Spatiotemporal Segmentation and Tracking of Objects for Visualization of Videoconference Image Sequences

Ioannis Kompatsiaris, *Student Member, IEEE,* and Michael Gerassimos Strintzis, *Senior Member, IEEE*

***Abstract*—In this paper, a procedure is described for the segmentation, content-based coding, and visualization of videoconference image sequences. First, image sequence analysis is used to estimate the shape and motion parameters of the person facing the camera. A spatiotemporal filter, taking into account the intensity differences between consequent frames, is applied, in order to separate the moving person from the static background. The foreground is segmented in a number of regions in order to identify the face. For this purpose, we propose the novel procedure of K-Means with connectivity constraint algorithm as a general segmentation algorithm combining several types of information including intensity, motion and compactness. In this algorithm, the use of *spatiotemporal regions* is introduced since a number of frames are analyzed simultaneousl,y and as a result, the same region is present in consequent frames. Based on this information, a 3-D ellipsoid is adapted to the person's face using an efficient and robust algorithm. The rigid 3-D motion is estimated next using a least median of squares approach. Finally, a Virtual Reality Modeling Language (VRML) file is created containing all the above information; this file may be viewed by using any VRML 2.0 compliant browser.**

***Index Terms*—Model-based image sequence analysis, spatiotemporal image sequence segmentation, VRML.**

I. Introduction

DIGITAL VIDEO is an integral part of many newly emerging multimedia applications. New video coding standards, such as MPEG-4 and MPEG-7, do not concentrate only on efficient compression methods, but also on providing better ways to represent, integrate and exchange visual information [1]–[3]. These efforts aim to provide the user with greater flexibility for "content-based" access and manipulation of multimedia data.

In order to obtain a model-based representation, an input video sequence must first be segmented into an appropriate set of arbitrarily shaped objects (termed the *video object planes* in the MPEG-4 Verification Model), where each of the objects may represent a particular meaningful content of the video stream [4]. The features of each object such as shape, motion, and texture information can subsequently be coded into the so-called *video object layer* for transmission or storage. Although the standards will provide the needed functionalities in order to compose, manipulate and transmit the "object-based" information, the production of these objects is out of the scope of the standards and is left to the content developer. Thus, the success of any object-based approach depends largely on the segmentation of the scene based on its image contents. In a videophone-type application, for example, an accurate segmentation of the face object can serve two purposes: 1) it can allow the encoder to place more emphasis on the facial area since this area (the eyes and mouth and particular) is the focus of attention of the human visual system and 2) it can also be used to extract features so that higher level descriptions can be generated (e.g., personal characteristics, facial expressions, and composition information). In a similar fashion, the contents of a video database can be segmented into individual objects, where the following features can be supported: 1) sophisticated query and retrieval operations; 2) advanced editing and composition; and 3) better compression ratios. These issues and objectives are currently addressed within the framework of the upcoming MPEG-4 and future MPEG-7 standards [2].

Segmentation methods for 2-D images may be divided primarily into region- and boundary-based methods [5]–[8]. Region-based approaches [9], [10] rely on the homogeneity of spatially localized features such as gray-level intensity, texture, motion [11], and other pixel statistics. Region growing and split and merge techniques also belong to the same category. On the other hand, boundary-based methods use primarily gradient information to locate object boundaries. Deformable whole boundary methods [12], [13] rely on the gradient features of parts of an image near an object boundary.

Other techniques include the segmentation by anisotropic diffusion [14] introduced by Perona and Malik [15]. Anisotropic diffusion can be seen as a robust procedure which estimates a piecewise smooth image from a noisy input image. The "edge-stopping" function in the anisotropic diffusion equation, allows the preservation of edges while diffusing the rest of the image. Mathematical morphology [16], [17] methods have been also used for segmentation. In particular, the watershed transformation [18] has received considerable attention in use for image segmentation. This transformation determines the minima of the gradient of the image to be segmented, and associates a segment to each minimum. Conventional gradient operators generally produce many local minima which are caused by noise or quantization errors, and hence, the watershed transformation

Manuscript received December 15, 1998; revised April 21, 2000. This work was supported by the EU CEC Project INTERFACE (IST Project 10036) and by the PENED99 Project of the Greek Secretariat of Science and Technology. This paper was recommended by Associate Editor R. Koenen.

The authors are with the Information Processing Laboratory, Electrical and Computer Engineering Department, Aristotle University of Thessaloniki, Thessaloniki, Greece, and also with the Centre for Research and Technology—Hellas/Informatics and Telematics Institute, Thessaloniki 54639, Greece (e-mail: strintzi@eng.auth.gr).

Publisher Item Identifier S 1051-8215(00)10630-5.

with a conventional gradient operator usually results in oversegmentation. To alleviate this problem, the use of multiscale morphological gradient operators has been proposed. Methods for the segmentation of image sequences have been presented in, among others, [19]–[22]. In most of these methods, region growing and merging techniques are used, depending on the homogeneity of the 2-D or 3-D motion. When advanced multiview setups are available, 3-D models of the scene can be created and the segmentation may be applied at the triangle level of the 3-D model using several characteristics such as intensity, motion, and depth [23].

In the sequel, the term "region" is used to describe an area of the frame which is homogeneous in characteristics such as grey level, texture, motion, or their combinations. The term "object" is used to describe regions or combinations of regions which have a semantic meaning, e.g., the head object. Finally, the facial region is identical to the face object.

In this paper a novel procedure for the segmentation of image sequences using both spatial and temporal information is presented. As a basis for the segmentation algorithm, the K-Means algorithm is used, modified so as to take into account the coherence of the regions. The regularization parameters of the K-Means with Connectivity Constraint (KMC) algorithm are evaluated automatically using the min–max criterion [24]. The algorithm is extended so as to separate and track regions appearing in consequent frames of an image sequence. The methodology proposed here is similar to that proposed in [25], and thus differs significantly from that most common in the literature, where an image is first separated into regions which are then tracked through time [26]. In the proposed approach, a number of consequent frames of the image sequence are analyzed *simultaneously* in order to segment the images into regions. This *higher order segmentation* implicitly solves the problem of correspondence of objects between consequent frames. Following region separation, and depending on the application, knowledge-based methods may be used to decide whether a separated region corresponds to a specific semantic object (e.g., the head or face in a videoconference image sequence).

A direct application of the segmentation into regions which may correspond to meaningful objects is the coding and visualization of these objects. Model-based methods have been used to enable these functionalities. The ability of model-based techniques to describe a scene in a structural way has opened up new areas of applications. Very low bitrate coding, video production, realistic computer graphics, multimedia interfaces, and databases and medical visualization are some of the applications that may benefit by exploiting the potential of model-based schemes [27]–[30].

In this paper, a robust and very efficient method is adopted for fitting a 3-D ellipse to the facial region based on initial 2-D data. The available 3-D model may be used to estimate the rigid 3-D motion and use the small set of rigid 3-D motion parameters in order to update the model in the next time instance and also reconstruct the next frame using only information from the previous one.

These extracted parameters are converted into Virtual Reality Modeling Language (VRML) format so that a moving 3-D representation of the image sequence may be viewed by any VRML compliant browser [31]. The visualization offers enhanced telepresence to the viewer, since a 3-D representation of the scene is created. Furthermore, the user of the VRML browser may interact with the scene; for example the user may rotate the 3-D model in order to see an object from the side. Once the 3-D representation derived from the real image data is available, an environment is created in which it is very easy to integrate synthetic objects. For example, a new background may be added to replace the old one; such a background may be entirely synthetic or extracted from another real image. Also several characteristics of the virtualized environment, such as lighting, may be directly controlled through VRML nodes. This also solves a major problem facing model-based schemes, which is the problem of interportability: since for each model-based scheme, a different specific decoder and viewer is required, the usefulness of such schemes is limited. By adopting VRML, a standardized file format and a general-purpose viewer can be used.

The paper is organized as follows. In the following section, the foreground/background separation procedure and the KMC algorithm are described. In Section III, the min–max criterion is used for the automatic evaluation of the segmentation regularization parameters The final spatiotemporal segmentation algorithm is presented in Section IV. The 3-D ellipsoid adaptation procedure to the person's face is described in Section V. In Section VI, the 3-D model is used for rigid 3-D motion estimation. The visualization process using VRML is described in Section VII-A. Experimental results evaluating the performance of the algorithm are given in Section VIII. Finally, conclusions are drawn in Section IX.

II. Shape Parameter Estimation

A. Foreground/Background Segmentation

For videoconference scenes, it is reasonable to assume that the moving object is a person in front of a static camera. In order to extract the motion information needed to separate the foreground from the background, a spatiotemporal filter, similar to that in [32], is used. The filter takes into account mainly intensity differences between pixels in consequent time instances. In order to smooth the background/foreground segmentation mask and to avoid random erroneous changes, the number of changes of pixel intensity inside a time window T_w is multiplied with a smoothing function as follows: for each pixel $\mathbf{p} = (x, y)$, the following index is calculated:

$$V(\mathbf{p}) = \sum_{k=0}^{\mathcal{F}/T_w} \left(\sum_{t=k\cdot T_w}^{(k+1)\cdot T_w} |I_{t+1}(\mathbf{p}) - I_t(\mathbf{p})| \right) \cdot f(c_{k,\,t}) \quad (1)$$

where

$I_t(\mathbf{p})$ pixel intensity at time t;

$\mathcal{F}$ the last frame used and is an integer multiple of T_w;

$c_{k,\,t}$ a counter incremented by one whenever $I_{t+1}(\mathbf{p}) - I_t(\mathbf{p}) \neq 0$.

Each time a new time window is processed (whenever that is, k is incremented) $c_{k,t}$ is set equal to $-T$, where $T < T_w$. The function $f(c_{k,t})$ is given by

$$f(c_{k,t}) = \begin{cases} 0, & \text{if } c_{k,t} < 0 \\ 1, & \text{if } c_{k,t} \geq 0. \end{cases}$$

As is easily seen, if no more than T changes of pixel intensity occur within a time window, they do not contribute to the final decision, since $f(c_{k,t}) = 0$. A simple thresholding algorithm suffices in order to obtain the final segmentation mask

$$F(\mathbf{p}) = \begin{cases} 1 \; (\mathbf{p} \in \text{ foreground}), & \text{if } V(\mathbf{p}) \geq th \\ 0 \; (\mathbf{p} \in \text{ background}), & \text{if } V(\mathbf{p}) < th \end{cases}$$

where th is a threshold. Thus, $F(\mathbf{p})$ is the background/foreground segmentation mask, indicating whether a pixel belongs to the foreground (i.e., to the moving person), or to the static background. All ensuing operations, described in the sections that follow, are performed only on pixels $\mathbf{p}$ belonging to the foreground (i.e., only on pixels with $F(\mathbf{p}) = 1$).

