'jelena' and 'roberto'.

Fig. 6–8. Fig. 6 shows an intermediate result, for the sequence 'jim', consisting of the output of the algorithm *before* the multiple candidate elimination step. The ellipses found at that stage are 'candidate' face locations. Figs. 7 and 8 show four pairs of images altogether. The images on the left show in white the input binary edge data with the best-fitting ellipse found by the automatic face location detection algorithm overlayed in gray. Note that these images are magnified by a factor of eight in both the horizontal and vertical directions. The images on the right show the best-fit magnified to the original image size of $360 \times 240$, and overlaid in gray on the originals.

The algorithm performs well, even in difficult situations such as partial occlusion of the face by a hand-held moving object. In the sequence 'jim' for example (Fig. 7), the sweeping motion of the magazine in front of Jim's face does not 'confuse' the algorithm.[6] In other words, the elliptical model fits Jim's head outline better (in terms of the model-fitting ratio of Eq. (6)) than the shape defined by the outline of the magazine – as it should – and even though the magazine severely occludes the face. In the case of more than one person, the algorithm tracks the location of the person's face for which the fit is best.

Fig. 9 shows sample results from automatic eyes–nose–mouth region detection. This time, the binary edge images on the left, magnified by a factor two in both dimensions, with best-fitting window overlayed in gray. It is of particular interest to note that a slanted rectangular window tightly captures the region of interest in 'jelena', where

[6] Of course, a hand-held *oral* object of roughly the same size as jim's face probably would.

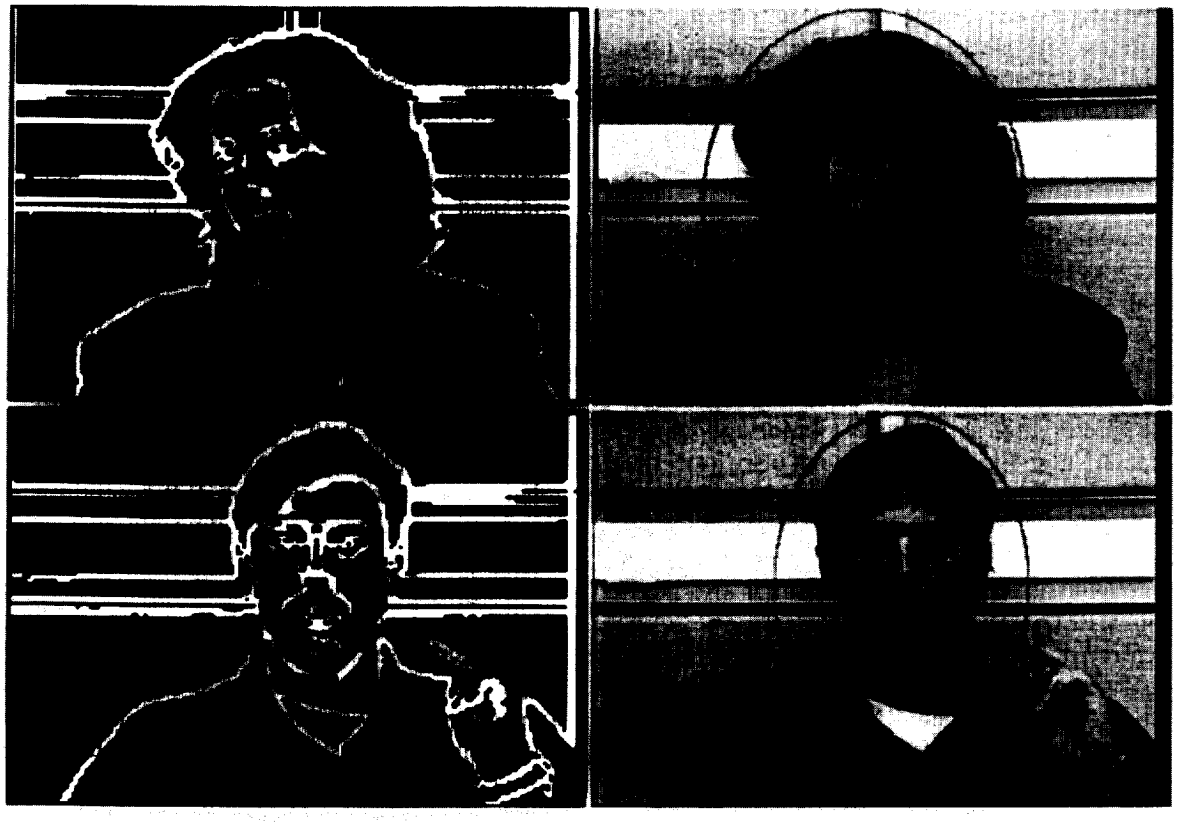

Fig. 9. Automatically detected eyes–nose–mouth locations in sequences 'jelena' and 'roberto'.

a simple window with no slant fails to exploit adequately facial axial symmetry and results in a detection failure.

## 4. Model-assisted coding using H.261

CCITT Recommendation H.261 [31,13] describes an algorithm for video coding at the rates of $p \times 64$ kbps, where $p = 1, 2, \ldots, 30$. The algorithm is a hybrid of discrete cosine transform (DCT) and DPCM schemes, with block-based motion estimation (ME) and compensation (MC). Although it lacks many of the coding features of algorithms designed by later standardization efforts (MPEG-1 and MPEG-2), it has been widely used to design ISDN-based video conferencing systems (the standard's main target application area). The normative part of the standard includes the specification of the decoder only; the encoder design is not specified, and hence significant flexibility is provided to its designer, allowing for various optimizations and resolutions of unavoidable trade-offs.

