M.I.T. Media Laboratory Perceptual Computing Group Technical Report No. 364.
(Submitted, IEEE Trans. PAMI; Portions previously appeared, "Space-Time Gestures", IEEE Proc. CVPR '93, and in "Correlation and Interpolation Networks for Real-time Expression Analysis/Synthesis", Proc. NIPS '94)

Task-specific Gesture Analysis in Real-Time using Interpolated Views

Trevor J. Darrell, Irfan A. Essa, Alex P. Pentland
MIT Media Lab

Revised, December, 1995

Abstract

Hand and face gestures are modeled using an appearance-based approach in which patterns are represented as a vector of similarity scores to a *set* of view models defined in space and time. These view models are learned from examples using unsupervised clustering techniques. A supervised learning paradigm is used to interpolate view scores into a task-dependent coordinate system appropriate for recognition and control tasks. We apply this analysis to the problem of context-specific gesture interpolation and recognition, and demonstrate real-time systems which perform these tasks.

1 Introduction

Gestures are an important aspect of human interaction, both interpersonally and in the context of man-machine interfaces. There are many facets to the modeling and recognition of human gesture: gestures can be made by hands, faces, or the entire body, they can be static or dynamic, person-specific or cross-cultural, tied to linguistic utterances or meaningful in their own right. Here we consider vision-based analysis of user-specific *spatio-temporal* gestures, which can be described as a set of poses or expressions observed over a finely sampled temporal sequence. We will focus on methods which learn a task-specific representation of hand and face gestures for use in real-time recognition and tracking tasks.

Our system achieves robust real-time performance by exploiting the principle of using only as much "representation" as needed. Hands and faces are complex

3D articulated structures, whose kinematics and dynamics are difficult to model with full realism. Consequently, instead of performing model-based reconstruction and attempting to extract explicit 3D model parameters (for example see [4, 8, 9]), we use a direct approach which represents the object performing the gesture with a vector of similarity scores to a set of 2-D views. With this approach we can perform recognition and tracking on objects that are either too difficult to model explicitly or for which a model recovery method is not feasible in real time.

As we shall see below, our appearance-based approach affords several advantages, such as the ability to form a sparse representation that models only the poses of the hands that are relevant to desired gestures, and the ability to learn the models directly from the data using unsupervised clustering. We combine the dimensionality reduction offered by appearance-based analysis with a supervised learning interpolation stage that maps view model outputs into a task dependent coordinate system, in which recognition and interactive control are straightforward.

2 Appearance-based Representation

We adopt a appearance-based representation of gesture performance, where the appearance of a target object (e.g., the face or hand) is described by its similarity to a set of iconic views that span the set of poses and configurations of the target object.

This approach is related to the idea of view-based representation, as advocated by Ullman [18] and Poggio [14], for representing 3-D objects by interpolating between a small set of 2-D views. Recognition using views was analyzed by Breuel, who established that there are reasonable bounds on the number of views needed for a given error rate [3]. However, the view-based models used in these approaches rely on a feature-based representation of an image, in which a "view" is the list of vertex locations of semantically relevant features. Unfortunately, the automatic extraction of these features remains a difficult problem.

More closely related to our approach is the work of Turk and Pentland[17], who used combinations of low-order eigenvectors to describe a *space* of target appearances. In this way they were able to detect and recognize human faces. Murase and Nayar[12] later generalized this appearance-based approach to accurately recognize a set of industrial objects and determine their pose. The difference between these appearance-based methods and the earlier view-based methods is that precise feature detection is unnecessary (they can be applied directly to edge maps, optical flow or normalized intensity), and that they can capture much larger range of variation in target appearance.

In this paper we are not interested in recognition from static imagery, but rather in real-time analysis of a spatio-temporal pattern, in particular people's hand gestures and facial expressions. To accomplish this we have extended the notion of appearance modeling into the temporal domain by use of elastic spatio-temporal matching. We have also coupled a task-dependent interpolation stage with our appearance analysis framework, providing direct connection between the user's gestures and task control.

Our work differs from other appearance-modeling research in the use of combinations of spatial views instead of eigenvectors to describe image appearance. We use the view with the maximum similarity (minimum distance) to localize the position of the object, and the entire set of view scores at that point to characterize the actual pose of the object. We will use the term "view model" to mean the iconic representation of a single example of a target, and the term "appearance model" to mean the vector of similarities between a target and a set of view models.

The use of sets of normalized-correlation view matches is somewhat less accurate than the eigenvector-based methods for characterizing target appearance (because image variation is less efficiently described), but it has sufficiently less complexity that a real-time implementation was possible with current hardware. Another important advantage of using view matches rather than an eigenrepresentation that it is possible to employ a direct example-based approach rather than assuming a prior statistical model of the data.