B. The KMC Algorithm

Clustering based on the K-Means algorithm is a widely used region segmentation method [33]–[35] which, however, tends to produce unconnected regions. This is due to the propensity of the classical K-Means algorithm to ignore spatial information about the intensity values in an image, since it only takes into account the global intensity or color information. In order to alleviate this problem, we propose the use of an extended K-Means algorithm: the KMC algorithm. In this algorithm, the *spatial proximity* of each region is also taken into account by defining a new center for the K-Means algorithm and by integrating the K-Means with a component labeling procedure.

For the sake of easy reference we shall first describe the traditional K-Means (KM) algorithm.

Step 1) For every region s_k, $k = 1, \ldots, K$, random initial intensity values are chosen for the region intensity centers $\overline{I}_k$.

Step 2) For every pixel $\mathbf{p} = (x, y)$, the difference is evaluated between $I(x, y)$ and $\overline{I}_k$, $k = 1, \ldots, K$. If $|I(x, y) - \overline{I}_i| < |I(x, y) - \overline{I}_k|$ for all $k \neq i$, $\mathbf{p}(x, y)$ is assigned to region s_i.

Step 3) Following the new subdivision, $\overline{I}_k$ is recalculated. If M_k elements are assigned to s_k then

$$\overline{I}_k = \frac{1}{M_k} \sum_{m=1}^{M_k} I(\mathbf{p}_m^k) \tag{2}$$

where $\mathbf{p}_m^k$, $m = 1, \ldots, M_k$, are the pixels belonging to region s_k.

Step 4) If the new $\overline{I}_k$ are equal with the old then stop, else goto *Step 2*.

The results of the application of the above algorithm are improved using the KMC algorithm, which consists of the following steps.

Step 1) The classical KM algorithm is performed for a small number of iterations. This result in K regions, with intensity centers $\overline{I}_k$, as described above, and spatial centers $\overline{\mathbf{S}}_k = (\overline{S}_{k,x}, \overline{S}_{k,y})$, $k = 1, \ldots, K$

$$\overline{S}_{k,x} = \frac{1}{M_k} \sum_{m=1}^{M_k} p_{m,x}^k$$

$$\overline{S}_{k,y} = \frac{1}{M_k} \sum_{m=1}^{M_k} p_{m,y}^k \tag{3}$$

where $\mathbf{p}^k = (p_x^k, p_y^k)$. The differential motion centers $\overline{V}_k$ are defined by

$$\overline{V}_k = \frac{1}{M_k} \sum_{m=1}^{M_k} |I(\mathbf{p}_{m,t+1}^k) - I(\mathbf{p}_{m,t}^k)| \tag{4}$$

where $\mathbf{p}_{m,t}^k$, $m = 1, \ldots, M_k$ are the pixels of the kth region at time t. The area of each region A_k is defined by

$$A_k = M_k$$

and the mean area of all regions

$$\overline{A} = \frac{1}{K} \sum_{k=1}^{K} A_k.$$

Step 2) For every pixel $\mathbf{p} = (x, y)$, the intensity differences are evaluated between center and pixel intensities as well as the distances between $\mathbf{p}$ and $\overline{\mathbf{S}}$ and $V(\mathbf{p})$ and $\overline{V}_k$. A generalized distance of a pixel $\mathbf{p}$ from a region s_k is defined as follows:

$$D(\mathbf{p}, k) = \frac{\lambda_1}{\sigma_I^2} ||I(\mathbf{p}) - \overline{I}_k|| + \frac{\lambda_2}{\sigma_V^2} ||V(\mathbf{p}) - \overline{V}_k|| + \frac{\lambda_3}{\sigma_S^2} \overline{A} \frac{||\mathbf{p} - \overline{\mathbf{S}}_k||}{A_k}$$

where $||\mathbf{p} - \overline{\mathbf{S}}_k||$ is the Euclidean distance, $V(\mathbf{p})$ is now simply defined as $V(\mathbf{p}) = |I(\mathbf{p}_{t+1}) - I(\mathbf{p}_t)|$, σ_I, σ_V, σ_S are the standard deviations of intensity, motion, and spatial distance, respectively, and λ_1, λ_2, λ_3 are regularization parameters. Normalization of the spatial distance $||\mathbf{p} - \overline{\mathbf{S}}_k||$ with the area of each region $\overline{A}/A_k$ is necessary in order to allow the creation of large connected regions; otherwise, pixels with similar intensity and motion values with those of a large region would be assigned to neighboring smaller regions. If $|D(\mathbf{p}, i)| < |D(\mathbf{p}, k)|$ for all $k \neq i$, $\mathbf{p} = (x, y)$ is assigned to region s_i.

Step 3) Based on the above subdivision, an eight connectivity component labeling algorithm is applied. This algorithm finds all connected components and assigns a unique value to all pixels in the same component. Regions whose area remains below a predefined threshold are not labeled as separate ones. The component labeling algorithm produces L connected regions. For these connected regions, the intensity $\overline{I}_l$, spatial $\overline{\mathbf{S}}_l$, and motion centers $\overline{V}_l$, $l = 1, \ldots, L$ are calculated using equations (2)–(4), respectively.

Step 4) If the difference between the new and the old centers $\overline{\mathbf{I}}_l$, $\overline{\mathbf{S}}_l$, and $\overline{V}_l$ is below a threshold

$$\frac{1}{L}\sum_{l=1}^{L}\left(\frac{\lambda_1}{\sigma_I^2}\|\overline{I}_l^i - \overline{I}_l^{i-1}\| + \frac{\lambda_2}{\sigma_V^2}\|\overline{V}_l^i - \overline{V}_l^{i-1}\| + \frac{\lambda_3}{\sigma_S^2}\overline{A}\,\frac{\|\overline{\mathbf{S}}_k^i - \overline{\mathbf{S}}_k^{i-1}\|}{A_k}\right) \le threshold$$

where C^i is the corresponding center at the ith iteration of the algorithm, then stop, else goto *Step 2* with $K = L$ using the new intensity, motion, and spatial centers.

Through the use of this algorithm, the ambiguity in the selection of the number K of regions, which is another weakness of the K-Means algorithm, is also resolved. Starting from any K, the component labeling algorithm produces or rejects regions according to their compactness. In this way K is automatically adjusted during the segmentation procedure.

An example of the segmentation procedure is shown in Fig. 1. Fig. 1(a) shows the original image of the videoconference sequence "Claire" of size 176×144. Fig. 1(b) shows the result of the first iteration of the KMC algorithm (result of Step 2 of the KMC algorithm, for the first iteration). The result of the component labeling algorithm (Step 3 of the algorithm) is in Fig. 1(c). The initial number of regions was set to $K = 5$ and the component labeling algorithm produced $L = 6$ regions. Fig. 1(d) shows the final segmentation after only four iterations.

After the segmentation procedure, a face-detection algorithm must be applied in order to identify the region that corresponds to the facial area. For this purpose, the algorithm presented in [36] can be used. In [36], the facial area is detected among other candidate regions using fuzzy functions and information including relative size and position.

III. Selection of the Regularization Parameters

The segmentation procedure described in Section II-B produces a segmentation $\mathcal{S}^*$ by assigning each pixel of the foreground object to one of the L regions $s_l, l = 1, \ldots, L$. This segmentation minimizes the following energy function [37], [38]

$$E(\mathcal{S}, \boldsymbol{\lambda}) = \sum_{l=1}^{L}\sum_{p \in s_l} \boldsymbol{\lambda}^T \mathbf{D}(\mathbf{p}, l) \tag{5}$$

where

$$\boldsymbol{\lambda} = \begin{bmatrix} \dfrac{\lambda_1}{\sigma_I^2} \\ \dfrac{\lambda_2}{\sigma_V^2} \\ \dfrac{\lambda_3}{\sigma_S^2} \end{bmatrix}, \quad \mathbf{D}(\mathbf{p}, l) = \begin{bmatrix} \|\mathbf{I}(\mathbf{p}^l) - \overline{I}_l\| \\ \|V(\mathbf{p}^l) - \overline{V}_l\| \\ \dfrac{A_l}{\overline{A}}\|\mathbf{p}^l - \overline{\mathbf{S}}_l\| \end{bmatrix}.$$

Clearly, the final segmentation depends on the regularization parameter $\boldsymbol{\lambda}$. In our case, setting $\lambda_1 \gg \lambda_2$ emphasizes the importance of intensity information and encourages the production of many unconnected regions. By contrast, setting $\lambda_2 \gg \lambda_1$, regions with similar motion are created. In both cases, if $\lambda_1 + \lambda_2 \gg \lambda_3$ the segmentation results in a partition of L connected regions regardless of the intensity or motion information. The weight factor $\boldsymbol{\lambda}$ must be constrained, because otherwise, as $\boldsymbol{\lambda}$ increases without limit, so does the energy $E(\mathcal{S}, \boldsymbol{\lambda})$. Thus, we set

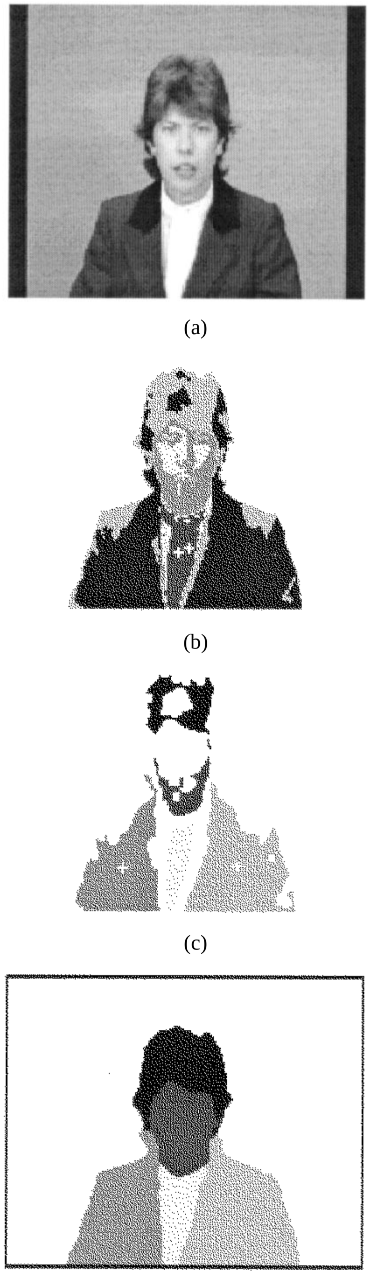

(a)
(b)
(c)
(d)

Fig. 1. (a) Original image "Claire." (b) Result of the KMC algorithm (Step 2, first iteration). (c) Result of the component labeling algorithm (Step 3). (d) Final segmentation after only four iterations.

$$\|\boldsymbol{\lambda}\| = 1.$$

This parameter may be chosen heuristically or by using *a priori* knowledge. Alternately, the *min–max criterion* [24], [39]

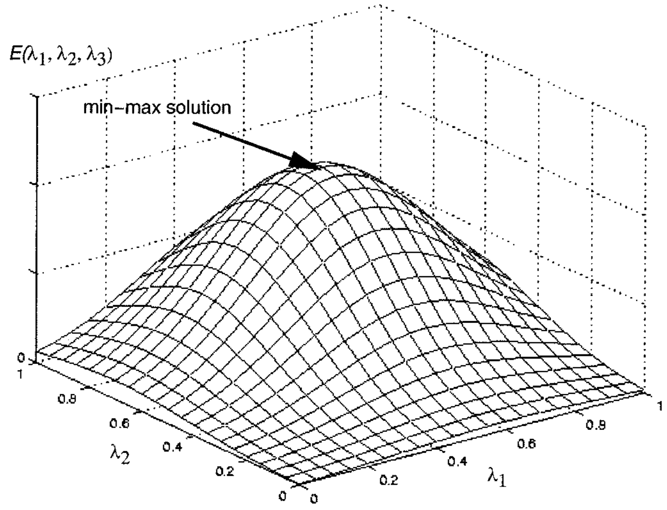


Fig. 2. Min–max selection of the regularization parameters $\boldsymbol{\lambda}$.

can be used for automatically evaluating the best $\boldsymbol{\lambda}$. This criterion is based on the assumption that in situations with conflicting alternatives, the most rational strategy is the one aiming to minimize the maximum possible losses. Thus, the function is minimized over all possible combinations of weight values and the min–max criterion selects the combination producing the maximum of these minima.