Our selection of H.261 as the basis for the design of a model-assisted codec was motivated by several factors. Firstly, it is a widely used algorithm for medium-to-low bit-rate video, with a large number of systems currently under use. Since, as we will see, model-assisted coding can be used with complete decoder compatibility, this makes it a very attractive target platform. Secondly, its performance at its lowest range (64 kbps) suffers from a significant amount of coding artifacts, as the complexity of the task exceeds the capabilities of the algorithms used. In order to mitigate this effect, most current implementations aim at keeping a fairly 'constant' coded picture quality at the expense of temporal resolution. When the sequence motion is moderate to

high, temporal subsampling down to a frame rate as low as 2 fps is usually required. This in turn results in synchronization problems between audio and video (as perceived by users), and in particular between lip movement and speech (lip-sync).[7] For these reasons, the frame rate used in this paper was fixed at a constant 5 fps.

An additional consideration in our selection of H.261 was that it incorporates the two mostly used coding schemes today, namely the DCT and MC. Consequently, by using it as a target implementation, we could examine how much lower model-assisted coding could extend the coding rate range, while preserving perceptually important image characteristics. Finally, we should also note the importance of the capability of the H.261 syntax to specify different quantization levels at a sufficiently fine scale (macroblock), which is a necessary feature for model-assisted coding (this feature is also present in MPEG-1 and MPEG-2).

Before describing the model-assisted H.261 codec, we first give a brief overview of the characteristics of the so-called RM8 encoder. The discussion is limited to the features directly affected by its model-assisted counterpart; a detailed description can be found in [11].

### 4.1. Reference Model 8

H.261 is a block-based, motion-compensated transform coding (DCT) design. It provides support for intra (I) and predicted (P) pictures, but not for bidirectionally interpolated (B) pictures. It prescribes the quantization of DCT coefficients using identical uniform quantizers with dead zones for all AC coefficients, and 8-bit uniform quantization for the DC coefficients (with a step size of 8). In other words, there is no perceptual frequency weighting, and there is also no capability to institute one dynamically (except from contrived cases). The AC coefficient quantizer step size is determined as twice the value of a parameter $Q_p$ (or MQUANT, as it is referred to in the standard), which can be indicated at up to the macroblock (MB) level. A rectangular array of $11 \times 3$ MBs defines a group of blocks (GOB). The source material used in our experiments had a resolution of $180 \times 120$ (QCIF), resulting in a total of $2\frac{1}{3}$ GOBs per picture (frame).

Reference Model 8 (RM8) is a 'reference implementation' of an H.261 encoder that was developed and used internally by the H.261 working group [11], with the purpose of providing a common environment in which experiments could be conducted. It features a technique called 'variable thresholding', in order to increase the length of the run-lengths in the zig-zag scanned DCT coefficients. In effect, this technique eliminates series of DCT coefficients with small enough values; thresholding is applied prior to quantization, and is an inherently nonlinear operation. It is generally effective, however, in improving coding efficiency, particularly at lower bit-rates. In addition to thresholding, spatial low-pass filtering of blocks with non-zero motion vectors is also used. The MC/no-MC decision uses the values of the macroblock and displaced macroblock differences, and is based on an experimentally determined curve. Similarly, the intra/non-intra decision is based on comparison of the variances of the original and motion-compensated macroblocks. Predicted macroblocks in P pictures are skipped if their motion vector is zero and all of their blocks have zero components after quantization. Macroblocks are also skipped in cases of output buffer overflow (forced skipping).

The rate control strategy in RM8 is as follows. The very first picture, which is an I-picture, is coded with a constant $Q_p$ of 16. The output buffer is then initialized at 50% occupancy. For the remaining pictures, $Q_p$ is adapted at the start of each line of MBs within a GOB. In other words, $Q_p$ will be adapted three times within each GOB. The buffer occupancy is examined after the transmission of each MB and, if overflow occurs, the next MB is skipped. Note that this will result in a small temporary buffer overflow; in other words, the MB that caused the overflow *will* be transmitted (this can easily be done by using a 'guard zone' in the output buffer). $Q_p$ is updated 'linearly' with the buffer occupancy according to the relation

$$Q_{p_i} = \min\left\{31, \left\lfloor \frac{B_i}{B_{\max}/32} \right\rfloor + 1\right\}, \tag{9}$$

[7] It is generally agreed upon that the minimum frame rate required for lip-synch is about 5 fps.

where $Q_{p_i}$ is the value of $Q_p$ selected for MB $i$, $B_i$ is the output buffer occupancy just prior to coding MB $i$, and $B_{\max}$ is the output buffer size. A buffer size of $6400 \times q$ bits is used, given a bit-rate of $q \times 64$ kbps for the video signal only. Hence, in our case a buffer size of 6400 bits was used.

This formula performs reasonably well for moderate-to-low complexity frames, but can result in serious artifacts when highly detailed features and/or moderately high motion activity are present. In particular, forced skipped macroblocks that may be caused due to buffer overflow may result in 'shearing' – an object appears split (at a macroblock boundary) and its pieces shifted, even under moderate motion. The facial area is particularly susceptible to this kind of artifacts, since its complexity can quickly drive the output buffer to overflow.

