PERGAMON

Pattern Recognition 33 (2000) 1369–1382

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

LAFTER: a real-time face and lips tracker with facial expression recognition

Nuria Oliver[a,]*, Alex Pentland[a], François Bérard[b]

[a]Vision and Modeling, Media Laboratory, MIT, Cambridge, MA 02139, USA
[b]CLIPS-IMAG, BP 53, 38041 Grenoble Cedex 9, France

Received 22 October 1998; accepted 15 April 1999

Abstract

This paper describes an active-camera real-time system for tracking, shape description, and classification of the human face and mouth expressions using only a PC or equivalent computer. The system is based on use of 2-D *blob features*, which are spatially compact clusters of pixels that are similar in terms of low-level image properties. Patterns of behavior (e.g., facial expressions and head movements) can be classified in real-time using hidden Markov models (HMMs). The system has been tested on hundreds of users and has demonstrated extremely reliable and accurate performance. Typical facial expression classification accuracies are near 100%. © 2000 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Face and facial features detection and tracking; Facial expression recognition; Active vision; Hidden Markov Models

1. Introduction

This paper describes a real-time system for accurate tracking and shape description, and classification of the human face and mouth using 2-D *blob features* and hidden Markov models (HMMs). The system described here is real-time, at 20–30 frames per second, and runs on SGI Indy workstations or PentiumPro Personal Computers[1] without any special-purpose hardware.

In recent years, much research has been done on machine recognition of human facial expressions. Feature points [1], physical skin and muscle activation models [2–4], optical flow models [5], feature based models using manually selected features [6], local parametrized optical flow [7], deformable contours [8,9], combined with optical flow [10] as well as deformable templates [11–14] among several other techniques have been used for facial expression analysis.

This paper extends these previous efforts to real-time analysis of the human face using our blob tracking methodology. This extension required development of an incremental Expectation Maximization method, a new mixture-of-Gaussians blob model, and a continuous, real-time HMM classification method suitable for classification of shape data.

The notion of "blobs" as a representation for image features has a long history in computer vision [15–18], and has had many different mathematical definitions. In our usage it is a compact set of pixels that share a visual property that is not shared by the surrounding pixels. This property could be color, texture, brightness, motion, shading, a combination of these, or any other salient spatio-temporal property derived from the signal (the image sequence). In the work described in this paper blobs are a coarse, locally adaptive encoding of the images' spatial and color/texture/motion/etc. properties. A prime motivation for our interest in blob representations is our discovery that they can be reliably detected and tracked even in complex, dynamic scenes, and that

*Corresponding author. Tel.: +617-253-0608; fax: +617-253-8874.

E-mail addresses: nuria@media.mit.edu (N. Oliver), sandy@media.mit.edu (A. Pentland), francois.berard@imag.fr (F. Bérard).

[1] The active-camera face detection and tracking system has been ported to a PentiumPro using Microsoft VisualC++ under Windows NT. It also works in real-time (30 fps).

PII: S0031-3203(99)00113-2

they can be extracted in real-time without the need for special purpose hardware. These properties are particularly important in applications that require tracking people, and recently we have used 2-D blob tracking for real-time whole-body human interfaces [18] and real-time recognition of American Sign Language hand gestures [19].

Applications of this new system, called LAFTER [20] (Lips and Face TrackER) include video-conferencing, real-time computer graphics animation, and "virtual windows" for visualization. Of particular interest is our ability for accurate, real-time classification of the user's mouth shape without constraining head position; this ability makes possible (for the first time) real-time facial expression recognition in unconstrained office environments.

The paper is structured as follows: the general mathematical framework is presented in Section 2; LAFTER's architecture is described in Section 3; the face detection and tracking module appears in Section 4; Section 5 comprises the mouth detection and tracking; mouth expression recognition is in Section 7; results and applications are contained in Section 8 and finally the main conclusions and future work appear in Section 9.

2. Mathematical framework

The notion of grouping atomic parts of a scene together to form blob-like entities based on proximity and visual appearance is a natural one, and has been of interest to visual scientists since the Gestalt psychologists studied grouping criteria early in this century [21].

In modern computer vision processing we seek to group pixels of images together and to "segment" images based on visual coherence, but the "features" obtained from such efforts are usually taken to be the boundaries, or contours, of these regions rather than the regions themselves. In very complex scenes, such as those containing people or natural objects, contour features often prove unreliable and difficult to find and use.

The blob representation that we use was developed by Pentland and Kauth et al. [15,16] as a way of extracting an extremely compact, structurally meaningful description of multi-spectral satellite (MSS) imagery. In this method feature vectors at each pixel are formed by adding (x, y) spatial coordinates to the spectral (or textural) components of the imagery. These are then clustered so that image properties such as color and spatial similarity combine to form coherent connected regions, or "blobs", in which all the pixels have similar image properties. This blob description method is, in fact, a special case of recent minimum description length (MDL) techniques [22–25].

We have used essentially the same technique for real-time tracking of people in color video [18]. In that application the spatial coordinates are combined with color and brightness channels to form a four-element feature vector at each point $(x, y, \tilde{r}, \tilde{g}) = (x, y, (r/(r + g + b)), (g/(r + g + b)))$. These were then clustered into blobs to drive a "connected-blob" representation of the person.

By using the expectation–maximization [26] (EM) optimization method to obtain Gaussian mixture models for the spatio-chrominance feature vector, very complex shapes and color patterns can be adaptively estimated from the image stream. In our system we use an incremental version of EM, which allows us to adaptively and continuously update the spatio-chromatic blob descriptions. Thus not only can we adapt to very different skin colors, etc., but also to changes in illumination.

2.1. *Blobs: a probabilistic representation*

We can represent shapes in both 2-D and 3-D by their low-order statistics. Clusters of 2-D points have 2-D spatial means and covariance matrices, which we shall denote $\bar{q}$ and C_q. The blob spatial statistics are described in terms of their second-order properties. For computational convenience we will interpret this as a Gaussian model. The Gaussian interpretation is not terribly significant, because we also keep a pixel-by-pixel *support map* showing the actual occupancy.

Like other representations used in computer vision and signal analysis, including superquadrics, modal analysis, and eigen-representations, blobs represent the global aspects of the shape and can be augmented with higher-order statistics to attain more detail if the data supports it. The reduction of degrees of freedom from individual pixels to blob parameters is a form of regularization which allows the ill-conditioned problem to be solved in a principled and stable way.

For both 2-D and 3-D blobs, there is a useful physical interpretation of the blob parameters in the image space. The mean represents the geometric center of the blob area (2-D) or volume (3-D). The covariance, being symmetric semi-definite positive, can be diagonalized via an eigenvalue decomposition: $C = \Phi L \Phi^{\mathrm{T}}$, where Φ is orthonormal and L is diagonal.

The diagonal L matrix represents the size of the blob along uncorrelated orthogonal object-centered axes and Φ is a rotation matrix that brings this object-centered basis in alignment with the coordinate basis of C. This decomposition and physical interpretation is important for estimation, because the shape L can vary at a different rate than the rotation Φ. The parameters must be separated so they can be treated appropriately.

2.2. *Maximum likelihood estimation*

The blob features are modeled as a mixture of Gaussian distributions in the color (or texture, motion, etc.)

space. The algorithm that is generally employed for learning the parameters of such a mixture model is the *Expectation–Maximization* (EM) algorithm of Dempster et al. [26,27].