In this case, the application of the min–max criterion gives (Fig. 2)

$$\begin{aligned} &E(\mathcal{S}^*, \boldsymbol{\lambda}^*) \geq E(\mathcal{S}^*, \boldsymbol{\lambda}), \quad \text{if } \|\boldsymbol{\lambda}\| = 1,\ \boldsymbol{\lambda} \neq \boldsymbol{\lambda}^* \\ &E(\mathcal{S}^*, \boldsymbol{\lambda}^*) \leq E(\mathcal{S}, \boldsymbol{\lambda}^*), \quad \text{if } \mathcal{S} \neq \mathcal{S}^* \end{aligned}$$

The resulting min–max segmentation $E(\mathcal{S}^*, \boldsymbol{\lambda}^*)$ will be seen to yield very satisfactory results for the segmentation of practical videoconference sequences.

IV. Spatiotemporal Segmentation

The segmentation procedure, along with the selection of the regularization parameters described above, can be easily extended so as to separate and track regions appearing in consequent frames of an image sequence. The methodology proposed here differs from that, most common in the literature, where an image is first separated into regions and then these regions are tracked through time. In the proposed approach, a number of consequent frames of the image sequence are analyzed *simultaneously* in order to segment the images into regions.

If a small number T of consequent frames is used, it is reasonable to assume that the region intensities remain substantially the same and hence that their intensity centers $\overline{I}_k$ are independent of time. In this way, the region s_k is composed of all pixels $\mathbf{p}_t^k$ from T consequent frames. The KMC algorithm presented in Section II-B can be applied for a number of frames using the generalized distance $D(\mathbf{p}_t, k)$, $t = 1, \ldots, T$ and the intensity centers

$$\overline{I}_k = \frac{1}{M_k} \sum_{m=1}^{M_k} I(\mathbf{p}_{m,t}^k)$$

where $\mathbf{p}_{m,t}^k$ are all pixels belonging to region s_k at time t. A similar assumption of independence of time for the spatial and motion centers cannot be made, since the region may move or move with different velocity from frame to frame (Fig. 3). Thus

$$\overline{\mathbf{S}}_{k,t} = \frac{1}{M_k} \sum_{m=1}^{M_k} \mathbf{p}_{m,t}^k$$

where only the pixels belonging to region k at frame t contribute to the estimation of the spatial center of subobject k at frame t. Similarly

$$\overline{V}_{k,t} = \frac{1}{M_k} \sum_{m=1}^{M_k} \|I(\mathbf{p}_{m,t}^k) - I(\mathbf{p}_{m,t+1}^k)\|.$$

The energy measure to be minimized becomes

$$E(\mathcal{S}) = \sum_{l=1}^{L} \sum_{p \in s_l} \boldsymbol{\lambda}^T \mathbf{D}(\mathbf{p}_t^l, l) \tag{6}$$

where

$$\mathbf{D}(\mathbf{p}_t^l, l) = \begin{bmatrix} \|I(\mathbf{p}_t^l) - \overline{I}_l\| \\ \|V(\mathbf{p}_t^l) - \overline{V}_{l,t}\| \\ \dfrac{A_l}{\overline{A}} \|\mathbf{p}_t^l - \overline{\mathbf{S}}_{l,t}\| \end{bmatrix}.$$

A temporal component labeling algorithm is performed at Step 3 of the algorithm and the connectivity is now checked in three dimensions: x, y, and time t. The algorithm produces connected regions appearing in consequent frames.

V. 3-D Shape Parameter Estimation

A straightforward application of the above spatiotemporal segmentation algorithm is in the coding of the facial area in videoconference image sequences. The procedure in [36] is used to extract the facial area in consequent regions. This procedure employs a number of fuzzy functions of characteristics such as color, size and spatial position of the regions in order to detect the face region. The 3-D shape of the face is modeled by a 3-D ellipsoid which best fits the face in each consequent frame. Then, 3-D motion between each ellipsoid is estimated and finally only information concerning the first frame and motion parameters for consequent frames need to be transmitted; the following frames are obtained from the previous ones using motion compensation. The use of 3-D ellipse tracking gives better results than 2-D motion estimation as is also observed in [40].

The 3-D ellipse calculated must fit to the boundaries of the face and at the same time provide a unique correspondence relation between 3-D points belonging to the 3-D ellipse and 2-D pixel locations on the image. The general equation of a 3-D ellipsoid is

$$F(X, Y, Z) = \mathbf{a}^T \mathbf{Q} = 0 \tag{7}$$

where $\mathbf{a} = [a_1 \ldots a_{10}]^T$ is 10×1 vector and

$$\mathbf{Q} = [XY \quad Y^2 \quad YZ \quad Z^2 \quad ZX \quad X \quad Y \quad Z \quad 1]^T.$$

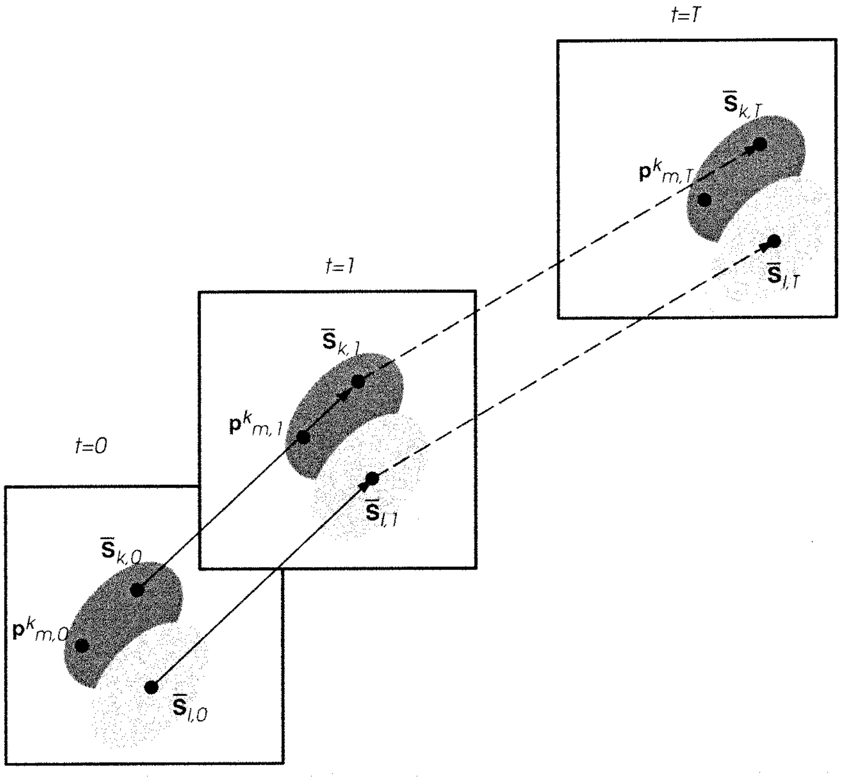


Fig. 3. Spatiotemporal segmentation of two regions.

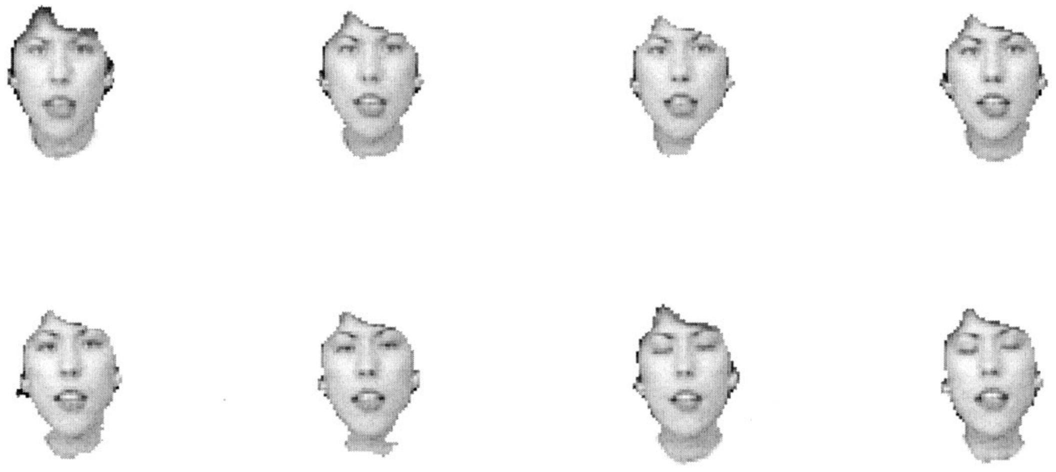


Fig. 4. Resulting segmentation of the facial region for the first eight frames of the “Claire” sequence.