### 4.2. Model-assisted rate control

The basic premise of the concept of model-assisted coding is to assign different 'quality levels' at different portions of an image, that bear perceptual significance to a viewer. Although a theoretically optimal (under specific assumptions) approach similar to [39] could be followed (with appropriate modifications to allow for a spatially weighted distortion measure), it would have the significant drawback of high complexity and high delay – the latter being very undesirable in person-to-person communication applications. Besides, an optimal rate control approach would make comparisons with RM8 unfair, as the difference would be mostly attributable to the optimality rather than the model-assisted component.

With the above considerations, we developed an approach that maintains the sequential processing structure of RM8, i.e. macroblocks are coded in their regular left to right, top to bottom order within each GOB, and quantizer selection is based on the current buffer occupancy level. The precise location of an MB, however, now plays an important role in the process.

#### 4.2.1. Buffer rate modulation

In order to be able to spend more bits in regions of interest while staying within a prescribed bit budget (or avoiding buffer overflow), it is necessary to spend less bits on the remaining image areas. We assume that there are $M$ regions of interest $\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_M$ in an image, with corresponding areas $A_1, A_2, \ldots, A_M$. We require that the regions are non-overlapping, i.e.

$$\mathcal{A}_i \cap \mathcal{A}_j = \emptyset \quad \text{when } i \neq j. \tag{10}$$

The rectangular region encompassing the whole image is denoted by $\mathcal{A}$, and its area by $A$. We also assume that the coding of each macroblock uses $\beta$ bits on the average, when the target budget rate is $R$ and the buffer size is $B_{\max}$. Let $\beta_1, \beta_2, \ldots, \beta_M$ denote the target average number of bits per macroblock for the coding of each of the regions of interest. Note that in general we would require $\beta_i > \beta$, i.e., an improved quality within the regions of interest. Let also $\mathcal{A}_0$ denote the portion of the image that belongs to none of the regions of interest, with a corresponding area $A_0$ and average number of bits per macroblock $\beta_0$. In order to satisfy the given average bit budget, the following relation must hold:

$$\sum_{i=0}^{M} \beta_i A_i = \beta A. \tag{11}$$

If the parameters $\beta_1, \beta_2, \ldots, \beta_M$ are given, it follows from Eq. (11) that

$$\beta_0 = \frac{\beta A - \sum_{i=1}^{M} \beta_i A_i}{A - \sum_{i=1}^{M} A_i}. \tag{12}$$

This formula defines the 'equivalent average quality' for the image region that is exterior to all objects (henceforth called exterior region for brevity), and is uniquely specified by the desired average coding quality of the coded regions and their sizes. In most cases, it is more convenient to express (12) in terms of the relative average qualities $\gamma_i \equiv \beta_i/\beta$, $i = 0, \ldots, M$:

$$\gamma_0 = \frac{A - \sum_{i=1}^{M} \gamma_i A_i}{A - \sum_{i=1}^{M} A_i}. \tag{13}$$

Note that if $\gamma_i > 1$ for all $i > 0$, then $\gamma_0 < 1$ as should be expected.

Let us assume that the rate control operation of the generic (not model-assisted) encoder is governed

by the function

$$Q_{p_i} = f(B_i), \tag{14}$$

of which a particular example is Eq. (9). The function $f(\cdot)$ can be quite general, and may also depend on the input signal. It is generally assumed that it is at least an increasing function of $B_i$. The output buffer evolution can be described by

$$B_i = B_{i-1} + c(i-1) - r, \tag{15}$$

where $B_i$ is the buffer occupancy prior to coding MB $i$, $r$ is the average target rate (in bits per MB), and $c(i)$ is the number of bits spent to code the $i$th MB and all its immediately preceding overhead information (headers, etc.). The function $c(i)$ depends on the input signal, as well as the current value of $Q_p$; the latter depends on the selection of the function $f(\cdot)$.

In order to convert Eqs. (14) and (15) to provide location-dependent, model-assisted operation, we introduce the concept of *buffer rate modulation*. The idea is to modulate the target rate in Eq. (15) so that more bits are spent for MBs that are inside regions of interest, and less for MBs that are not. The rate $r$ in Eq. (15) now becomes location-dependent, and is given by

$$r_i = \gamma_{\zeta(i)} r, \tag{16}$$

where the region index function $\zeta(i)$ associates the position of MB $i$ with the region in which it belongs to.[8] The buffer evolution can now be described by

$$B_i = B_{i-1} + c_{\zeta(i-1)}(i-1) - \gamma_{\zeta(i)} r, \tag{17}$$

where the number of bits spent $c_{\zeta(i)}(i)$ is now region-dependent. Assuming stationarity for $c_k(i)$ for fixed $k$, and taking expectations on both sides of Eq. (17), we obtain the average rate for region $k$ as

$$\bar{c}_k = \gamma_k r. \tag{18}$$

If the values of $\gamma_i$ satisfy the budget constraint given by Eq. (11), then the total average rate will be exactly $r$.