In our system the input data vector d is the normalized R, G, B content of the pixels in the image, $\mathbf{x} = (\tilde{r}, \tilde{g}) = (r/(r+g+b), g/(r+g+b))$. Our own work [18], or that of Schiele et al. or Hunke et al. [28,29] have shown that use of normalized or chromatic color information $(\tilde{r}, \tilde{g}) = (r/(r+g+b), g/(r+g+b))$ can be reliably used for finding flesh areas present in the scene despite wide variations in lighting. The color distribution of each of our blobs is modeled as a mixture of Gaussian probability distribution functions (PDFs) that are iteratively estimated using EM. We can perform a maximum likelihood decision criterium after the clustering is done because human skin forms a compact, low-dimensional (approximately 1-D) manifold in color space. Two different clustering techniques, both derived from EM are employed: an off-line training process and an on-line adaptive learning process.

In order to determine the mixture parameters of each of the blobs, the unsupervised EM clustering algorithm is computed off-line on hundreds of samples of the different classes to be modeled (in our case, face, lips and interior of the mouth), in a similar way as is done for skin color modeling in Ref. [30]. When a new frame is available the likelihood of each pixel is computed using the learned mixture model and compared to a likelihood threshold. Only those pixels whose likelihood is above the threshold are classified as belonging to the model.

2.3. Adaptive modeling via EM

Even though general models make the system relatively user-independent, they are not as good as an adaptive, user-specific model would be. We therefore use adaptive statistical modeling of the blob features to narrow the general model, so that its parameters are closer to the specific users' characteristics.

The first element of our adaptive modeling is to update the model priors as soon as the user's face has been detected. Given n independent observations $x_i = (\tilde{r}_i, \tilde{g}_i)$, $i = 1, \dots, n$ of the user's face, we model them as being samples of a Normal distribution in color space with mean the sample mean μ_{user} and covariance matrix, $\sum_{user}$. The skin color prior distribution is also assumed to be normal $p(x|\mu_{general}, \sum_{general}) = \mathrm{N}(\mu_{general}, \sum_{general})$ whose parameters have been computed from hundreds of samples of different users. By applying Bayesian integration of the prior and user's distributions we obtain a Normal posterior distribution $N(\mu_{post}, \sum_{post})$ whose sufficient statistics are given by:

$$\sum_{post} = \left[\sum_{general}^{-1} + \sum_{user}^{-1}\right]^{-1},$$

$$\mu_{post} = \sum_{post}\left[\sum_{general}^{-1} * \mu_{general} + \sum_{user}^{-1} * \mu_{user}\right]. \tag{1}$$

Eq. (1) corresponds to the computation of the posterior skin color probability distribution from the prior (general) and the user's (learned from the current image samples) models.

This update of skin model occurs only at the beginning of the sequence, assuming that the blob features are not going to drastically change during run time. To obtain a fully adaptive system, however, one must also be able to handle second-to-second changes in illumination and user characteristics.

We therefore use an *on-line* Expectation–Maximization algorithm [31,32] to adaptively model the image characteristics. We model both the background and the face as a mixture of Gaussian distributions with mixing proportions π_i and K components:

$$p(x/\Theta) = \sum_{i}^{K} \pi_i \frac{\mathrm{e}^{-1/2(x-\mu_i)^{\mathrm{T}}\sum_i^{(-1)}(x-\mu_i)}}{(2\pi)^{d/2}|\sum_i|^{1/2}}. \tag{2}$$

The unknown parameters of such a model are the sufficient statistics of each Normal distribution $(\mu_i, \sum_i)$, the mixing proportions π_i and the number of components of the mixture K.

The incremental EM algorithm is data driven, i.e., it estimates the distribution from the data itself. Two update algorithms are needed for this purpose: A criterium for adding new components to the current distribution as well as an algorithm for computing the sufficient statistics of each Normal Gaussian component.

The sufficient statistics are updated by computing an on-line version of the traditional EM update rules. If the first n data points have already been computed, the parameters when data point $(n+1)$ [2] is read are estimated as follows: First, the posterior class probability $p(i|x^{n+1})$ or *responsibility (credit)* h_i^{n+1} for a new data point x^{n+1} is computed:

$$h_i^{n+1} = \frac{\pi_i^n p(x^{n+1}/\theta_i^n)}{\sum_j \pi_j^n p(x^{n+1}/\theta_j^n)}. \tag{3}$$

This responsibility can be interpreted as the probability that a new data point x^{n+1} was generated by component i. Once this responsibility is known, the sufficient statistics of the mixture components are updated, weighted by the responsibilities:

$$\pi_i^{n+1} = \pi_i^n + \frac{h_i^{n+1} - \pi_i^n}{n}, \tag{4}$$

$$\mu_i^{n+1} = \mu_i^n + \frac{h_i^{n+1}}{n * w_i^n}(x^{n+1} - \mu_i^n), \tag{5}$$

[2] Superscript n will refer in the following to the estimated parameters when n data points have already been processed.

$$\sigma_i^{2(n+1)} = \sigma_i^{2(n)} + \frac{h_i^{n+1}}{n*w_i^n}((x^{n+1} - \mu_i^n)^2 - \sigma_i^{2(n)}), \tag{6}$$

where σ_i is the standard deviation of component i and w_i^{n+1} is the *average* responsibility of component i per point: $w_i^{n+1} = w_i^n + (h_i^n - w_i^n)/n$. The main idea behind this update rules is to distribute the effect of each new observation to all the terms in proportion to their respective likelihoods.

A new component is added to the current mixture model if the most recent observation is not sufficiently well explained by the model. If the last observed data point has a very low likelihood with respect of each of the components of the mixture, i.e. if it is an outlier for all the components, then a new component is added with mean the new data point and weight and covariance matrix specified by the user. The threshold in the likelihood can be fixed or stochastically chosen. In the latter case the algorithm would randomly choose whether to add a component or not given an outlier. There is a maximum number of components for a given mixture as well.

The foreground models are initialized with the off-line unsupervised learned *a priori* mixture distributions described above. In this way, the algorithm quickly converges to a mixture model that can be directly related to the *a priori* models' classes. The background models are not initialized with an *a priori* distribution but learned on-line from the image.

2.4. MAP segmentation

Given these models, a MAP foreground-background decision rule is applied to compute *support maps* for each of the classes, that is, pixel-by-pixel maps showing the class membership of each model. Given several statistical blob models that could potentially describe some particular image data, the membership decision is made by searching for the model with the maximum a posteriori (MAP) probability.

Once the class memberships have been determined, the statistics of each class are then updated via the EM algorithm, as described above. This approach can easily be seen to be a special case of the MDL segmentation algorithms developed by Darrell and Pentland [23,24] and later by Ayer and Sawhney [25].

2.5. Kalman filtering

Kalman filters have extensively been used in control theory as stochastic linear estimators. The Kalman filter was first introduced by Kalman [33] for discrete systems and by Kalman and Bucy [34] for continuous-time systems. The objective is to design an estimator that provides estimates of the non-observable estate of a system taking into account the known dynamics and the measured data. Note here that the Kalman filter provides the optimal *linear* estimate of the state, but, if all noises are Gaussian, it provides the *optimal* estimator.