In [41], a solution for (7) was presented, using the 3-D nodes of a known wireframe. The method is a generalization of the extremely efficient method for 2-D ellipse fitting presented in [42]. In this approach, no 3-D information is available and (7) is projected to the 2-D image plane, assuming a perspective projection (Fig. 5)

$$x = f\,\frac{X}{Z}, \quad y = f\,\frac{Y}{Z}$$

where f represents the camera focal length. Thus, multiplying (7) by f^2/Z^2 the following equation is obtained:

$$F(x, y, Z, f) = \mathbf{a}^T\mathbf{q} = 0 \tag{8}$$

where

$$\mathbf{q} = \left[x^2 \;\; xy \;\; y^2 \;\; fy \;\; f^2 \;\; fx \;\; \frac{f}{Z}x \;\; \frac{f}{Z}y \;\; \frac{f^2}{Z^2}\right]^T.$$

In the above equation, the unknown parameters $\mathbf{a}$ are those of the 3-D ellipsoid. Since for ellipse fitting, only the pixels with coordinates (x, y) lying on the boundaries of the head region are used, it can be assumed that all these points have the same depth Z, which can be arbitrarily chosen if only relative depth is needed. As a result, the ellipse fitting procedure will provide a relative depth information based on the scale on the Z-axis. In this way, the technique in [41] produces 3-D ellipses fitting

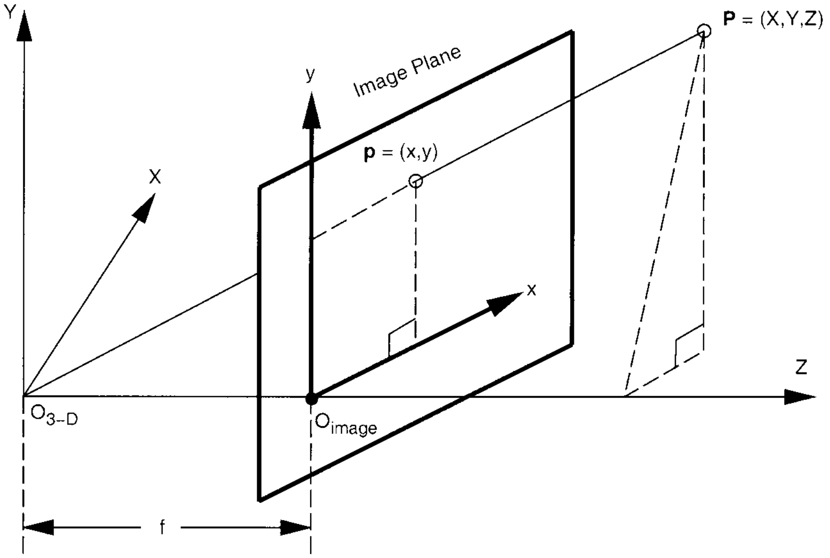


Fig. 5. Perspective camera geometry.

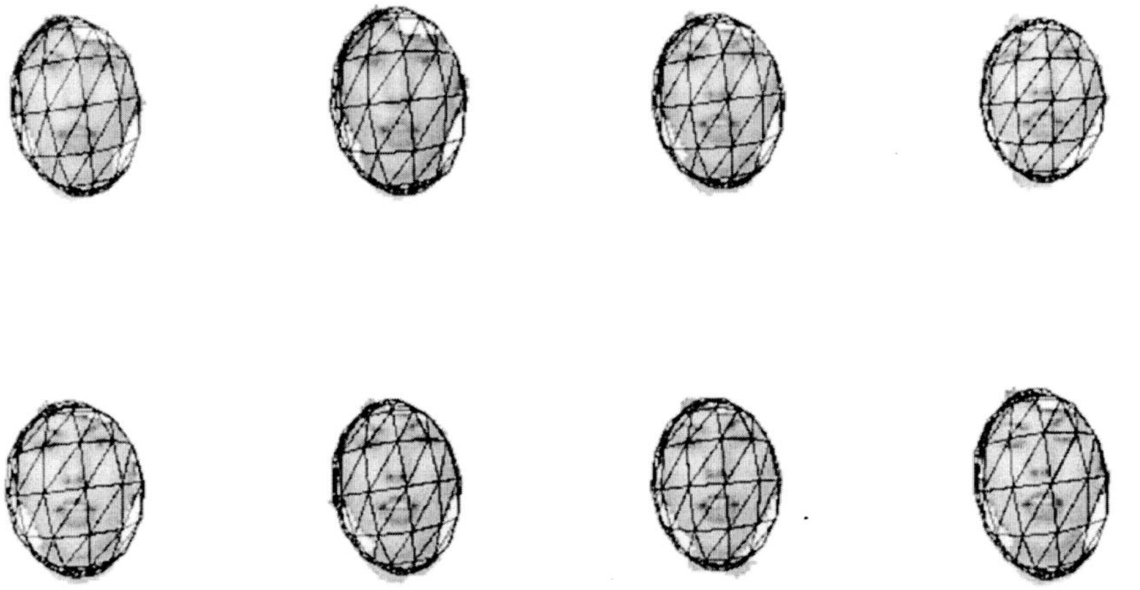


Fig. 6. The resulting ellipse fitting to the facial region for the eight first frames of the “Claire” sequence.

using the 2-D coordinates of the object boundaries, found in the previous section.

VI. Rigid 3-D Motion Estimation

The available 3-D model may be used to estimate the rigid 3-D motion, update the model in the next time instance using also a small set of rigid 3-D motion parameters, and also reconstruct the next frame using information from the previous one.

The motion of an arbitrary point $\mathbf{P}(t)$ to its new position $\mathbf{P}(t+1)$ is described by [43]

$$\mathbf{P}(t+1) = \mathbf{R}[\mathbf{P}_w(t) - \mathbf{G}(t)] + \mathbf{T}_w + \mathbf{G}(t) \qquad (9)$$

where

$$\mathbf{R} = \begin{bmatrix} 1 & -\omega_Z & \omega_Y \\ \omega_Z & 1 & -\omega_X \\ -\omega_Y & \omega_X & 1 \end{bmatrix}, \quad \mathbf{T}_w = \begin{bmatrix} T_{X,\,w} \\ T_{Y,\,w} \\ T_{Z,\,w} \end{bmatrix}$$

$\mathbf{P}_w(t)$ is a 3-D point in world coordinates and $\mathbf{G}(t) = (G_X(t),\, G_Y(t),\, G_Z(t))$ is the center of the 3-D model

$$\mathbf{G}(t) = \frac{1}{N} \sum_{i=1}^{N} \mathbf{P}_w(t)$$

where N is the number of 3-D points of the 3-D ellipsoid model. Setting

$$\mathbf{P}(t) = \mathbf{P}_w(t) - \mathbf{G}(t) = [\,X(t) \quad Y(t) \quad Z(T)\,]^T$$

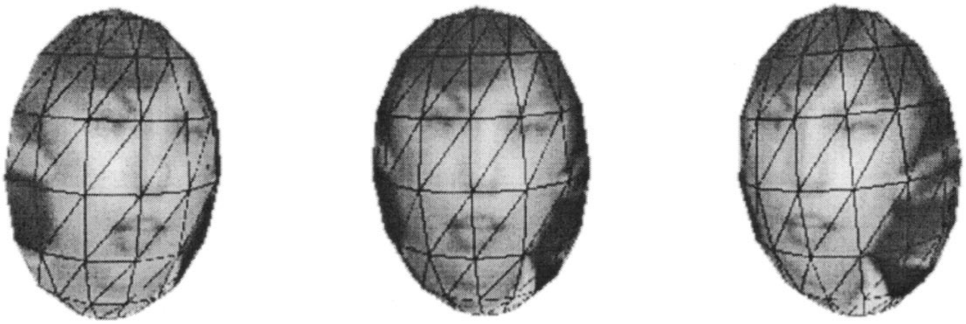
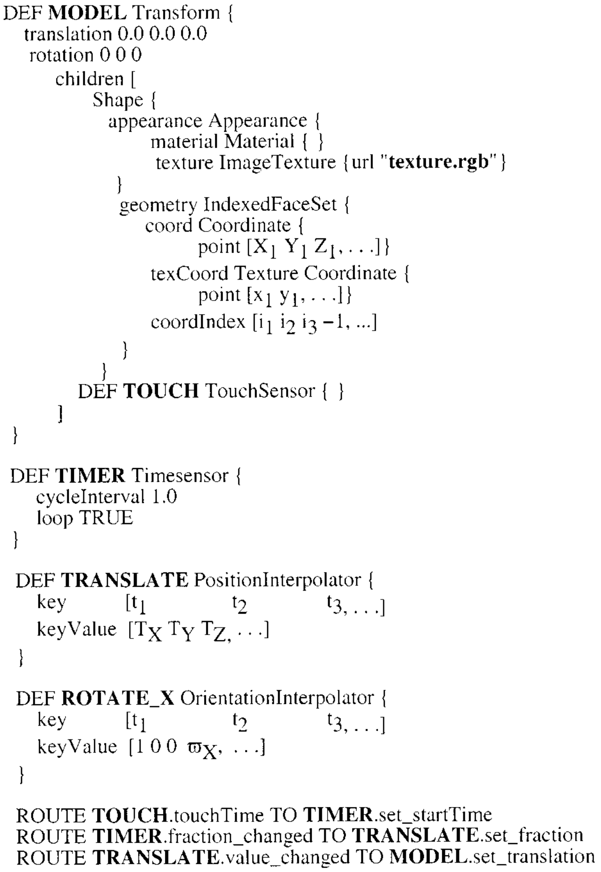


Fig. 7. Different views of the resulting 3-D textured model for “Claire.”



```
DEF MODEL Transform {
  translation 0.0 0.0 0.0
   rotation 0 0 0
      children [
         Shape {
           appearance Appearance {
               material Material { }
                texture ImageTexture {url "texture.rgb"}
            }
            geometry IndexedFaceSet {
              coord Coordinate {
                   point [X1 Y1 Z1, . . .]}
               texCoord Texture Coordinate {
                   point [x1 y1, . . .]}
               coordIndex [i1 i2 i3 -1, ...]
            }
         }
       DEF TOUCH TouchSensor { }
    ]
}

DEF TIMER Timesensor {
   cycleInterval 1.0
   loop TRUE
}

 DEF TRANSLATE PositionInterpolator {
   key        [t1           t2           t3, . . .]
   keyValue  [TX TY TZ, . . .]
 }

 DEF ROTATE_X OrientationInterpolator {
   key        [t1           t2           t3, . . .]
   keyValue  [1 0 0  ϖX, . . .]
 }

 ROUTE TOUCH.touchTime TO TIMER.set_startTime
 ROUTE TIMER.fraction_changed TO TRANSLATE.set_fraction
 ROUTE TRANSLATE.value_changed TO MODEL.set_translation
```

Fig. 8. A sample VRML 2.0 file describing all necessary parameters in order to represent the videoconference scene in 3-D model-based form.

and

$$\mathbf{T} = \mathbf{T}_w + \mathbf{G}(t) = [T_X \quad T_Y \quad T_Z]^T$$

expression (9) reduces to

$$\mathbf{P}(t+1) = \mathbf{R}\mathbf{P}(t) + \mathbf{T}. \tag{10}$$

The goal of the 3-D motion estimation procedure is to compute the parameter vector $(\omega_X, \omega_Y, \omega_Z, T_X, T_Y, T_Z)$. The procedure commences with block-based initial 2-D motion estimation [19], [44], [45].