Obviously, an actual system will operate with a regular, unmodulated output buffer which will be emptied at the constant rate $r$. Consequently, both Eqs. (15) and (17) have to be tracked in order to avoid overflow or underflow (the latter is of importance only if alternate synchronization mechanisms are not available). The modulated, 'virtual' buffer is used to drive the evolution of $Q_p$ via the function $f(\cdot)$ of Eq. (14), while the actual buffer is monitored to force MB skipping in cases of overflow. When the virtual buffer overflows, no particular action is taken and $Q_p$ is typically assigned its maximum value (depending on $f(\cdot)$).

Note that in general the $\gamma_i$ do not have to be all positive. Although a negative modulated buffer rate is somewhat surprising and counterintuitive (as it means that the buffer is actually receiving bits from the channel instead of always transmitting them), it helps to severely constrain bit allocation in the exterior, particularly in pictures which are difficult to code.

We should note that Eq. (18) is valid for stationary processes (or, more generally, for those processes possessing the ergodic property) and in steady state (after a large number of steps). In practice, stationarity can only be approximately assumed. In addition, since the image regions $\mathscr{A}_i$ have in general rather smaller dimensions, one cannot expect that Eq. (17) will converge while operating inside each of the regions. Furthermore, taking into account the particular pattern with which macroblocks are scanned within each image (due to GOB-based partitioning), it is possible that only a very small number of macroblocks of each region is continuously processed at a time.

For example, consider the case when while scanning the macroblocks we move from region $k$ to $l$ with $\gamma_k \ll \gamma_l$. If we reached steady-state conditions in region $k$, then the (virtual) buffer occupancy should be high, making it difficult to reach quickly a lower value while in region $l$. If the portion of $l$ that is being processed is small (1–2 macroblocks), the effect of rate modulation will not be significant.

Consequently, we see that there is a tradeoff between *long-term convergence* and *convergence speed*. The former is desirable, in order to satisfy the a priori rate allocation given by Eq. (11), while the latter is necessary in order to accommodate region segments of small size.

[8] A macroblock is considered to belong to a region if at least one of its pixels is inside that particular region.

*4.2.2. Buffer size modulation*

To address the issue of convergence speed, the concept of *buffer size modulation* was introduced. It is similar to the rate modulation approach discussed above, with the difference that we are now modulating the buffer occupancy (or size[9]) instead of its rate. This is achieved by modifying Eq. (14) as follows:

$$Q_{p_i} = g(B_i, i) \equiv f\left(\frac{B_i}{\mu_{\zeta(i)}}\right), \tag{19}$$

where $\mu_i$ are the modulation factors for each region. In effect, this function operates in regions of low interest ($\gamma_i < 1, \mu_i < 1$) as if the buffer occupancy was higher that it actually is, while in regions of high interest ($\gamma_i > 1, \mu_i > 1$) it operates as if the buffer occupancy was lower. The result is that $Q_{p_i}$ is 'pushed' to a higher or lower value, depending on the position of the MB within the image. Since here we are interested in boosting the transient behavior, the $\mu_i$ values for each region are time-dependent, converging to the value of 1 within each step. This 'fading' behavior ensures that the steady-state behavior of Eq. (17) is not affected. The precise values of $\mu_i$ are empirically obtained, but we have found that they are insensitive to the sequence content and target coding rate.

Of particular importance are the implications of the above scheme regarding potential actual buffer overflows. Although there is obviously no problem in regions with $\mu \leqslant 1$, since the buffer occupancy is overestimated, buffer overflow in regions with high $\mu_i$'s can potentially occur rather quickly. We should note, however, that when entering an object region from a lower coding quality region, the buffer occupancy is low (e.g. less than $B_{\max}/\gamma_0$ on the average for the exterior). This leaves ample buffer space to absorb the rapid increase in the number of bits generated while coding blocks inside an object region. Furthermore, due to the MB scanning pattern, series of MBs of one region alternate with those of another, and hence 'relief' intervals are present in which the output buffer is allowed to drain.

Applying Eq. (19) to (9), we obtain

$$Q_{p_i} = \min\left\{31, \left\lfloor\frac{B_i/\mu_{\zeta(i)}}{B_{\max}/32}\right\rfloor + 1\right\}. \tag{20}$$

In addition to RM8s updates of $Q_p$ at the start of each line of MBs at each GOB, in this case $Q_p$ is also updated for each macroblock that is inside a region with $\gamma_i > 1$. A very frequent update is generally undesirable due to the overhead bits required in specifying the new value each time. Note that in this particular case of $f(\cdot)$, the scheme could be equivalently considered as $Q_p$ modulation; this is not the case if $f(\cdot)$ takes more complicated forms. An example is the MPEG-2 Test Model rate control where additional local activity measures are taken into account. In general, both such schemes could be used simultaneously, with each region having its own target rate.

In summary, buffer rate modulation allows one to force the rate control algorithm to spend a specified number of bits more in regions of interest, while buffer size modulation ensures that these bits are evenly distributed in the macroblocks of each region. It is important to note that both techniques can be applied in general to any rate control scheme; by modulating parameters which are considered external and fixed to the rate control algorithm (buffer size and buffer rate) they would simply operate it at a number of different target rates depending on the image area being coded. Furthermore, complete decoder compatibility is maintained as the region location information does not have to be transmitted to the decoder, and is only used in the encoder.