In our system to ensure stability of the MAP segmentation process, the spatial parameters for each blob model are filtered using a zero-order Kalman filter. For each blob we maintain two independent, zero-order filters, one for the position of the blob centroid and another for the dimensions of the blob's bounding box. The MAP segmentation loop now becomes:

1. For each blob predict the filter state vector, $X^* = \hat{X}$ and covariance matrix, $C^* = \hat{C} + (\Delta t)^2 W$, where the matrix W measures the precision tolerance in the estimation of the vector X and depends on the kinematics of the underlying process.
2. For each blob new observations Y (e.g., new estimates of blob centroid and bounding box computed from the image data) are acquired and the Mahalanobis distance between these observations (Y, C) and the predicted state $(\hat{X}, \hat{C})$ is computed. If this distance is below threshold, the filters are updated by taking into account the new observations:

$$\hat{C} = [C^{*-1} + C^{-1}]^{-1}, \tag{7}$$

$$\hat{X} = \hat{C}[C^{*-1}X^* + C^{-1}Y]^{-1}. \tag{8}$$

   Otherwise a discontinuity is assumed and the filters are reinitialized: $\hat{X} = X^*$ and $\hat{C} = C^*$.

A generalized version of this technique is employed in Ref. [35] for fusing several concurrent observations. This Kalman filtering process is used in the tracking of all of the blob features. In our experience the stability of the MAP segmentation process is substantially improved by use of the Kalman filter, specially given that LAFTER's real-time performance yields small errors in the predicted filter state vectors. Moreover, smooth estimates of the relevant parameters are crucial for preventing jittering in the active camera, as described in Section 4.2.

2.6. Continuous real-time HMMs

Our approach to temporal interpretation of facial expressions uses Hidden Markov Models (HMMs) [36] to recognize different patterns of mouth shape. HMMs are one of the basic probabilistic tools used for time series modeling. A HMM is essentially a mixture model where all the information about the past of the time series is summarized in a single discrete variable, the *hidden state*. This hidden state is assumed to satisfy a *first-order Markov condition*: any information about the history of the process needed for future inferences must be reflected in the current state.

HMMs fall into our Bayesian framework with the addition of time in the feature vector. They offer dynamic time warping, an efficient learning algorithm and clear

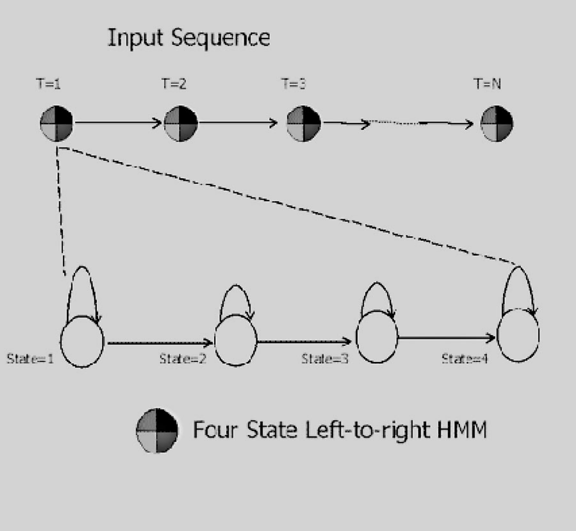

Fig. 1. Graphical representation of real-time left-to-right hidden Markov models.

Bayesian semantics. HMMs have been prominently and successfully used in speech recognition and, more recently, in handwriting recognition. However, their application to visual recognition purposes is more recent [37–40]. HMMs are usually depicted rolled-out in time, as Fig. 1 illustrates.

The posterior state sequence probability in a HMM is given by $P(S|O) = P_{s_1}p_{s_1}(0_1)\prod_{t=2}^{T}p_{s_t}(o_t)P_{s_t}|s_{t-1}$, where $S = \{a_1, \dots, a_N\}$ is the set of discrete states, $s_t \in S$ corresponds to the state at time t. $P_{i|j} \doteq P_{s_t=a_i|s_{t-1}=a_j}$ is the state-to-state transition probability (i.e. probability of being in state a_i at time t given that the system was in state a_j at time $t-1$). In the following we will write them as $P_{s_t|s_{t-1}}$. The prior probabilities for the initial state are expressed as $P_i \doteq P_{s_1=a_i} = P_{s_1}$. Finally, $p_i(o_t) \doteq p_{s_t=a_i}(o_t) = p_{s_t}(o_t)$ are the output probabilities for each state.[3] The Viterbi algorithm provides a formal technique for finding the most likely state sequence associated with a given observation sequence. To adjust the model parameters (transition probabilities $\mathscr{A}$, output probabilities parameters $\mathscr{B}$ and prior state probabilities π) such that they maximize the probability of the observation given the model an iterative procedure – such as the Baum–Welch algorithm — is needed.

We have developed a real-time HMM system that computes the maximum likelihood of the input sequence with respect to all the models during the testing or recognition phase. This HMM-based system runs in real time on an SGI Indy, with the low-level vision processing occurring on a separate Indy, and communications occurring via a socket interface.

3. System's architecture

LAFTER's main processing modules are illustrated in Fig. 2 and will be explained in further detail in the next sections.

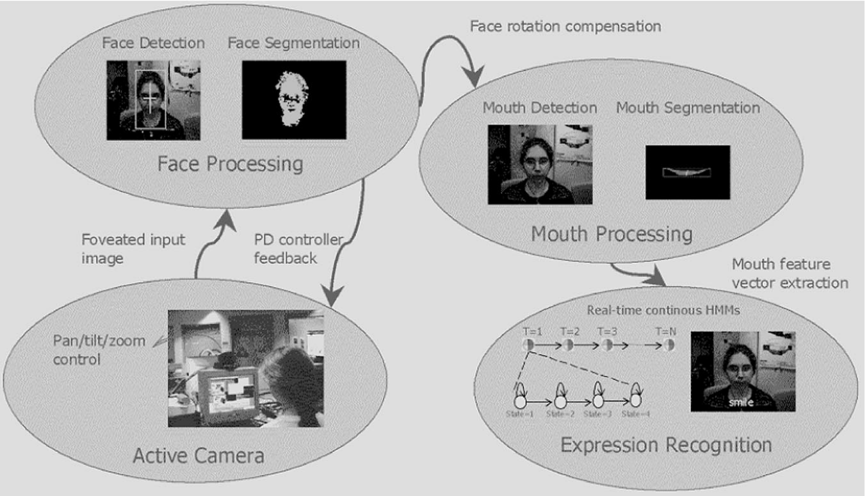

Fig. 2. LAFTER's architecture.

[3] The output probability is the probability of observing o_t given state a_i at time t.

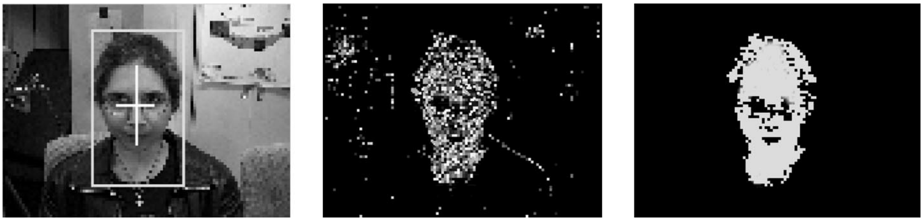

Fig. 3. Face detection, per-pixel probability image computation and face blob growing.