2.1 Correlation-Based Similarity

We have chosen normalized correlation to be the similarity measure between an image and a set of spatial view models to obtain real-time performance. Given a set of models indexed by the variable m, $1 \leq m \leq M$, the view-model similarity function is

$$R_m(i,j) = \frac{S_m(i,j) - S'_m(i,j)}{n_m^2 \sigma_I \sigma_{T_m}} \quad , \tag{1}$$

where

$$S_m(i,j) = n_m \sum_{\{u,v \mid d_m(u,v)=1\}} T_m(u,v) I(i+u, j+v) \quad ,$$

$$S'_m(i,j) = \sum_{\{u,v \mid d_m(u,v)=1\}} T_m(u,v) \sum_{\{u,v \mid d_m(u,v)=1\}} I(i+u, j+v) \quad ,$$

and where $T_m()$ is the value of pixels in the image used to define the view model, $I(i,j), 0 < i < W, 0 < j < H$ is the new image being searched, $d_m(i,j)$ is set 1 for valid view model locations and 0 for "don't-care" locations, $n_m = \sum_{i,j} d_m(i,j)$ is the number of valid pixel locations in the view model, and σ_I, σ_{T_m} are the standard deviation of the observed and model images, respectively. We define the spatial maxima and the corresponding image offsets for each view model:

$$r_* = \max_{i,j,m} R_m(i,j), \qquad (\hat{i}, \hat{j}, \hat{m}) = \arg\max_{i,j,m} R_m(i,j).$$

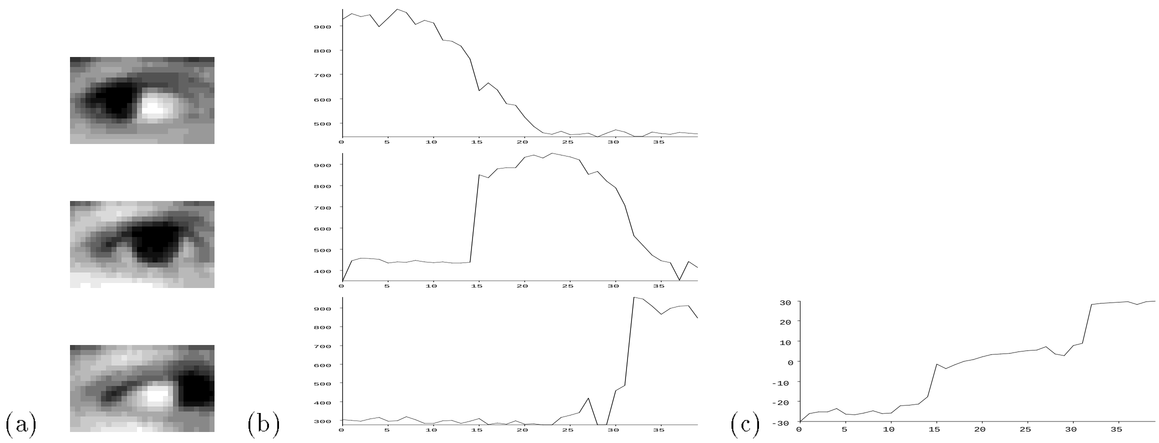


Figure 1: (a) Three spatial views of an eyeball at +30, 0, and -30 of gaze angle. (b) Normalized correlation scores of the view models when tracking a eyeball tracking from approximately -30 to $+30$ degrees of gaze angle with two reported saccades. (c) Interpolated gaze angle showing these saccades, using RBF method described in Section 5.

and store them in a vector of spatial similarity scores, called an "appearance model":

$$\mathbf{r} = [\hat{i}/W, \hat{j}/H, R_1(\hat{i},\hat{j}), R_2(\hat{i},\hat{j})..., R_M(\hat{i},\hat{j})]^T.$$

With a smooth similarity function, the similarity score of a particular view model as the object undergoes non-linear transformations such as rotation, scale, or articulation will be a roughly convex function. The peak of the function will be centered at the parameter values corresponding to the pose of the object used to create the view model. For example, Figure 1(a) shows three images of an eyeball that were used to create view models for gaze tracking; one looking 30 degrees left, one looking center-on, and one looking 30 degrees to the right. Figure 1(b) shows the normalized correlation score for each view model when tracking a eyeball rotating from left to right, with two saccades. Each view model shows a roughly convex curve centered about the gaze angle used to create the view model.

Given a set of view models which sample a transformation parameter finely enough over a range of interest, the spatial appearance model (set of similarity scores) is a sufficient representation of the signal such that one can estimate the actual transform parameters for new views by interpolation. Section 5 will present a method to perform this task. In general, rather than actually estimate 3-D parameters explicitly, our method uses the normalized correlation scores directly for recognition and control. Examples of this will be shown in Section 6.