*4.3. Coding results*

The sequences in QCIF format referred to as 'jelena' and 'roberto' were processed with the standard RM8 algorithm, and with RM8 augmented with the model-assisted rate control algorithm modifications (MA) described in the previous section. Both input sequences were at QCIF resolution,

[9] As described here, the buffer occupancy $B_i$ is modulated; since, however, the function $f(\cdot)$ typically involves $B_i$ as part of the fraction $B_i/B_{\max}$, one can equivalently consider that the buffer size is modulated.

Table 1
Model-assisted coding simulation parameters

| | Jelena | Roberto |
|---|---|---|
| $\gamma_{face}$ | 6 | 6 |
| $1/\mu_{face}$ | 0.85 | 0.85 |
| $\Delta(1/\mu_{face})$ | 0.1 | 0.1 |
| $\mu_{head}$ | 1 | 1 |
| $\mu_{ext}$ | 1 | 1 |

Table 2
Bits per macroblock for each region for RM8 and model-assisted coding (48 kbps)

| | Jelena | | Roberto | |
|---|---|---|---|---|
| | RM8 | MA | RM8 | MA |
| Face | 236.67 | 419.17 | 294.24 | 465.65 |
| Exterior | 105.45 | 73.29 | 93.58 | 61.00 |
| Overall | 125.05 | 125.20 | 126.15 | 126.65 |

temporally downsampled to 5 fps, and were coded at the target rate of 48 kbps. The MA algorithm used the detected facial area (rectangle) as the single region of interest. The values of the various parameters for each region are shown in Table 1. An inversely linear fading was used for the $\mu_i$ parameters; in other words, $1/\mu_i$ increased linearly ($\Delta(1/\mu_i) = \text{const.}$), up to the point where it obtained the value 1.

Table 2 shows the average number of bits spent in the various image regions – the facial (rectangular) region and the exterior region. We observe that the number of bits spent in the facial area increased by a factor of 60–75%, while the number of bits spent in the exterior dropped by 30–35%. More importantly, the bit distribution within each region was approximately uniform.

This is illustrated in Fig. 10, which depicts the $Q_p$ assignments for two pairs of coded frames from 'roberto' (frames 24 and 84), for regular RM8 and model-assisted coding. Each image macroblock is colored according to its type: light gray background macroblocks are skipped, while the rest are assigned a grayscale value proportional to their $Q_p$ value (the *darker* the macroblock appears, the *finer* the quantization). In both cases we can clearly see that with the model-assisted approach the exterior region is quantized more coarsely, while the interior one more finely.

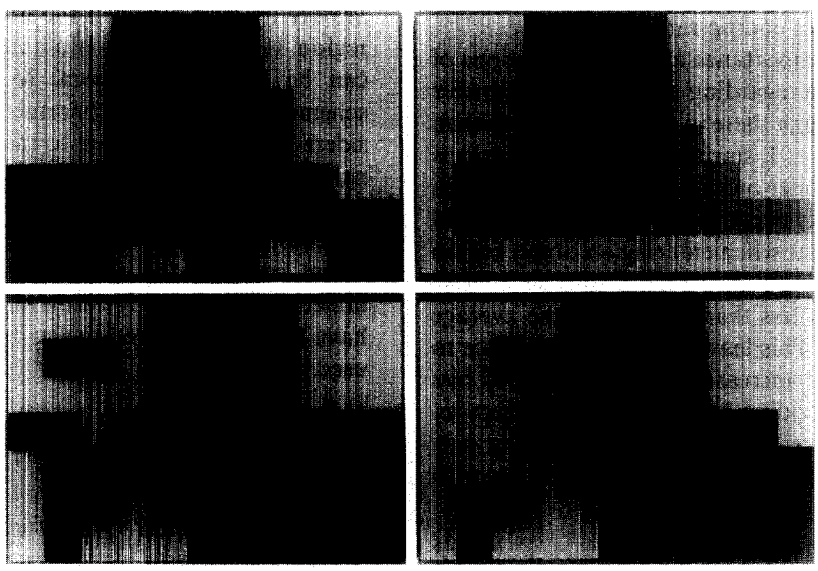

Fig. 10. Values of $Q_p$ for sequence 'roberto' (frames 24 and 84, respectively) coded with $p \times 64$ at 48 kbps, without (left) and with (right), model-assisted rate control. Light gray background macroblocks are skipped, grayscale values are proportional to the macroblock's $Q_p$ value.