If (x, y) are the coordinates of the perspective projection of the 3-D point $(X(t), Y(t), Z(t))$ on the image plane at time t, then

$$x = f\,\frac{X(t)}{Z(t)} \quad \text{and} \quad y = f\,\frac{Y(t)}{Z(t)}. \tag{11}$$

From (10) and (11), the 2-D motion vectors $\mathbf{v}(x, y)$ that correspond to the pixels (x, y) of each object are defined by projection of the 3-D motion on the 2-D image plane, as follows:

$$\begin{aligned} u_x(x, y) &= x(t-1) - x(t) \\ &= f\,\frac{x - \omega_Z y + f\omega_Y + fT_X/Z(t)}{-\omega_Y x + \omega_X y + f + fT_Z/Z(t)} - x, \end{aligned} \tag{12}$$

$$\begin{aligned} u_y(x, y) &= y(t-1) - y(t) \\ &= f\,\frac{\omega_Z x + y - f\omega_X + fT_Y/Z(t)}{-\omega_Y x + \omega_X y + f + fT_X/Z(t)} - y. \end{aligned} \tag{13}$$

Equations (12) and (13) are derived by writing (10) in analytical form, multiplying with $f/Z(t)$ and using (11). Using (12) and (13) and assuming that initially u_x and u_y equal the initial block-based motion displacements u_{bx} and u_{by}, respectively, the following system for the model parameters is obtained:

$$\begin{aligned} &y(x+u_{bx})\omega_X - (x(x+u_{bx}) + f^2)\omega_Y + fy\omega_Z - \frac{f^2}{Z(t)}T_X \\ &\quad + \frac{f}{Z(t)}(x+u_{bx})T_Z = -fu_{bx}, \end{aligned} \tag{14}$$

$$\begin{aligned} &(y(y+u_{by}) + f^2)\omega_X - x(y+u_{by})\omega_Y - fx\omega_Z - \frac{f^2}{Z(t)}T_Y \\ &\quad + \frac{f}{Z(t)}(y+u_{by})T_Z = -fu_{by}. \end{aligned} \tag{15}$$

In the first stage, (14) and (15) are used for N of the initially estimated 2-D vectors, forming a system of $2 \times N$ equations and six unknowns. With $N \geq 3$, this system is overdetermined and can be solved using least median of squares methods [46]. Thus, if $\mathbf{m} = (\omega_X, \omega_Y, \omega_Z, T_X, T_Y, T_Z)$ is the parameter vector, the parameter estimation problem is of the form

$$\mathbf{A}\mathbf{m} = \mathbf{b} \tag{16}$$

where $\mathbf{A}$ is a $2N \times 6$ matrix, N is the number of pixels in the region and $\mathbf{b}$ is a $2N$ component vector

$$\mathbf{A} = [\mathbf{A_x} \;\; \mathbf{A_y}]^T, \quad \mathbf{b} = [\mathbf{b_x} \;\; \mathbf{b_x}]^T$$

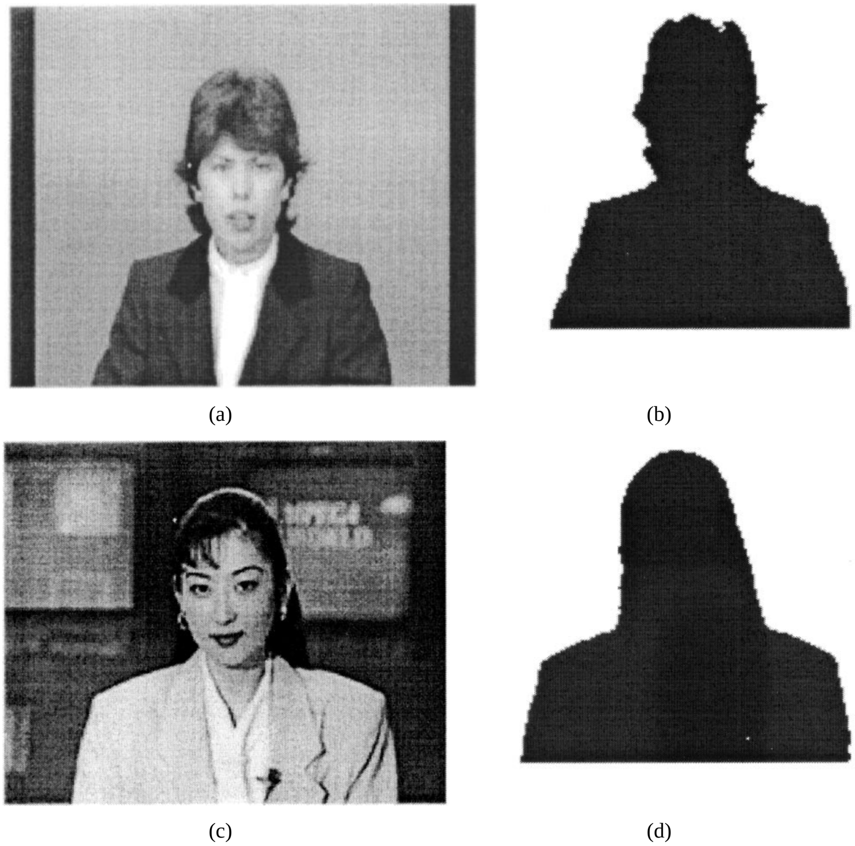


Fig. 9. (a) Original image "Claire." (b) Background / foreground segmentation mask for "Claire." (c) Original image "Akiyo." (d) Background/foreground segmentation mask for "Akiyo."

where we get (17) and (18), as shown at the bottom of the page, and

$$\begin{aligned}\mathbf{b_x} &= \left[-fu_{bx}^{(1)} \ldots -fu_{bx}^{(N)}\right]^T \\ \mathbf{b_x} &= \left[-fu_{by}^{(1)} \ldots -fu_{by}^{(N)}\right]^T\end{aligned} \tag{19}$$

where $\mathbf{v}_b^{(i)} = \mathbf{v}_b(x_i, y_i)$ is the initial displacement vector at pixel (x_i, y_i), $i = 1, \ldots, N$ and $Z_i(t)$ is the corresponding depth.

For each set of N_r, where $3 < N_r < N$, randomly selected points the following least squares solution is estimated for (16)

$$\begin{aligned}\mathbf{m}_j &= (\mathbf{A^T A})^{-1}\mathbf{A^T b} \\ &= (\mathbf{A_x^T A_x} + \mathbf{A_y^T A_y})^{-1}(\mathbf{A_x^T b_x} + \mathbf{A_y^T b_y}).\end{aligned} \tag{20}$$

where $j = 1, \ldots, N_r$. The least median of squares solution of (16) is

$$\mathbf{m} = \arg\min_{\mathbf{m}_j} \text{med}\, ||\mathbf{A}\mathbf{m}_j - \mathbf{b}||. \tag{21}$$

$$\mathbf{A_x} = \begin{bmatrix} y_1\left(x_1 + u_{by}^{(1)}\right) & -\left(x_1\left(x_1 + u_{bx}^{(1)}\right) + f^2\right) & fy_1 & -\dfrac{f^2}{Z_1(t)} & 0 & \dfrac{f}{Z_1(t)}\left(x_1 + u_{bx}^{(1)}\right) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ y_N\left(x_N + u_{by}^{(N)}\right) & -\left(x_N\left(x_N + u_{bx}^{(N)}\right) + f^2\right) & fy_N & -\dfrac{f^2}{Z_N(t)} & 0 & \dfrac{f}{Z_N(t)}\left(x_N + u_{bx}^{(N)}\right) \end{bmatrix} \tag{17}$$

$$\mathbf{A_y} = \begin{bmatrix} y_1\left(y_1 + u_{by}^{(1)}\right) + f^2 & -x_1\left(y_1 + u_{by}^{(1)}\right) & -fx_1 & 0 & -\dfrac{f^2}{Z_1(t)} & \dfrac{f}{Z_1(t)}\left(y_1 + u_{by}^{(1)}\right) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \\ y_N\left(y_N + u_{by}^{(N)}\right) + f^2 & -x_N\left(y_N + u_{by}^{(N)}\right) & -fx_N & 0 & -\dfrac{f^2}{Z_N(t)} & \dfrac{f}{Z_N(t)}\left(y_N + u_{by}^{(N)}\right) \end{bmatrix} \tag{18}$$

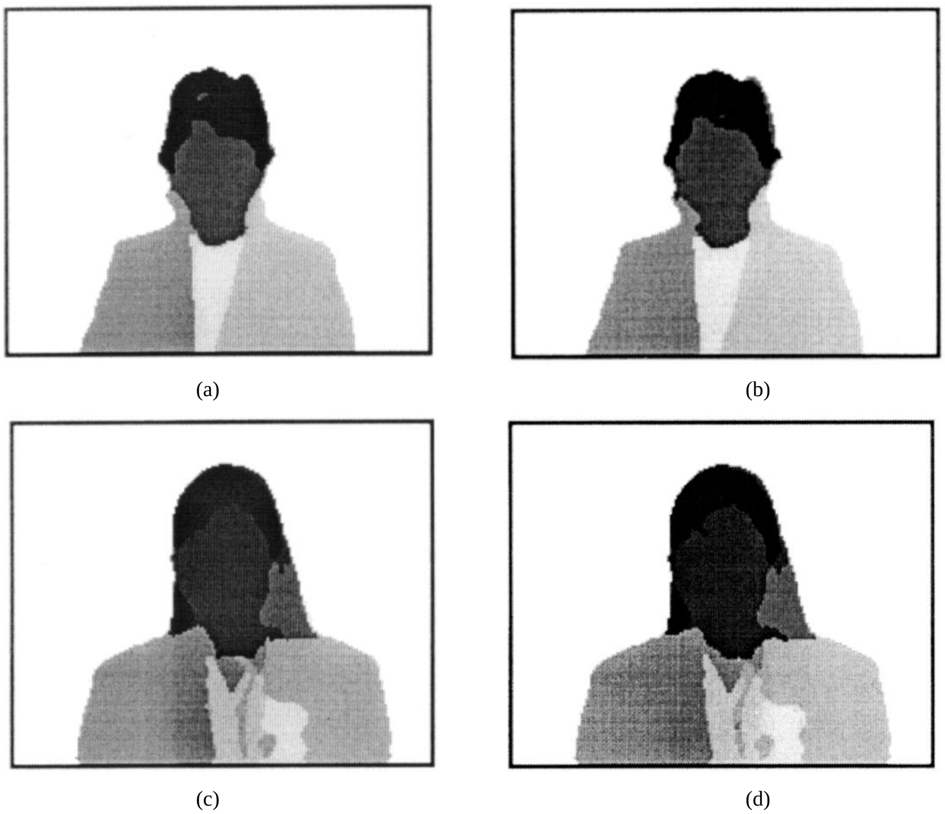


Fig. 10. Segmentation of the: (a) first and (b) eighth frame of the "Claire" sequence and (c) first and (d) eighth frame of the "Akiyo" sequence.