4. Automatic face detection and tracking

Our approach to the face finding problem uses coarse color and size/shape information. This approach has advantages over correlation or eigenspace methods, such as speed and rotation invariance under constant illumination conditions. As described in the mathematical framework (Section 2), our system uses an adaptive EM algorithm to accomplish the face detection process. Both the foreground and background classes are learned incrementally from the data. As a trade-off between the adaptation process and speed, new models are updated only when there is a significant drop in the posterior probability of the data given in the current model.

Two to three mixture components is the typical number required to accurately describe the face. Mouth models are more complex, often requiring up to five components. This is because the mouth model must include not only lips, but also the interior (tongue) of the mouth and the teeth.

4.1. Blob growing

After initial application of the MAP decision criterion to the image, often isolated and spurious pixels are misclassified. Thus local pixel information needs to be merged into connected regions that correspond to each of the blobs.

The transition from local to global information is achieved by applying a connected component algorithm which grows the blob. The algorithm we use is an speed-optimized version of a traditional connected component algorithm that considers for each pixel the values within a neighborhood of a certain radius (which can be varied at run-time) in order to determine whether this pixel belongs to the same connected region.

Finally, these blobs are then filtered to obtain the best candidate for being a face or a mouth. Color information alone is not robust enough for this purpose. The background, for instance, may contain skin colors that could be grown and erroneously considered as faces. Additional information is thus required. In the current system, geometric information, such as the size and shape of the object to be detected (faces) is combined with the color information to finally locate the face. In consequence, only those skin blobs whose size and shape (ratio of aspect of its bounding box) are closest to the canonical face size and shape are considered. The result is shown in Fig. 3.

4.2. Active camera control

Because our system already maintains a Kalman filter estimate of the centroid and bounding box of each blob, it is a relatively simple matter to use these estimates to control the active camera so that the face of the user always appears in the center of the image and with the desired size. Our system uses an abstraction of the camera control parameters, so that different camera/motor systems (currently the Canon VCC1 and Sony EVI-D30) can be successfully used in a transparent way. In order to increase tracking performance, the camera pan-tilt-zoom control is done by an independent light-weight process (thread) which is started by the main program.

The current estimation of the position and size of the user's face provides a reference signal to a PD controller which determines the tilt, pan and zoom of the camera so that the target (face) has the desired size and is at the desired location. The zoom control is relatively simple, because it just has to be increased or decreased until the face reaches the desired size. Pan and tilt speeds are controlled by $S_c = (C_{e*}E + C_{d^*}\mathrm{d}E/\mathrm{d}t)/F_z$, where C_e and C_d are constants, E is the error, i.e. the distance between the face current position and the center of the image, F_z is the zoom factor, and S_c is the final speed transmitted to the camera.

The zoom factor plays a fundamental role in the camera control because the speed with which the camera needs to be adjusted depends on the displacement that a fixed point in the image undergoes for a given rotation angle, which is directly related to the current zoom factor. The relation between this zoom factor and the current camera zoom position follows a non-linear law which needs to be approximated. In our case, a second order polynomial provides a good approximation. Fig. 4 illustrates the processing flow of the PD controller.

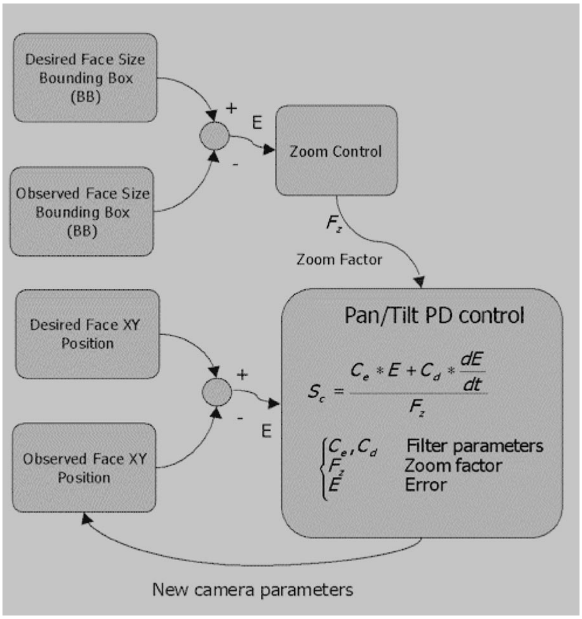
Desired Face Size Bounding Box (BB)
+
E
Zoom Control
-
Observed Face Size Bounding Box (BB)
F_z
Zoom Factor
Desired Face XY Position
Pan/Tilt PD control
$S_c = \frac{C_e * E + C_d * \frac{dE}{dt}}{F_z}$
+
-
E
Observed Face XY Position
C_e, C_d Filter parameters
F_z Zoom factor
E Error
New camera parameters

Fig. 4. PD controller.

5. Mouth extraction and tracking

Once the face location and shape parameters are known (center of the face, width, height and image rotation angle), we can use anthropometric statistics to define a bounding box within which the mouth must be located.

The mouth is modeled using the same principles as the face, i.e. through a second-order mixture model that describes both its chromatic color and spatial distribution. However to obtain good performance we must also produce a more finely detailed model of the face region surrounding the mouth. The face model that is adequate for detection and tracking might not be adequate for accurate mouth shape extraction.

Our system, therefore, acquires image patches from around the located mouth[4] and builds a Gaussian mixture model. In the current implementation, skin samples of three different facial regions around the mouth are extracted during the initialization phase and their statistics are computed, as is depicted in Fig. 5. The second image in the same figure is an example of how the system performs in the case of facial hair. The robustness of the system is increased by computing at each time step the linearly predicted position of the center of the mouth. A confidence level on the prediction is also computed, depending on the prediction error. When the prediction is not available or its confidence level drops below a threshold, the mouth's position is reinitialized.

5.1. Mouth shape

The mouth shape is characterized by its area, its spatial eigenvalues (e.g., width and height) and its bounding box. Fig. 6 depicts the extracted mouth feature vector. The use of this feature vector to classify facial expressions has been suggested by psychological experiments [41,42], which examined the most important discriminative features for expression classification.

Rotation invariance is achieved by computing the face's image-plane rotation angle and rotating the region of interest with the negative of this angle. Therefore, even though the user might turn the head the mouth always appears nearly horizontal, as Fig. 5 illustrates.

6. Speed, accuracy, and robustness

Running LAFTER on a single SGI Indy with a 200Mhz R4400 processor, the average frame rate for tracking is typically 25 Hz. When mouth detection and parameter extraction are added to the face tracking, the average frame rate is 14 Hz.

To measure LAFTER's 3-D accuracy during head motion, the RMS error was measured by having users make large cyclic motions along the $\bar{X}$, $\bar{Y}$, and Z-axis, respectively, with the true 3-D position of the face being determined by manual triangulation. In this experiment the camera actively tracked the face position, with the image-processing/camera-control loop running at a nearly constant 18 hz. The image size was 1/6 full

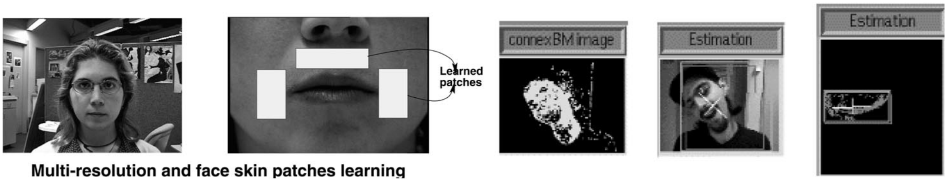

Fig. 5. Multi-resolution mouth extraction, skin model learning. Head and mouth tracking with rotations and facial hair.