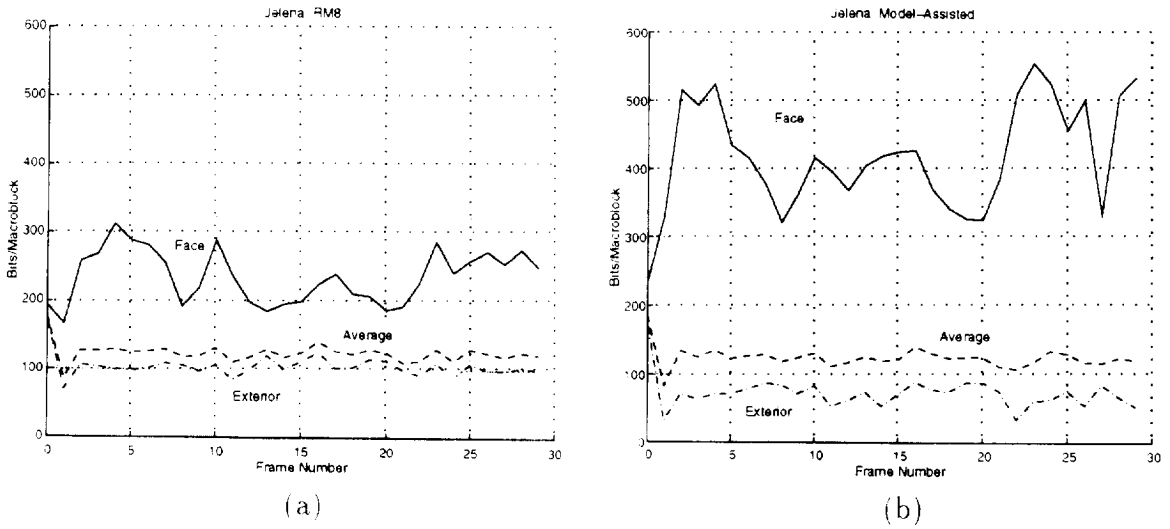


Fig. 11. Bit distributions in face, head, and exterior locations for the 'jelena' sequence for (a) RM8 and (b) model-assisted coding (48 kbps).

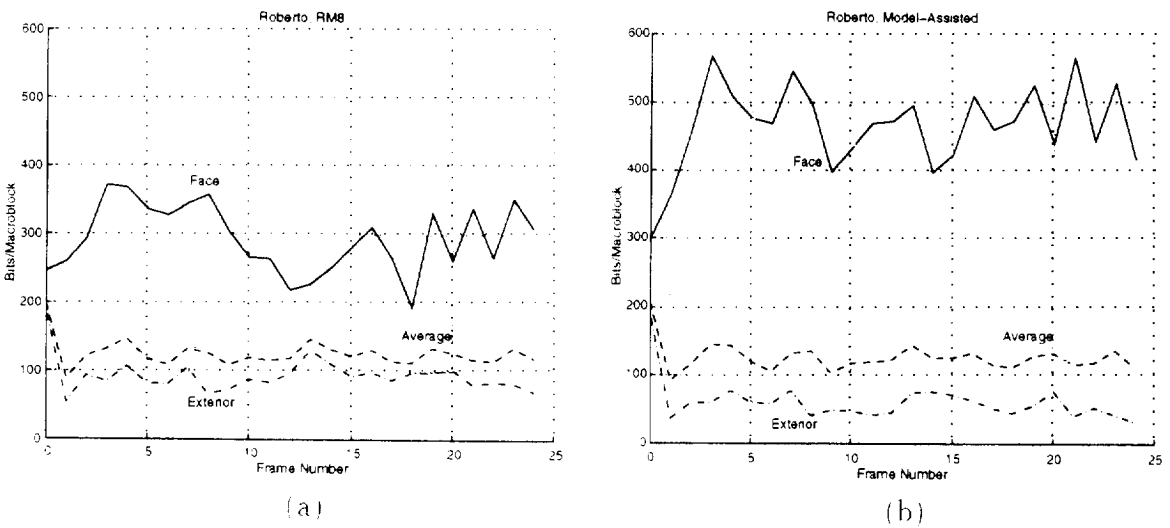


Fig. 12. Bit distributions in face, head, and exterior locations for the 'roberto' sequence for (a) RM8 and (b) model-assisted coding (48 kbps).

Figs. 11 and 12 show the per-frame distribution of bits in the facial, head, and exterior areas for the sequences 'jelena' and 'roberto', respectively, for both RM8 and model-assisted coding. The total, per-frame averages are also shown. In both cases we can see a significant increase (exceeding 100% in some cases) in the number of bits spent in the facial area. We should note that due to the structure of the coding process, there is a minimum number of bits that will always be spent in the exterior region inspite of the values of $\gamma_i$, and which corresponds to coding all the non-skipped macroblocks at the

Fig. 13. Stills from sequence 'jelena' (frames 210 and 408) coded with $p \times 64$ at 48 kbps, respectively, without (left) and with (right), model-assisted rate control.

coarsest level ($Q_p = 31$). Consequently, Eq. (18) can only be approximately satisfied, with a threshold value for $\gamma_{\text{face}}$ after which the model-assisted approach cannot affect the bit distribution any further.

Representative frames from 'roberto' and 'jelena' are shown in Figs. 13 and 14. The left-hand side images show the results of RM8, while the right-hand side depict the results with model-assisted coding. Noticeable improvement is evident, particularly in the eye and mouth areas. For example, the contour and interior of the eyes is clearly visible in the model-assisted case for both sequences. Similarly, the mouth area is more clearly delineated. Significant improvement is also visible in the moving sequence. In contrast with the RM8 results, maintaining eye contact with the subjects is not hindered by severe coding artifacts.

It should be noted that in the moving sequences, other artifacts (color bleeding, mosquito effects, etc.) present in the sequences help to (unfavorably) mask the effects of model-assisted bit allocation. To mitigate this problem, an edge-preserving smoothing post-filter can be used [3]. A perceptual quality evaluation experiment involving 20 subjects, and including the two sequences 'jelena' and 'roberto', resulted in a non-negligible increment of mean opinion score (MOS) of half a point on a five-point scale, elevating subjective audiovisual quality from less-than-fair to fair-to-good. The experiment clearly indicates that subjects are indeed sensitive to improved coded image quality in facial areas.

Fig. 14. Stills from sequence 'roberto' (frames 24 and 84) coded with $p \times 64$ at 48 kbps, respectively, without (left) and with (right), model-assisted rate control.