VII. Visualization

A. Visualization With VRML 2.0

VRML is a file format for describing 3-D virtual objects and interactive environments on the World Wide Web[1]. The first version, VRML 1.0, allowed the building static 3-D worlds with limited interactivity. The 3-D world is described by a scene graph which consists of nodes representing virtual objects with their geometry and textures maps and scene descriptors such as transformations of objects, as well as lighting and shading. The geometry is defined in a XYZ coordinate system, together with some simple shapes, such as cubes, cones, cylinders, spheres, and polygon meshes. Transformations allow positioning and scaling of virtual objects in relation to others. Lighting, shading, and texturing are used to add realism to the 3-D scene.

The most recent version, VRML 2.0, provides enhanced static worlds together with interaction, animation and scripting. Animation can be obtained by nodes called interpolators. Similarly, the script nodes allow animation of objects through events they generate. In this work, we used the *IndexedFaceSet, Transform, PositionInterpolator, OrientationInterpolator, TimeSensor,* and *Route* nodes in order to visualize all the information extracted from the image sequence analysis. The *IndexedFaceSet* node is used to describe the geometry of the 3-D model, consisting of 3-D points and their triangular connections. The *IndexedFaceSet* node also contains an image for texture mapping onto the shape. The texture mapping is controlled by the texture coordinates, which take values in $[0.0,\ 1.0]$ and are specified in the 2-D texture space $(s,\ t)$ for each vertex of the shape. The interpolator nodes enable incorporating animation into the VRML 2.0 scene. An interpolator node takes a set of key values, and generates an interpolated value at the specified time instance using these key values. The *TimeSensor*, being a sensor node, keeps track of time instant and generates events as time passes. Typically, it is up for animations, periodic utilities, or timed events. Each of the two objects, head and shoulders, are grouped under a different *Transform* node, which controls their position in the 3-D space. The rigid motion parameters of each sub-object are stored as key values in the *PositionInterpolator* and *OrientationInterpolator* nodes and using the *TimeSensor* node and the *Route* node, those values are applied at regular intervals to the head and shoulders objects through the *Transform* node. The *PositionInterpolator* node holds the translation parameters, while the *OrientationInterpolator* node holds the rotation parameters.

All parameters needed in order to represent the videoconference scene in model-based form, 3-D model, motion, and texture, can be stored and viewed in VRML format, as can be seen in Fig. 8. This solves a major problem facing model-based schemes, which is the problem of interportability: since for each model-based scheme a different specific decoder and viewer is required, the usefulness of such schemes is limited. By adopting VRML, a standardized file format and a general-purpose viewer can be used. Further, the coding of the extracted parameters is standardized following the expected standard VRML compression format.

B. Visualization with MPEG-4 BIFS

Alternatively, for the visualization and transmission of all extracted parameters, the MPEG-4 standard could be used,

[1] Web3D Consortium website, available at: http://www.vrml.org

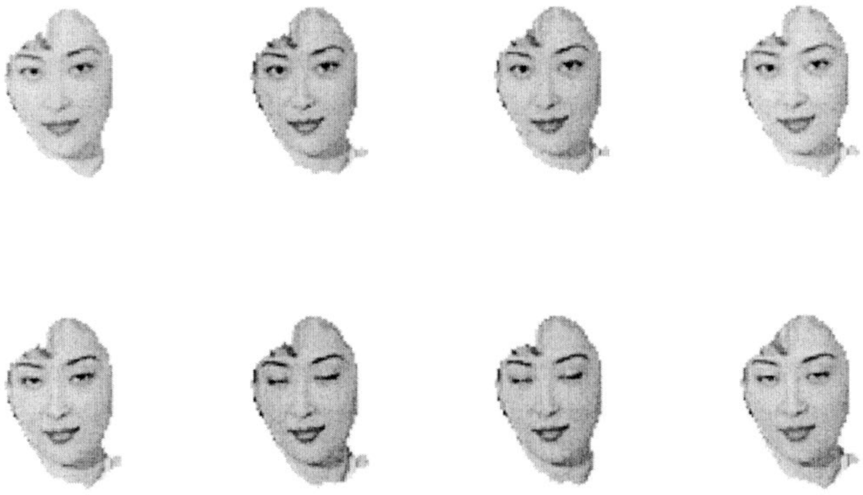


Fig. 11. Segmentation of the facial region for the eight first frames of the "Akiyo" sequence.

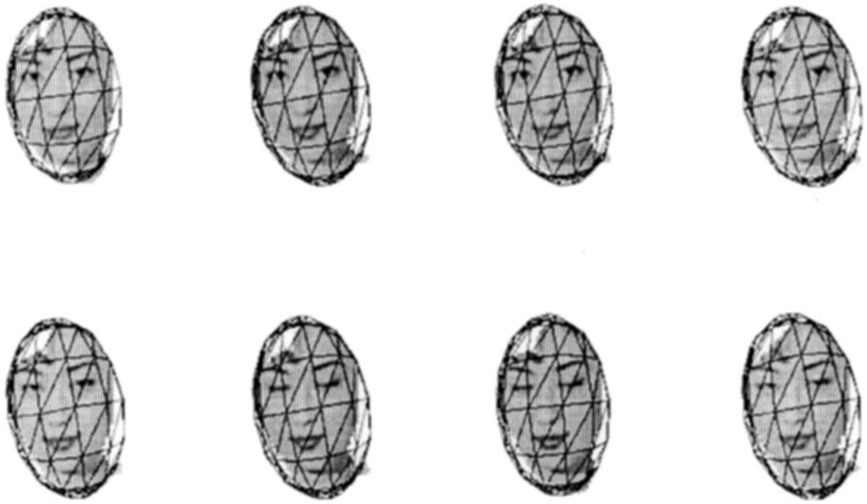


Fig. 12. Ellipse fitting to the facial region for the eight first frames of the "Akiyo" sequence.

and more specifically, the scene description format, the BInary Format for Scene description (BIFS). This scene description is a result of a common work of MPEG-4 and VRML community and has a number of added functionalities, including, among others:

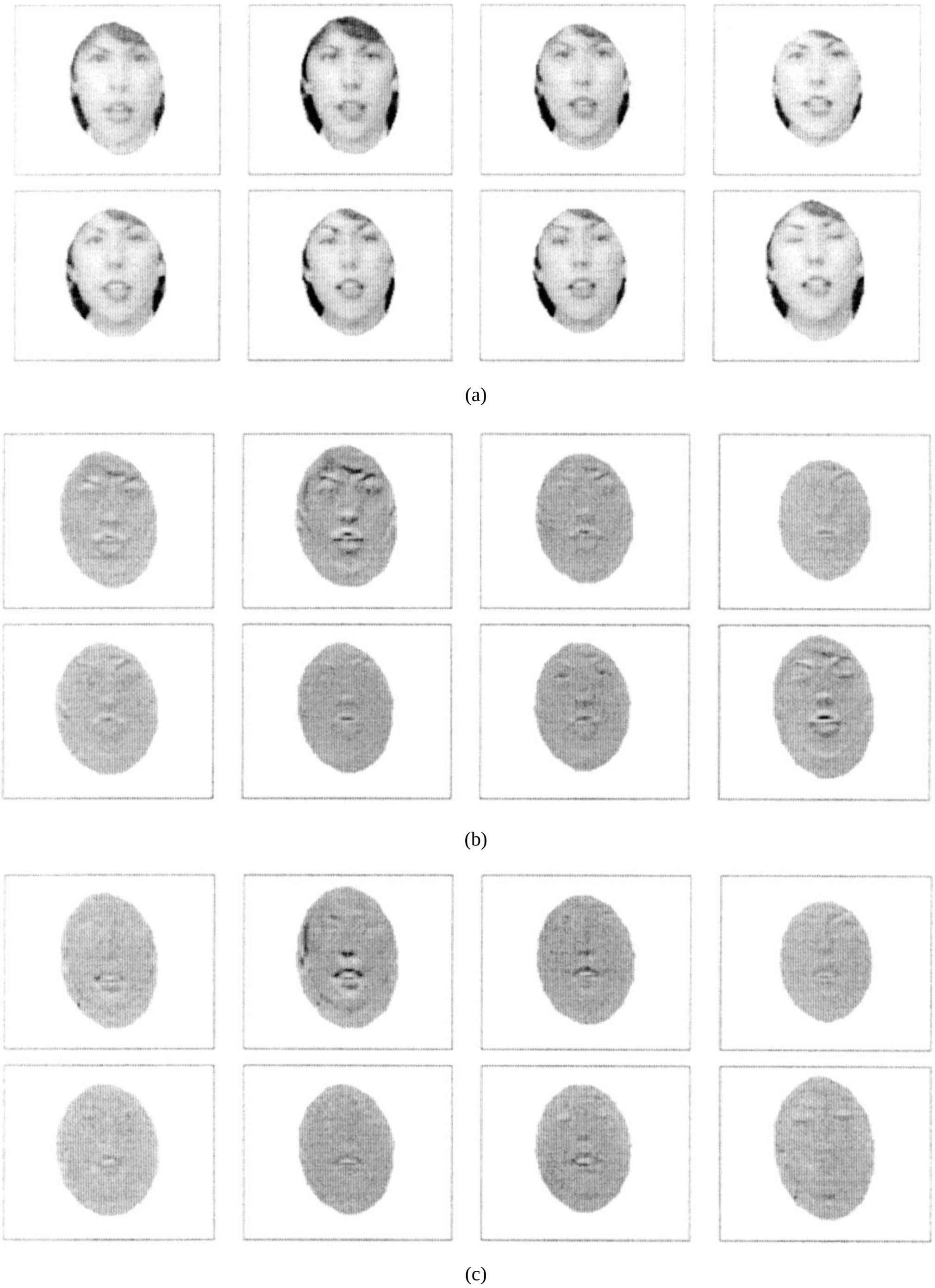
1) efficient data compression (more efficient compression comparing to binary VRML);
2) 2-D nodes and mixing 2-D and 3-D nodes;
3) encapsulation in streams, making the BIFS suitable for broadcast environment.

All nodes described and used above could be used with minor modifications in order to make the visualization MPEG-4 compatible.

VIII. Experimental Results

The algorithm described above was used for the segmentation and coding of two videoconference image sequences, "Claire" and "Akiyo" of size 176×144. The original images are shown in Fig. 9(a) and (c). The spatiotemporal filter as described in Section II-A was applied to a number of frames, producing the foreground/background segmentation mask for each image sequence shown in Fig. 9(b) and (d). The KMC algorithm was applied next in the foreground object. First, the algorithm was applied to one frame only in order to estimate the regularization parameters $\boldsymbol{\lambda}$, as described in Section III. The algorithm was then used for spatiotemporal segmentation (Section IV) producing meaningful objects in a number of frames. Among them, the face—as well as other regions like the hair—could be easily recognized. No region tracking or region correspondence was necessary, since the algorithm automatically located the regions in consequent frames. The segmentation results for the first and eight frames are shown in Fig. 10(a) and (b) for the "Claire" and in Fig. 10(c) and (d) for the "Akiyo" sequence. The extracted face region for the eight frames is shown in Figs. 4 and 11.