[4] The mouth extraction and processing is performed on a Region of Interest (ROI) extracted from a full resolution image (i.e. 640 × 480 pixels) whereas the face detection and processing is done on an image of 1/6 full resolution, i.e. 106 × 80 pixels.

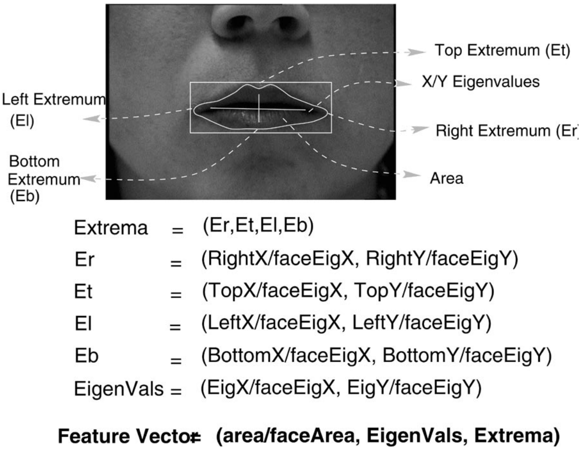
Top Extremum (Et)
X/Y Eigenvalues
Left Extremum (El)
Right Extremum (Er)
Bottom Extremum (Eb)
Area
Extrema = (Er,Et,El,Eb)
Er = (RightX/faceEigX, RightY/faceEigY)
Et = (TopX/faceEigX, TopY/faceEigY)
El = (LeftX/faceEigX, LeftY/faceEigY)
Eb = (BottomX/faceEigX, BottomY/faceEigY)
EigenVals = (EigX/faceEigX, EigY/faceEigY)
Feature Vector= (area/faceArea, EigenVals, Extrema)

Fig. 6. Mouth feature vector extraction.

resolution, i.e. 106 × 80 pixels, and the camera control law varied pan, tilt, and zoom to place the face in the center of the image at a fixed pixel resolution. Fig. 7 illustrates the active-camera tracking system in action. The RMS error between the true 3-D location and the system's output was computed in pixels and is shown in Table 1. Also shown is the variation in apparent head size, e.g., the system's error at stabilizing the face image size. As can be seen, the system gave quite accurate estimates of 3-D position. Perhaps most important, however, is the robustness of the system. LAFTER has been tested on hundreds of users at many different events, each with its own lighting and environmental conditions. Examples are the *Digital Bayou*, part of SIGGRAPH 96', the *Second International Face & Gesture Workshop* (*October 96*) or several open houses at the Media Laboratory during the last two years. In all cases the system failed in

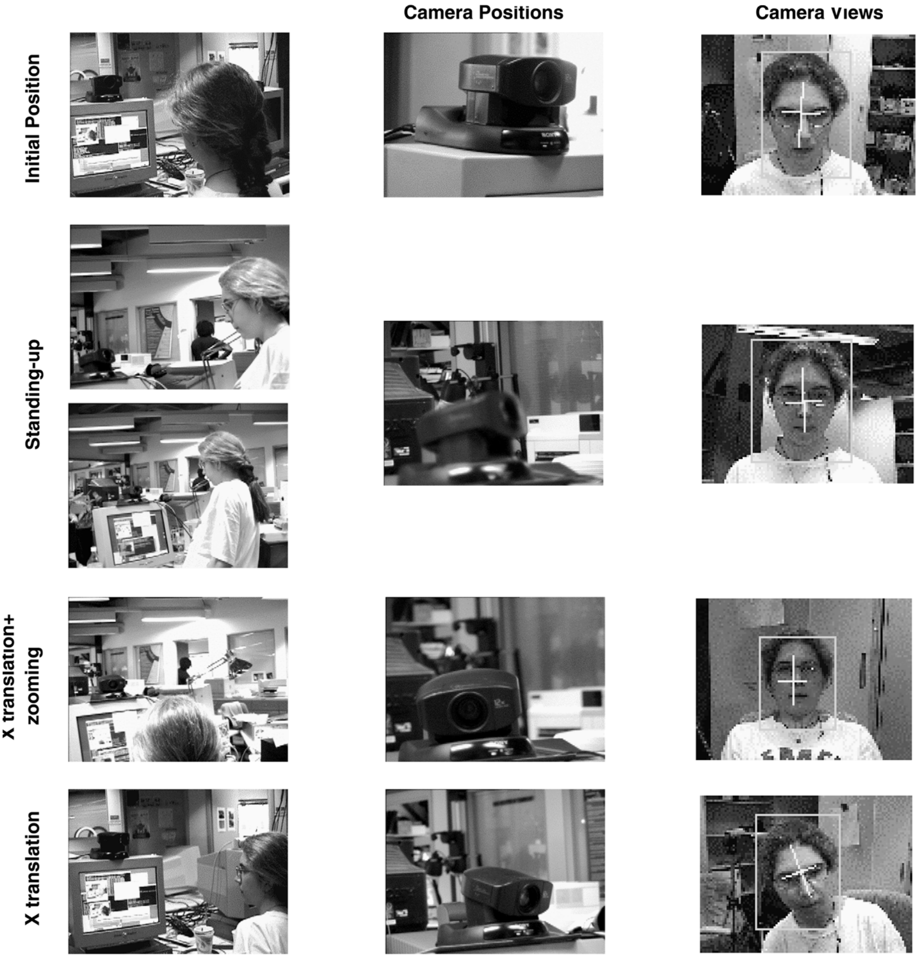

Fig. 7. Active camera tracking.

Table 1
Translation and zooming active tracking accuracies

	Translation Range	X RMS Error (pixels)	Y RMS Error (pixels)
Static Face	0.0 cm	0.5247 (0.495%)	0.5247 (0.6559%)
X translation	± 76 cm	0.6127 (0.578%)	0.8397 (1.0496%)
Y translation	± 28 cm	0.8034 (1.0042%)	1.4287 (1.7859%)
Z translation	± 78 cm	0.6807 (0.6422%)	1.1623 (1.4529%)
	Width Std (pixels)	Height Std (pixels)	Size change (pixels)
Zooming	2.2206 (2.09%)	2.6920 (3.36%)	Max. size: 86 × 88 Min. size: 14 × 20

approximately 3–5% of the cases, when the users had dense beard, extreme skin color or clothing very similar to the skin color models.

7. Mouth-shape recognition

Using the mouth shape feature vector described above, we trained five different HMMs for each of the following mouth configurations (illustrated in Fig. 8): neutral or default mouth position, extended/smile mouth, sad mouth, open mouth and extended + open mouth (such as in laughing).

The neutral mouth acted to separate the various expressions, much as a silence model acts in speech recognition. The final HMMs we derived for the non-neutral mouth configurations consisted of 4-state forward HMMs. The neutral mouth was modeled by a 3-state forward HMM.

Recognition results for a eight different users making over 2000 expressions are summarized in Table 2. The data were divided into different sets for training and testing purposes. The first line of the recognition results shown in Table 2 corresponds to training and testing with all eight users. The total number of examples is denoted by N, having a total $N = 2058$ instances of the mouth expressions ($N = 750$ for training and $N = 1308$ for testing). The second line of the same table corresponds to person-specific training and testing. As can be seen, accurate classification was achieved in each case.