## 5. Conclusion

In this paper, we described a way to selectively encode different areas in head-and-shoulders video sequences typical of teleconferencing situations, thereby ensuring that facial features are sharp in image sequences coded at a low bit-rate. The approach, referred to as model-assisted coding, relies on the automatic detection of face locations in video sequences. The face location information is used by a low bit-rate video coding system in a model-assisted dynamic bit allocator module which uses object-selective quantization.

Head location detection is based on a low-complexity hierarchical algorithm that models the head as an ellipse, and utilizes physical structure information to robustly identify head contours, even in cases of severe occlusion. Face location is then estimated starting from the head position information, and exploiting the natural symmetry that can be found in the facial features with respect to a vertical axis.

In order to utilize the face and head location information for object-selective quantization, two novel approaches were developed: buffer rate modulation and buffer size modulation. The former ensures that particular percentages of bits are spent in regions of perceptual interest, while the latter ensures that these bits are uniformly spent within each region. The algorithms are quite general, and can be applied to any rate control scheme that depends on the channel rate and buffer occupancy levels in selecting quantization parameters.

Moreover, they offer complete decoder compatibility, as the region models are only utilized in the encoder.

The above algorithms were incorporated and evaluated on a CCITT H.261 encoder, using RM8 as a baseline encoder and a target rate of 48 kbps for the video signal. In all cases, model-assisted coding achieved significantly improved coding results, with markedly improved facial features. The modulation algorithms were very effective in achieving bit transfer to perceptually significant areas, and evenly distributing the bits inside each of them. With an average transfer of about 30–35% of the available bits from the coding of the exterior area to the coding of the facial area, maintaining eye contact with the video sequence subject was no longer hindered by severe coding artifacts.

We should note that although these techniques can significantly improve the perceptual quality of low bit-rate coded sequences, they are ultimately limited by the structure of the coding algorithm used. Since their operation consists of 'shaping' the bit allocation of the encoder to favor perceptually important regions, the maximum achievable quality is limited by what the algorithm can provide at the particular target (modulated) rates. These techniques can then be used to extend the lower range of coding rates by a factor of approximately two, while maintaining comparable perceptual image quality. This reduction is significant for the quality of visual communication applications, as it can introduce acceptable levels of quality within economically viable rates of transmission.

Even though a specific coding system is described, the concept is very general and could be used in the context of other video coders. The detection and tracking algorithm could also be tailored to different applications, i.e. to track any object with a simple geometric outline known a priori to be present in the scene, and also be extended to operate on multiple simultaneous objects.

## Acknowledgements

The authors would like to thank Nikil Jayant for suggesting the integration of face location tracking to a full-band codec such as one based on the $p \times 64$ video coding standard.

## References

[1] K. Aizawa, C.S. Choi, H. Harashima and T.S. Huang, "Human facial motion analysis and synthesis with applications to model-based coding", in: *Motion Analysis and Image Sequence Processing*, Chapter 11, Kluwer Academic Publishers, Dordrecht, 1993.

[2] K. Aizawa, H. Harashima and T. Saito, "Model-based analysis synthesis image coding (MBASIC) system for a person's face", *Signal Processing: Image Communication*, Vol. 1, No. 2, October 1989, pp. 139–152.

[3] J. Apostolopoulos and N.S. Jayant, Pre and postprocessing for very low bit rate video compression, Private communication.

[4] R. Aravind, G.L. Cash, D.L. Duttweiler, H.-M. Hang, B.G. Haskell and A. Puri, "Image and video coding standards", *AT&T Tech. J.*, Vol. 72, No. 1, January/February 1993.

[5] E. Badiqué, "Knowledge-based facial area recognition and improved coding in a CCITT-compatible low bit rate video codec", *Proc. PCS*, 1990.

[6] C. Braccini, S. Curinga and F. Lavagetto, "Visual-acoustic modeling of videophone sources for very low bit rate coding", *Italian National Council of Research Workshop*, Rome, January 1994.

[7] M. Buck and N. Diehl, "Model-based image sequence coding", in: *Motion Analysis and Image Sequence Processing*, Chapter 10, Kluwer Academic Publishers, Dordrecht, 1993.

[8] C.S. Choi, H. Harashima and T. Takebe, "Analysis and synthesis of facial expressions in knowledge-based coding of facial image sequences", *Proc. Internat. Conf. Acoust. Speech Signal Process. '91*, 1991.

[9] I. Craw, H. Ellis and J.R. Lishman, "Automatic extraction of face features", *Pattern Recognition Lett.*, Vol. 5, No. 2, February 1987.

[10] S. Curinga, A. Grattarola and F. Lavagetto, "Synthesis and animation of human faces: artificial reality in interpersonal video communication", in: B. Falcidieno and T.L. Kunii, eds., *Modeling in Computer Graphics*, Springer, Berlin, 1993.

[11] "Description of reference model 8 (RM8)", CCITT SGXV WG4, Specialists Group on Coding for Visual Telephony, Doc. 525, June 1989.

[12] N. Diehl, "Object-oriented motion estimation and segmentation in image sequences", *Signal Processing: Image Communication*, Vol. 3, No. 1, February 1991, pp. 23–56.

[13] "Draft revision of recommendation H.261: Video codec for audiovisual services at $p \times 64$ kbit/s", *Signal Processing: Image Communication*, Vol. 2, No. 2, August 1990, pp. 221–239.