The 3-D ellipse fitting algorithm based on initial 2-D data was performed next and the adaptation results are demonstrated in Figs. 6 and 12. The resulting 3-D textured model for "Claire" is shown for different views in Fig. 7. As can be seen in Fig. 6, the ellipse fits accurately the facial area and it follows the rotation



(a)

(b)

(c)

Fig. 13. Motion estimation results for the "Claire" sequence. (a) Motion compensated frames 1–8. The original can be seen in Fig. 4. (b) Difference between original consequent frames. (c) Difference between original and motion compensated frames.

of the head in consequent frames. Having the 3-D model available, the rigid 3-D motion of each object is estimated next as described in Section VI. The resulting motion compensated frames 1–8 are shown in Fig. 13(a) while the corresponding original frames are in Fig. 4. In order to demonstrate the performance of the rigid 3-D motion estimation algorithm, the original frame differences between consequent frames [Fig. 13(b)] are compared with the frame differences between original frames and rigid motion compensated frames in Fig. 13(c).

The algorithm described above was also tested for the segmentation of the more complicated image sequence "foreman" of size 176×144. The original image is shown in Fig. 14(a). In this sequence, there is also camera motion, and the background does not remain static. For this reason, the algorithm presented in Section II-A was not used and the spatiotemporal segmentation with the connectivity constraint algorithm was rather applied to the full image. The segmentation result for the first frame is shown in Fig. 14(b). The extracted face region for the eight frames is shown in Fig. 15. The 3-D ellipse adaptation results are shown in Fig. 16.

Finally, using the VRML 2.0 nodes as described in Section VII-A, the VRML 2.0 compliant file is created containing

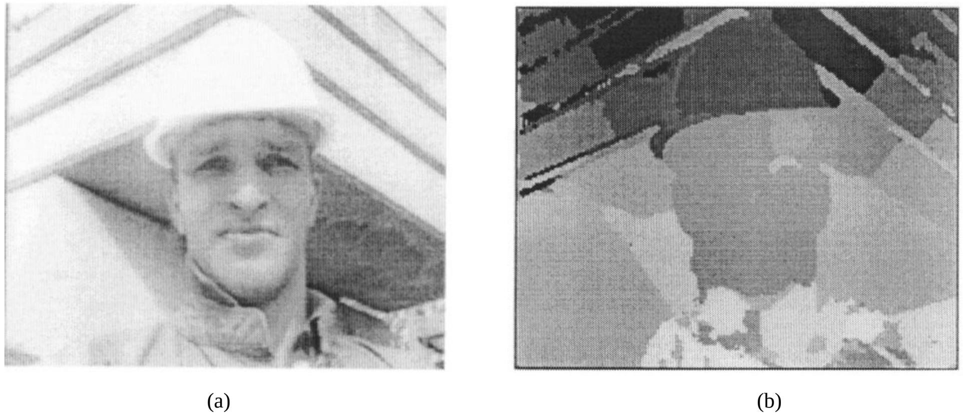


Fig. 14. (a) Original image "foreman." (b) Segmentation for the first frame of "foreman" sequence.

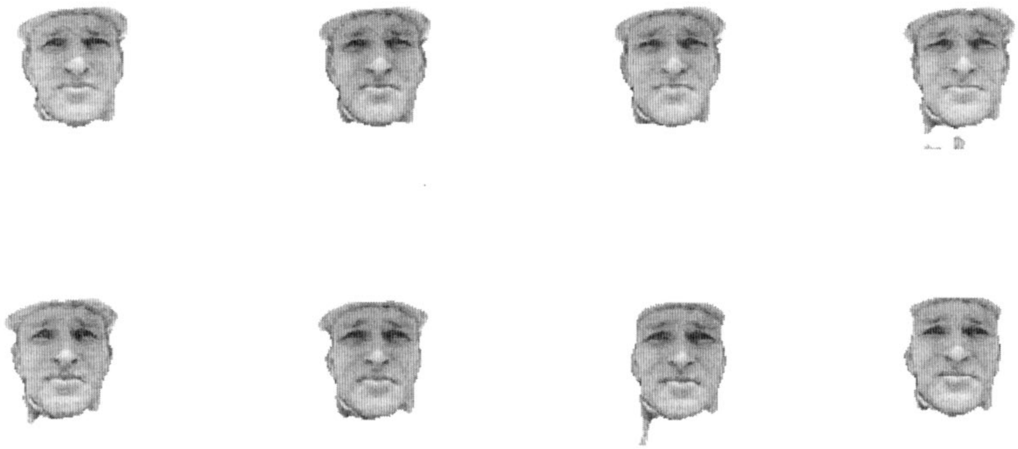


Fig. 15. Segmentation of the facial region for the first eight frames of the "foreman" sequence.

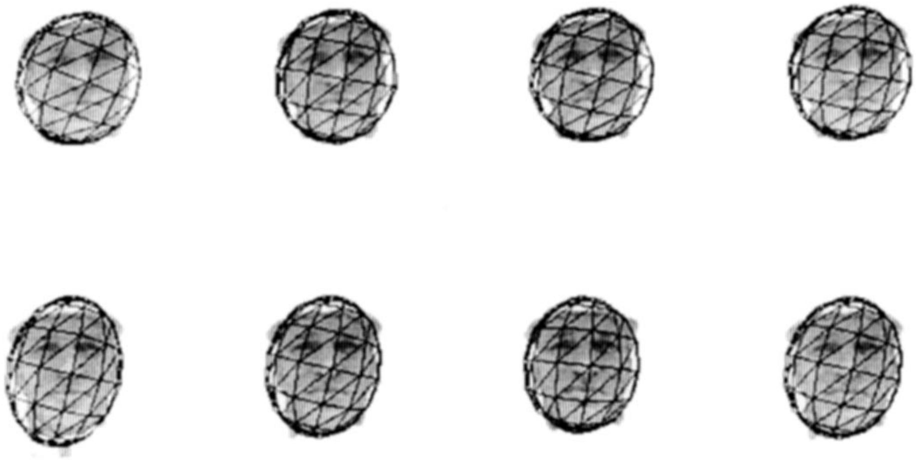


Fig. 16. Ellipse fitting to the facial region for the eight first frames of the "foreman" sequence.

the 3-D model, texture, and different rigid 3-D motion parameters for each object. Since the 3-D model is available, it can be easily adapted into virtual environments or combined with synthetic humans. In Fig. 17, "Claire" is shown next to a synthetically created desk and background. For this visualization, the ellipse fitting procedure was applied to one object created by merging the face and hair region and also the shoulders were included.

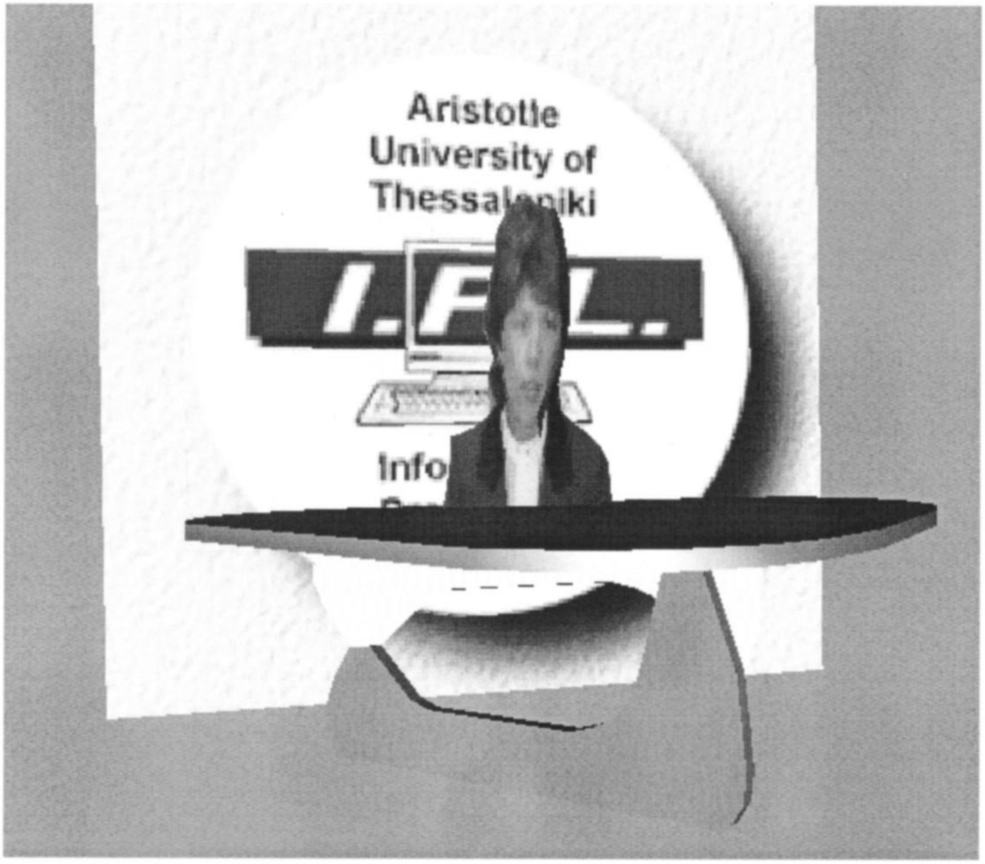


Fig. 17. "Claire" along a synthetically created desk and background.

IX. Conclusion

We have presented an algorithm for the segmentation, content-based coding and visualization of videoconference image sequences. For image sequence analysis, a general segmentation scheme was introduced, that can be used in any image sequence analysis scheme. The KMC algorithm is a novel segmentation algorithm combining information of several types including intensity, motion, and compactness. A number of frames is processed simultaneously and objects are detected in this number of frames. This *batch* approach implicitly solves the problem of correspondence of objects between consequent frames. The 3-D model is automatically created and adapted to the segmented objects using a robust method whereby 3-D ellipsoids are created from initial 2-D data. At the same time, complete correspondence between pixels and 3-D points is established. The rigid 3-D motion parameters are extracted separately for each object. For visualization purposes, the VRML 2.0 file format was used in order to provide compliance with a widespread format. Any World Wide Web browser may be used in order to download the file and view the moving scene. The visualization offers enhanced telepresence to the viewer, since a 3-D representation of the scene is created. Furthermore, the user can interact with the scene inside the VRML browser. Synthetic objects may be easily integrated with the virtualized scene in order to create synthetic natural hybrid video scenes.

Acknowledgment

The authors would like to thank COST 211 for its assistance.

References

[1] R. Koenen, "MPEG-4 multimedia for our time," *IEEE Spectrum*, vol. 36, no. 2, pp. 26–33, Feb. 1999.

[2] L. Chiariglione, "MPEG and multimedia communications," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 5–18, Feb. 1997.