In comparison with other facial expression recognition systems, the approach proposed by Matsuno et al. [2] performs extremely well on training data (98.4% accuracy) but more poorly on testing data, with 80% accuracy. They build models of facial expressions from deformation patterns on a potential net computed on training images and subsequent projection in the so called *Emotion Space*. Expressions of new subjects are recognized by projecting the image net onto the Emotion Space. Black et al. [7] report an overall average recognition of 92% for six different facial expressions (happiness, surprise, anger, disgust, fear and sadness) in 40 different subjects. Their system combines deformation and motion parameters to derive mid- and high-level descriptions of facial actions. The descriptions depend on a number of thresholds and a set of rules that need to be tuned for each expression and/or subject. The system described in Ref. [43] has a recognition rate of about 74% when using 118 testing images of the seven psychologically recognized categories across several subjects. They use flexible models for representing appearance variations of faces. Essa et al. [44] report 98% accuracy in recognizing five different facial expressions using both peak-muscle activations and spatio-temporal motion energy templates from a database of 52 sequences. An accuracy of 98.7% is reported by Yael Moses et al. [9] on real-time facial expression recognition. Their system detects and tracks the user's mouth, by representing it by a valley contour based between the lips. A simple classification algorithm is then

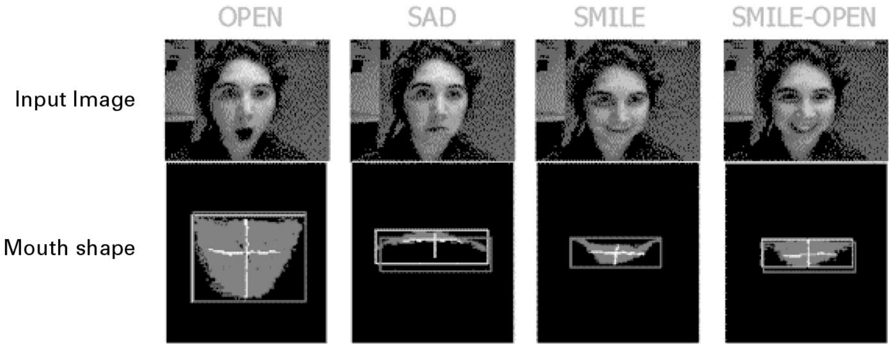

Fig. 8. Open, sad, smile and smile-open recognized expressions.

Table 2
Recognition results: training and testing data

	Test on:	
Train on	Training	Testing
All users	97.73	95.95
Single user	100.00	100.00

used to discriminate between five different mouth shapes. They consider only confusions but not false negatives (confusions of any expression to neutral) on two independent samples of about 1000 frames each and of a predetermined sequence of five different expressions plus the neutral face. Padgett et al. [45] report 86 accuracy on emotion recognition on novel individuals using neural networks for classification. The recognized emotions are happy, sad, fear, anger, surprise, disgust or neutral across 12 individuals. Finally the method adopted by Lien et al. [46] is the most similar to ours in the sense of the recognition approach, because they also use HMMs. The expression information is extracted by use of facial feature point tracking (for the lower face — mouth—) or by pixel-wise flow tracking (for the upper face — forehead and eyebrows—) followed by PCA to compress the data. Their system has an average recognition rate for the lower face of 93 and for the upper face of 91% using FACS.

8. Applications

8.1. Automatic camera man

The static nature of current video communication systems induces extra articulatory tasks that interfere with real world activity. For example, users must keep their head (or an object of interest) within the field of the camera (or of the microphone) in order to be perceived by distant parties. As a result, the user ends up being more attentive to the way how to using the interface than to the conversation itself. The communication is therefore degraded instead of enriched.

In this sense, LAFTER, with its active camera face tracking acts as an 'automatic camera man' that is continuously looking at the user while he/she moves around or gestures in a video-conference session. In informal teleconferencing testing, users have confirmed that this capability significantly improves the usability of the teleconferencing system.

8.2. Experiences with a virtual window system

Some of the limitations of traditional media spaces — with respect to the visual information — are [47]: restricted field of view on remote sites by the video, limited video resolution, spatial discontinuity, medium anisotropy and very restricted movement with respect to remote spaces. Each of these negatively affects the communication in a media space, with movement one of the most influential, as Gibson emphasized in Ref. [48]. Motion allows us to increase our field of view, can compensate for low resolution, provides information about the three-dimensional layout and allow people to compensate for the discontinuities and anisotropies of current media spaces, among other factors. Therefore, not only allowing movement in local media spaces is a key element for desktop mediated communication and video-conference systems — as we have previously emphasized —, but also the ability of navigating and exploring the remote site.

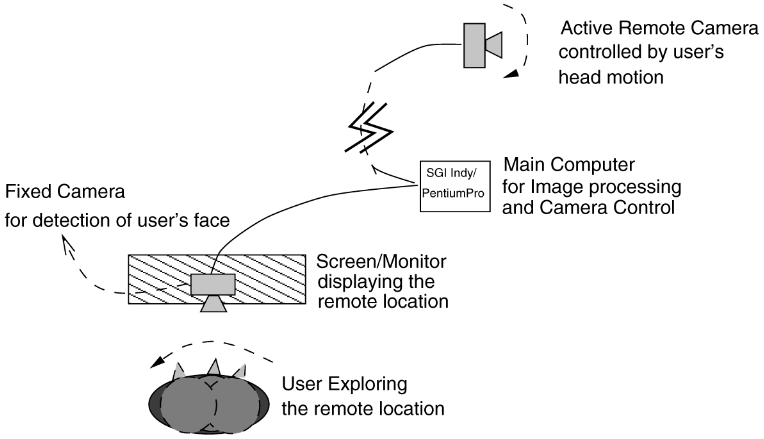
Active Remote Camera controlled by user's head motion
SGI Indy/ PentiumPro
Main Computer for Image processing and Camera Control
Fixed Camera for detection of user's face
Screen/Monitor displaying the remote location
User Exploring the remote location

Fig. 9. The virtual window: Local head positions are detected by the active tracking camera and used to control a moving camera in the remote site. The effect is that the image on the local monitor changes as if it were a window. The second image illustrates the virtual window system in use.

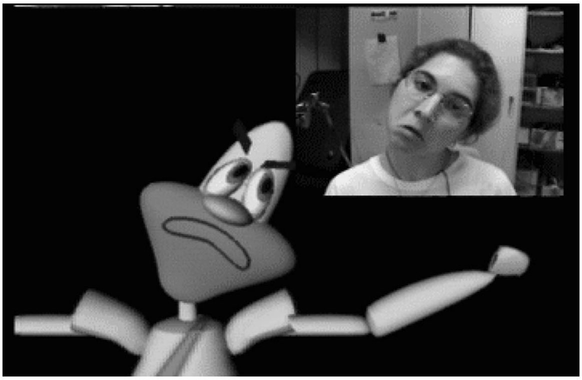

Fig. 10. Real-time computer graphics animation.

The Virtual Window proposed by Gaver [49] illustrates an alternative approach: as the user moves in front of his local camera, the distant motorized camera is moved accordingly: exploring a remote site by using head movements opens a broad spectrum of possibilities for systems design that allow an enriched access to remote partners. Fig. 9 depicts an example of a virtual window system.