[14] A. Eleftheriadis and A. Jacquin, "Model-assisted coding of video teleconferencing sequences at low bit rates", *Proc. ISCAS '94*, May–June 1994.

[15] A. Eleftheriadis and A. Jacquin, "Automatic face location detection and tracking for model-assisted coding of video teleconferencing sequences at low bit-rates", *Signal Processing: Image Communication*, Vol. 7, No. 3, September 1995, pp. 231–248.

[16] B. Falcidieno and T.L. Kunii, eds., *Modeling in Computer Graphics*, Springer, Berlin, 1993.
[17] G. Farin, *Curves and Surfaces for Computer-Aided Geometric Design*, Academic Press, New York, 1993.
[18] M.A. Fischler and R.A. Elschlager, "The representation and matching of pictorial structures", *IEEE Trans. Comput.*, January 1973.
[19] R. Forchheimer and T. Kronander, "Image coding – From waveforms to animation", *IEEE Trans. Acoust. Speech Signal Process.*, Vol. 37, No. 12, December 1989, pp. 2008–2023.
[20] V. Govindaraju, D.B. Sher and S.N. Srihari, "Locating human faces in newspaper photographs", *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition*, June 1989.
[21] V. Govindaraju, S.N. Srihari and D.B. Sher, "A computational model for face location", *Proc. 3rd Internat. Conf. on Computer Vision*, December 1990.
[22] M. Hötter, "Object-oriented analysis-synthesis coding based on moving two-dimensional objects", *Signal Processing: Image Communication*, Vol. 2, No. 4, December 1990, pp. 409–428.
[23] M. Hötter and R. Thoma, "Image segmentation based on object oriented mapping parameter estimation", *Signal Processing*, Vol. 15, No. 3, October 1988, pp. 315–334.
[24] T.S. Huang, S.C. Reddy and K. Aizawa, "Human facial motion modeling, analysis and synthesis for video compression", *Proc. SPIE VCIP*, Vol. 1605, 1991, pp. 234–241.
[25] *International Workshop on Coding Techniques for Very Low Bit-rate Video*, University of Essex, England, April 1994.
[26] N.S. Jayant, "Signal compression: Technology targets and research directions", *IEEE J. Selected Areas Commun. – Special issue on Speech and Image Coding*, June 1992.
[27] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active contour models", *Proc. Internat. Conf. on Computer Vision*, 1987, pp. 259–268.
[28] M. Kunt, A. Ikonomopoulos and M. Kocher, "Second-generation image coding techniques", *Proc. IEEE*, Vol. 73, No. 4, April 1985, pp. 549–574.
[29] F. Lavagetto and S. Curinga, "Object-oriented scene modeling for interpersonal video communication at very low bit-rate", *Signal Processing: Image Communication*, Vol. 6, No. 5, October 1994, pp. 379–395.
[30] C. Lettera and L. Masera, "Foreground/background segmentation in videotelephony", *Signal Processing: Image Communication*, Vol. 1, No. 2, October 1989, pp. 181–189.
[31] M. Liou, "Overview of the $p \times 64$ kbit/s video coding standard", *Commun. ACM*, Vol. 34, No. 4, April 1991.
[32] M. Menezes de Sequeira and F. Pereira, "Knowledge-based videotelephone sequence segmentation", *VCIP '93*, Vol. 2094, Part 2, November 1993.
[33] MPEG-4 Seminar organized by Dimitri Anastasiou, Columbia University, New York, NY, July 1993.
[34] H.G. Musmann, M. Hötter and J. Ostermann, "Object-oriented analysis-synthesis coding of moving images", *Signal Processing: Image Communication*, Vol. 1, No. 2, October 1989, pp. 117–138.
[35] Y. Nakaya, Y.C. Chuah and H. Harashima, "Model-based/waveform hybrid coding for videotelephone images", *Proc. Internat. Conf. Acoust. Speech Signal Process. '91*, 1991.
[36] Y. Nakaya and H. Harashima, "Model-based/waveform hybrid coding for low-rate transmission of facial images", *IEICE Trans Commun.*, Vol. E75-B, No. 5, May 1992.
[37] V.S. Nalwa, *A Guided Tour of Computer Vision*, Addison-Wesley, Reading, MA, 1993.
[38] H. Nasr, ed., *Automatic Object Recognition*, SPIE Milestone Series, Vol. MS 41, 1991.
[39] A. Ortega, K. Ramchandran and M. Vetterli, "Optimal Trellis-based buffered compression and fast approximations", *Trans. Image Processing*, Vol. 3, No. 1, January 1994, pp. 26–40.
[40] T. Pavlidis, *Structural Pattern Recognition*, Springer, Berlin, 1977.
[41] H. Ueno, K. Dachiku, K. Ohzeki and F. Sugiyama, "A study on facial region detection in the standard video coding method", *Proc. 3rd Internat. Workshop on 64 kbit/s Coding of Moving Video*, 1990.
[42] Y. Wang and O. Lee, "Active mesh – A video representation scheme for feature seeking and tracking", *VCIP '93*, Vol. 2094, Part 3, November 1993.
[43] *Workshop on Very Low Bit Rate Video Compression*, Co-Organizers: T. Huang and M. Orchard, University of Illinois at Urbana-Champaign, 1 May 1993.