[3] MPEG Requirements Group, "MPEG-7 context and objectives," in *Proc. MPEG Atlantic City Mtg.*, Oct. 1998, Doc. ISO/MPEC 2460.

[4] T. Sikora, "The MPEG-4 video standard verification model," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, no. 1, pp. 19–31, Feb. 1997.

[5] *IEEE Trans. Circuits Syst. Video Technol.—Special Issue on Image and Video Processing*, vol. 8, no. 5, Sept. 1998.

[6] *Signal Processing—Special Issue on Video Sequence Segmentation for Content-Based Processing and Manipulation*, vol. 66, no. 2, Apr. 1998.

[7] K. S. Fu and J. K. Mui, "A survey on image segmentation," *Pattern Recognit.*, vol. 13, pp. 3–16, 1981.

[8] R. M. Haralick and L. G. Sapiro, "Image segmentation techniques," *Comput. Vis., Graphics, Image Processing*, vol. 29, pp. 100–132, 1985.

[9] A. A. Alatan, L. Onural, M. Wollborn, R. Mech, E. Tuncel, and T. Sikora, "Image sequence analysis for emerging interactive multimedia services-The European COST 211 framework," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 8, pp. 19–31, Nov. 1998.

[10] D. Geiger and A. Yuille, "A common framework for image segmentation," *Int. J. Comput. Vis.*, vol. 6, pp. 227–243, 1991.

[11] T. Meier and K. N. Ngan, "Automatic segmentation of moving objects for video object plane generation," *IEEE Trans. Circuits Syst. Video Technol.—Special Issue on Image and Video Processing for Emerging Interactive Multimedia Services*, vol. 8, Sept. 1998.

[12] M. Kass, A. Witkin, and D. Terzopoulos, "Snakes: Active contour models," *Int. J. Comput. Vis.*, vol. 1, pp. 313–331, 1998.

[13] L. H. Staib and J. S. Duncan, "Boundary finding with parametric deformable models," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 14, pp. 161–175, 1992.

[14] S. T. Acton, A. C. Bovik, and M. M. Crawford, "Anisotropic diffusion pyramids for image segmentation," in *IEEE Int. Conf. Image Processing*, Austin, TX, Nov. 1994.

[15] P. Perona and J. Malik, "Scale space and edge-detection using anisotropic diffusion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, pp. 629–639, July 1990.

[16] M. Matheron, *Random Sets and Integral Geometry*. New York: Wiley, 1975.

[17] J. Serra, *Image Analysis and Mathematical Morphology*. New York: Academic, 1982.

[18] L. Vincent and P. Soille, "Watersheds in digital spaces: An efficient algorithm based on immersion simulations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, pp. 583–598, June 1991.

[19] D. Tzovaras, N. Grammalidis, and M. G. Strintzis, "Object-based coding of stereo image sequences using joint 3-D motion/disparity compensation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, pp. 801–811, Apr. 1997.

[20] M. Hötter, R. Mester, and F. Müller, "Detection and description of moving objects by stochastic modeling and analysis of complex scenes," *Signal Processing: Image Commun.*, vol. 8, pp. 281–193, 1996.

[21] N. Grammalidis, S. Malassiotis, D. Tzovaras, and M. G. Strintzis, "Stereo image sequence coding based on 3-D motion estimation and compensation," *Signal Processing: Image Commun.*, vol. 7, no. 2, pp. 129–145, Aug. 1995.

[22] I. Kompatsiaris and M. G. Strintzis, "Automatic 3D model construction for rigid 3D motion estimation of monocular videoconference image sequences," in *Int. Workshop Synthetic Natural Hybrid Coding and 3D Imaging*, Rhodes, Greece, Sept. 1997, pp. 44–47.

[23] I. Kompatsiaris, D. Tzovaras, and M. G. Strintzis, "3D model based segmentation of videoconference image sequences," *IEEE Trans. Circuits Syst. Video Technol.—Special Issue on Image and Video Processing for Emerging Interactive Multimedia Services*, vol. 8, pp. 547–561, Sept. 1998.

[24] M. A. Gennert and A. L. Yuille, "Determining the optimal weights in multiple objective function optimization," in *Proc. 2nd Int. Conf. Computer. Vision*, 1988.

[25] S. Ayer, "Sequential and competitive methods for estimation of multiple motions," Ph.D. dissertation, Ecole Polytechnique Federale de Lausanne, 1995.

[26] D. Wang, "Unsupervised video segmentation based on watersheds and temporal tracking," *IEEE Trans. Circuits Syst.s Video Technol.—Special Issue on Image and Video Processing for Emerging Interactive Multimedia Services*, vol. 8, Sept. 1998.

[27] H. G. Musmann, M. Hotter, and J. Ostermann, "Object-oriented analysis-synthesis coding of moving images," *Signal Processing: Image Commun.*, vol. 1, no. 2, pp. 117–138, Oct. 1989.

[28] M. Hotter, "Object-oriented analysis-synthesis coding based on moving two-dimensional object," *Signal Processing: Image Commun.*, vol. 2, no. 4, pp. 409–428, Dec. 1990.

[29] S. Malassiotis and M. G. Strintzis, "Model based joint motion and structure estimation from stereo images," *Comp. Vis. Image Understanding*, vol. 64, no. 2, Nov. 1996.

[30] J.-R. Ohm and E. Izquierdo, "An object-based system for stereoscopic viewpoint synthesis," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 7, Aug. 1997.

[31] I. Kompatsiaris and M. G. Strintzis, "Visualization of Videoconference Image Sequences Using VRML 2.0," in *Proc. IX Eur. Signal Processing Conference*, Rhodes, Greece, Sept. 1988.

[32] M. J. T. Reinders, "Model adaptation for image coding," Ph.D. dissertation, Delft University, Delft, The Netherlands, 1995.

[33] S. Z. Selim and M. A. Ismail, "K-means-type algorithms," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 6, pp. 81–87, Jan. 1984.

[34] S. Sakaida, Y. Shishikui, Y. Tanaka, and I. Yuyama, "Image segmentation by integration approach using initial dependence of k-means algorithm," in *Proc. Picture Coding Symp.'97*, Berlin, Germany, Sept. 1997, pp. 265–269.

[35] I. Kompatsiaris and M. G. Strintzis, "3D representation of videoconference image sequences using VRML 2.0," in *Proc. Eur. Conf. Multimedia Applications Services and Techniques (ECMAST'98)*, Berlin, Germany, May 1998, pp. 3–12.

[36] N. Herodotou, K. N. Plataniotis, and A. N. Venetsanopoulos, "Automatic location and tracking of the facial region in color video sequences," *Signal Processing: Image Commun.*, vol. 14, pp. 359–388, Mar. 1999.

[37] R. Castagno, T. Ebrahimi, and M. Kunt, "Video segmentation based on multiple features for interactive multimedia applications," *IEEE Trans. Circuits Syst. Video Technol.—Special Issue on Image and Video Processing for Emerging Interactive Multimedia Services*, vol. 8, pp. 562–571, Sept. 1998.

[38] P. Schroeter, "Unsupervised two-dimensional and three-dimensional image segmentation," Ph.D. dissertation, Swiss Federal Inst. Technol., Laussane, Switzerland, 1996.

[39] K. F. Lai and R. F. Chin, "Deformable contours: Modeling and extraction," *IEEE Trans. Pattern. Anal. Mach. Intell.*, vol. 17, pp. 1084–1090, Nov. 1995.

[40] S. Basu, I. Essa, and A. Pentland, "Motion regularization for model-based head tracking," in *Proc. Int. Conf. Pattern Recognition'96*, Vienna, Austria, Aug. 1996.

[41] N. Grammalidis and M. G. Strintzis, "Using 2-D and 3-D ellipsoid fitting for head and body segmentation and head tracking," in *Proc. IEEE Image and Multidimensional Digital Signal Processing (IMDSP) Workshop 1998*, Alpbach, Austria, July 1998.

[42] A. W. Fitzgibbon, M. Pilu, and R. B. Fisher, "Direct least squares fitting of ellipses," in *Proc. Int. Conf. Pattern Recognition*, Vienna, Austria, Aug. 1996.

[43] G. Adiv, "Determining three-dimensional motion and structure from optical flow generated by several moving objects," *IEEE Trans. Pattern. Anal. Machine Intell.*, vol. 7, pp. 384–401, July 1985.

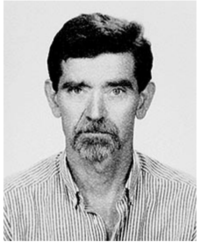
[44] D. Tzovaras, I. Kompatsiaris, and M. G. Strintzis, "3D object articulation and motion estimation in model based stereoscopic videoconference image sequence analysis and coding," *Signal Processing: Image Commun.*, vol. 14, no. 10, pp. 817–840, Aug. 1999.

[45] I. Kompatsiaris, D. Tzovaras, and M. G. Strintzis, "Flexible 3D motion estimation and tracking for multiview image sequence coding," *Signal Processing: Image Commun.*, vol. 14, no. 1-2, pp. 95–110, Nov. 1998.

[46] S. S. Sinha and B. G. Schunck, "A two-stage algorithm for discontinuity-preserving surface reconstruction," *IEEE Trans. Pattern. Anal. and Machine Intell.*, vol. 14, no. 1, Jan. 1992.

Ioannis Kompatsiaris (S'94) was born in Thessaloniki, Greece, in 1973. He received the five-year diploma in electrical engineering from the Electrical and Computer Engineering Department, Aristotle University of Thessaloniki (AUTH), Thessaloniki, Greece, in 1996, where he is currently working toward the Ph.D. degree.

He is a Graduate Research Assistant at the Electrical and Computer Engineering Department, AUTH, and at the Informatics and Telematics Institute, Thessaloniki, Greece. His research interests include image processing, computer vision, model-based monoscopic and multiview image sequence analysis and coding, medical image processing, and video coding standards. He is a member of the Technical Chamber of Greece.



Michael Gerassimos Strintzis (S'68–M'70–SM'80) received the diploma in electrical engineering from the National Technical University of Athens, Athens, Greece in 1967, and the M.A. and Ph.D. degrees in electrical engineering from Princeton University, Princeton, NJ in 1969 and 1970, respectively.

He then joined the Electrical Engineering Department, University of Pittsburgh, Pittsburgh, PA, where he served as Assistant (1970–1976) and Associate (1976–1980) Professor. Since 1980, he has been a Professor of Electrical and Computer Engineering, University of Thessaloniki, Thessaloniki, Greece, and since 1999, Director of the Informatics and Telematics Research Institute, Thessaloniki, Greece. His current research interests include 2-D and 3-D image coding, image processing, biomedical signal and image processing, and DVD and internet data authentication and copy protection.

Since 1999, Mr. Strintzis has served as an Associate Editor of the IEEE Transactions on Circuits and Systems for Video Technology. In 1984, Dr. Strintzis was awarded one of the Centennial Medals of the IEEE.