One of the main problems that Gaver recognized in his virtual window system was that its vision controller was too sensitive to lighting conditions and to moving objects. Consequently, the tracking was unstable; users were frustrated and missed the real purpose of the system when experiencing it.

We found that by incorporating our face tracker into a Virtual Window system, users could successfully obtain the effect of a window onto another space. To the best of our knowledge this is the first real-time robust implementation of the virtual window. In informal tests, users reported that the LAFTER-based virtual window system gives a good sense of the distant space.

8.3. Real-time computer graphics animation

Because LAFTER continuously tracks face location, image-plane face rotation angle, and mouth shape, it is a simple matter to use this information to obtain real-time animation of a computer graphics character. This character can, in its simplest version, constantly mimic what the user does (as if it were a virtual mirror) or, in a more complex system, understand (recognize) what the user is doing and react to it. A “virtual mirror” version of this system — using the character named Waldorf shown in Fig. 10 — was exhibited in the Digital Bayou section of SIGGRAPH’96 in New Orleans.

8.4. Preferential coding

Finally, LAFTER can be used as the front-end to a *preferential image coding system*. It is well known that people are most sensitive to coding errors in facial features. Thus it makes sense to use a more accurate (and more expensive) coding algorithm for the facial features,

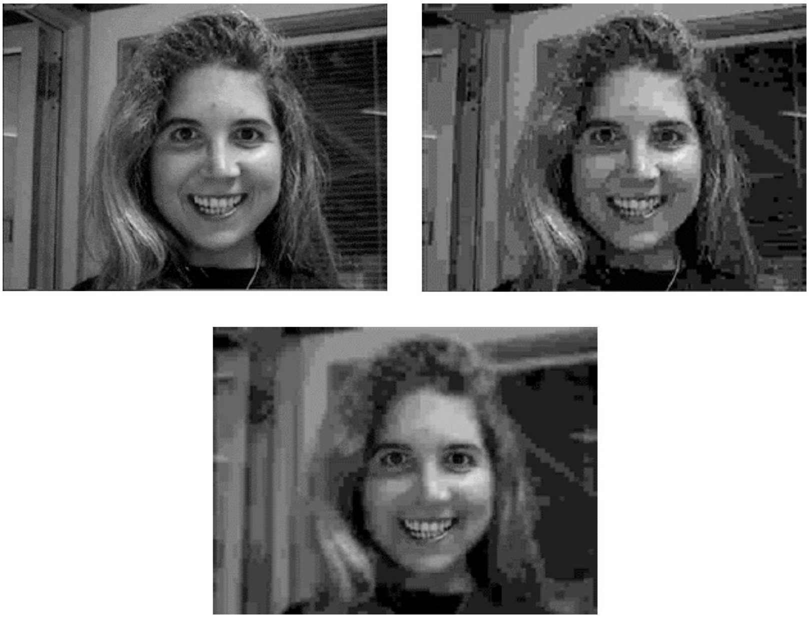

Fig. 11. Preferential coding: the first image is the JPEG flat encoded image (File size of 14.1 Kb); the second is a very low resolution JPEG encoded image using flat coding (File size of 7.1 Kb); the third one is a preferential coding encoded image with high resolution JPEG for the eyes and mouth but very low resolution JPEG coding for the face and background (File size of 7.1 Kb).

and a less accurate (and cheaper) algorithm for the remaining image data [50–52]. Because the location of these features is detected by our system, we can make use of this coding scheme. The improvement obtained by such system is illustrated in Fig. 11.

9. Conclusion and future work

In this paper we have described a real-time system for finding and tracking a human face and mouth, and recognizing mouth expressions using HMMs. The system runs on a single SGI Indy computer or PentiumPro Personal Computer, and produces estimates of head position that are surprisingly accurate.

The system has been successfully tested on hundreds of naive users in several physical locations and used as the base for several different applications, including an automatic camera man, a virtual window video communications system, and a real-time computer graphics animation system.

10. Summary

This paper describes an active-camera real-time system for tracking, shape description, and classification of the human face and mouth using only a PC or equivalent computer. The system is based on use of 2-D *blob features*, which are spatially-compact clusters of pixels that are similar in terms of low-level image properties. Patterns of behavior (e.g., facial expressions and head movements) can be classified in real-time using Hidden Markov Models (HMMs). The system has been tested on hundreds of users and has demonstrated extremely reliable and accurate performance. Typical facial expression classification accuracies are near 100%. LAFTER has been used as the base for several practical applications, including an automatic camera-man, a virtual window video communications system, and a real-time computer graphics animation system.

References

[1] A. Azarbayejani, A. Pentland, Camera self-calibration from one point correspondence, Tech. Rep. 341, MIT Media Lab Vision and Modeling Group, 1995. Submitted IEEE Symposium on Computer Vision.

[2] K. Matsuno, P. Nesi, Automatic recognition of human facial expressions, CVPR'95, IEEE, New York 1 (1995) 352–359.

[3] K. Waters, A muscle model for animating three-dimensional facial expression, in: M.C. Stone (Ed.), SIGGRAPH '87 Conference Proceedings, Anaheim, CA, July 27–31, 1987, Computer Graphics, Vol. 21, Number 4, July 1987, pp. 17–24.

[4] M. Rydfalk, CANDIDE: a parametrized face, Ph.D. Thesis, Linköpnik University, EE Dept., October 1987.

[5] I. Essa, A. Pentland, Facial expression recognition using a dynamic model and motion energy, ICCV'95 (1995) 360–367.

[6] I. Pilowsky, M. Thornton, M. Stokes, Aspects of face processing, Towards the quantification of facial expressions with the use of a mathematics model of a face (1986) 340–348.

[7] M. Black, Y. Yacoob, Tracking and recognizing rigid and non-rigid facial motions using local parametric models of image motion, ICCV'95 (1995) 374–381.

[8] R. Magnolfi, P. Nosi, Analysis and synthesis of facial motions, International Workshop on Automatic Face and Gesture Recognition, IEEE, Zurich 1 (1995) 308–313.

[9] Y. Moses, D. Reynard, A. Blake, Determining facial expressions in real time, ICCV'95 (1995) 296–301.

[10] Y. Yacoob, L. Davis, Recognizing human facial expressions from long image sequences using optical-flow, Pattern Anal. Mach. Intell. 18 (1996) 636–642.

[11] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, Int. J. Comput. Vision 1 (1988) 321–331.

[12] A. Yuille, P. Hallinan, D. Cohen, Feature extraction from faces using deformable templates, Int. J. Comput. Vision 8 (1992) 99–111.

[13] H. Hennecke, K. Venkatesh, D. Stork, Using deformable templates to infer visual speech dynamics, Tech. Rep., California Research Center, June 1994.

[14] C. Bregler, S.M. Omohundro, Surface Learning with Applications to Lipreading, Adv. Neural Inform. Process. Systems 6 (1994) 43–50.

[15] A. Pentland, Classification by clustering, IEEE Symposium on Machine Processing and Remotely Sensed Data, Purdue, IN, 1976.

[16] R. Kauth, A. Pentland, G. Thomas, Blob: an unsupervised clustering approach to spatial preprocessing of mss imagery, 11th International Symposium on Remote Sensing of the Environment, Ann Harbor MI, 1977.

[17] A. Bobick, R. Bolles, The representation space paradigm of concurrent evolving object descriptions, Pattern Anal. Mach. Intell. 14 (2) (1992) 146–156.

[18] C. Wren, A. Azarbayejani, T. Darrell, A. Pentland, Pfinder: real-time tracking of the human body, Photonics East, SPIE, Vol. 2615, Bellingham, WA, 1995.

[19] T. Starner, A. Pentland, Real-time asl recognition from video using hmm's, Technical Report 375, MIT, Media Laboratory, MIT, Media Laboratory, Cambridge, MA 02139.

[20] N. Oliver, F. Berard, A. Pentland, Lafter: lips and face tracking, IEEE International Conference on Computer Vision and Pattern Recognition (CVPR97), S. Juan, Puerto Rico, June 1997.

[21] W. Ellis, A Source Book of Gestalt Psychology. In Harcourt Brace and Co., New York, 1939.

[22] J. Rissanen, Encyclopedia of Statistical Sciences, Minimum-Description-Length Principle, Vol. 5, Wiley, New York, 1987, pp. 523–527.

[23] T. Darrell, S. Sclaroff, A. Pentland, Segmentation by minimal description, ICCV'90 (1990) 173–177.

[24] T. Darrell, A. Pentland, Cooperative robust estimation using layers of support, Pattern Anal. Mach. Intell. 17 (5) (1995) 474–487.

[25] S. Ayer, H. Sawhney, Layered representation of motion video using robust maximum-likelihood estimation of mixture models and mdl encoding, ICCV95.

[26] A. Dempster, N. Laird, D. Rubin, Maximum likelihood from incomplete data via de *em* algorithm, J. Roy. Statist. Soc. 39-B (1977) 1–38.

[27] R. Redner, H. Walker, Mixture densities, maximum likelihood and the *em* algorithm, SIAM Rev. 26 (1984) 195–239.

[28] B. Schiele, A. Waibel, Gaze tracking based on face color, International Workshop on Automatic Face and Gesture Recognition (1995) 344–349.

[29] H. Hunke, Locating and tracking of human faces with neural networks, Technical Report, CMU, CMU, Pittsburgh PA, August 1994.

[30] T. Jebara, A. Pentland, Parametrized structure from motion for 3d adaptive feedback tracking of faces, CVPR 97 (1997) 144–150.

[31] C. Priebe, Adaptive mixtures, J. Amer. Statist. Assoc. 89 (427) (1994) 796–806.

[32] D.M. Titternington, Recursive parameter estimation using incomplete data, J. Roy. Statist. Soc. B 46 (1984) 257–267.

[33] R. Kalman, A new approach to linear filtering and prediction problems, ASME J. Eng. 82 (1960) 35–45.

[34] R. Kalman, R. Bucy, New results in linear filtering and prediction theory, Trans. ASME Ser. D. J. Basic Engng. 83 (1961) 95–107.

[35] J. Crowley, F. Berard, Multi-modal tracking of faces for video communications, CVPR97 (1997) 640–645.

[36] L.R. Rabiner, B.H. Juang, An introduction to hidden Markov models, IEEE ASSP Mag. Jan. (1986) 4–16.

[37] J. Yamato, J. Ohya, K. Ishii, Recognizing human action in time-sequential images using hidden markov models, Trans. Inst. Electron. Inform. Commun. Eng. D-II, J76D-II, (12) (1993) 2556–2563.

[38] A. Wilson, A. Bobick, Learning visual behavior for gesture analysis, IEEE International Symposium on Computer Vision, 1995.

[39] A. Wilson, A. Bobick, Recognition and interpretation of parametric gesture, International Conference on Computer Vision, 1998.

[40] T. Starner, A. Pentland, Visual recognition of american sign language using hidden markov models, International Workshop on Automatic Face and Gesture Recognition, Zurich, Switzerland, 1995.

[41] H. Yamada, Dimensions of visual information for categorizing facial expressions, Japanese Psychol. Res. 35 (4) (1993) 172–181.

[42] S. Morishima, Emotion model, International Workshop on Automatic Face and Gesture Recognition, Zurich (1995) 284–289.

[43] A. Lanitis, C. Taylor, T. Cootes, A unified approach for coding and interpreting face images, ICCV'95 (1995) 368–373.

[44] I.A. Essa, Analysis, Interpretation, and Synthesis of Facial Expressions. PhD Thesis, MIT Department of Media Arts and Sciences, 1995.

[45] C. Padgett, G. Cottrell, Representing face images for emotion classification, Neural Information Processing Systems, NIPS'96, Denver, Colorado, USA, 1996.

[46] J. Lien, T. Kanade, J. Cohn, A. Zlochower, C. Li, Automatically recognizing facial expressions in spatio-temporal domain using hidden markov models, in: Proceedings of the Workshop on Perceptual User Interfaces, PUI97, Banff, Canada, 1997.

[47] W. Gaver, The affordances of media spaces for collaboration, CSCW, 1992.

[48] J. Gibson, The Ecological Approach to Visual Perception, Houghton Mifflin, New York, 1979.

[49] W. Gaver, G. Smets, K. Overbeeke, A virtual window on media space, CHI, 1995.

[50] A. Eleftheriadis, A. Jacquin, Model-assisted coding of video teleconferencing sequences at low bit rates, ISCAS, May–June 1994.

[51] K. Ohzeki, T. Saito, M. Kaneko, H. Harashima, Interactive model-based coding of facial image sequence with a new motion detection algorithm, IEICE E79B (1996) 1474–1483.

[52] K. Aizawa, T. Huang, Model-based image-coding: Advanced video coding techniques for very-low bit-rate applications, Proceedings of IEEE 83 (1995) 259–271.

About the Author—NURIA M. OLIVER is a Research Assistant in the Vision and Modeling Group at the Media Laboratory of Massachussetts Institute of Technology, pursuing a Ph.D. in Media Arts and Sciences. She works with Professor Alex Pentland. She received with honors her B.Sc. and M.Sc. degrees in Electrical Engineering and Computer Science from ETSIT at the Universidad Politecnica of Madrid (UPM), Spain, 1994. Before starting her Ph.D. at MIT she worked as a research engineer at Telefonica I + D. Her research interests are computer vision, machine learning and artificial intelligence. Currently she is working on the three previous disciplines in order to build computational models of human behavior.

About the Author—ALEX (SANDY) P. PENTLAND is the Academic Head of the M.I.T. Media Laboratory. He is also the Toshiba Professor of Media Arts and Sciences, an endowed chair last held by Marvin Minsky. His recent research focus includes understanding human behavior in video, including face, expression, gesture, and intention recognition, as described in the April 1996 issue of Scientific American. He is also one of the pioneers of wearable computing, a founder of the IEEE Wearable Computer technical area, and General Chair of the upcoming IEEE International Symposium on Wearable Computing. He has won awards from the AAAI, IEEE, and Ars Electronica. He is a founder of the IEEE Wearable Computer technical area, and General Chair of the upcoming IEEE International Symposium on Wearable Computing.

About the Author—FRANÇOIS BÉRARD is a Ph.D. student in Computer Science at CLIPS-IMAG laboratory at the University of Grenoble (France). His research interests concern the development of real-time Computer Vision systems and their use in the field of Human-Computer Interaction. His research advisors are Professor Joëlle Coutaz and Professor James L. Crowley. He spent two summers working with Prof. Alex Pentland at the MIT Media Laboratory's Vision and Modeling Group and with Michael Black at Xerox PARC's Image Understanding Area.