2.2 Temporal views

We also perform a view-based dimensionality reduction in the temporal domain. We construct temporal view patterns comprised of spatial appearance models observed over time, and use a correlation-based similarity function defined over these dimensions (t, m) to compute a spatio-temporal appearance model. Because of temporal scaling due to different sampling or performance rates, there is variation in time which we need to accommodate. Our solution is to allow the observed sequence to be arbitrarily time-warped to each stored temporal view model before computing a similarity score. The result of this stage is a spatio-temporal appearance model (vector of temporal similarity scores) containing scores which characterize both spatial *and* temporal properties of the input signal.

To find the similarity between two sequences, we again use a normalized correlation metric, but after using the Dynamic Time Warping (DTW) method to temporally align the two sequences, thus allowing the time-course of a gesture to vary. The DTW method involves the use of dynamic programming techniques [1] to solve an elastic pattern matching task, and was originally developed to solve the time alignment problem in the speech and signal processing literature [15]. Note that DTW is a simplification of Hidden Markov Modeling (HMM); they are equivalent for the relatively simple (non-branching) sequences we will be considering.

We use a version of the DTW method that has been modified to match backward in time, so that we can constrain the temporal endpoints of the two sequences to be the same, but allow the starting point to match elastically. To temporally align an observed sequence of spatial view model score vectors, $\mathcal{R} = \{\mathbf{r}[0], \mathbf{r}[1], ..., \mathbf{r}[T]\}$, with a temporal view model of spatial scores $\mathcal{T}_p = \{\mathbf{s}_p[0], \mathbf{s}_p[1], ..., \mathbf{s}_p[T'_p]\}$, where the number of vectors in each sequence are not necessarily equal, we consider a grid for each view model whose horizontal axis is associated with $\mathcal{R}$ and whose vertical axis is associated with $\mathcal{T}_p$. Each element of this grid contains a distance measure $D_{p,i,j}$ measuring the Euclidean distance between $\mathbf{r}[i]$ and $\mathbf{s}_p[j]$. The best time warp will minimize the accumulated distance backward along a monotonic path through the grid from (T, T'_p) to $(0, 0)$. The DTW algorithm uses a partial sum variable, $C_{p,i,j}$, to recursively compute a minimal solution; $C_{p,i,j}$ is defined to be the minimum cost to align $\mathbf{r}[i..T]$ with $\mathbf{s}_p[j..T'_p]$. A backward matching DTW method can be defined with $C_{p,i,j} = D_{p,i,j} + \min(C_{p,i+1,j+1}, C_{p,i+1,j}, C_{p,i,j+1})$, except on the border of the grid, where $C_{p,T,T'_p} = D_{p,T,T'_p}$, $C_{p,i,T'_p} = D_{p,i,T'_p} + C_{p,i+1,T'_p}$, and $C_{p,T,j} = D_{p,T,j} + C_{p,T,j+1}$. When the recursion is complete, the minimum distance between $\mathcal{R}$ and $\mathcal{T}_p$ is simply $C_{p,0,0}$.

This method constrains the observed sequence and the gesture pattern to be aligned at the current time step. However, we do not know *a priori* the actual start point of the gesture in our buffer of observed spatial view scores, so we must relax the requirement that the start point of both sequences also be aligned. We define the score of a gesture model to be the minimum of any of the partial sums which account for all of the time samples in the temporal model, independent of how much of the buffer of currently observed scores is matched: $\mathcal{D}(\mathcal{R}, \mathcal{T}_p) = \min_{0 \geq t \geq T} C_{p,t,0}$.

To find the actual alignment we simply backtrack through $C_{i,j}$ along the directions of partial sum minima. We define a warp function $w_p(j)$, which returns the index of the observed sequence $\mathbf{r}$ corresponding to the j-th element of $\mathbf{s}_p$ vectors in view $\mathcal{T}_p$. This can be computed by setting

$$j^* = 0; \qquad w_p(0) = i^* = \arg \min_{0 \geq t \geq T} C_{p,t,0} \ ,$$

and then iterating

$$d^* = \min(C_{p,i^*+1,j^*}, C_{p,i^*,j^*+1}, C_{p,i^*+1,j^*+1})$$

while incrementing i^* when $d^* = C_{p,i^*+1,j^*}$, j^* when $d^* = C_{p,i^*,j^*+1}$, and both when $d^* = C_{p,i^*+1,j^*+1}$. Setting $w_p(j^*) = i^*$ after each iteration, we have the optimal time-alignment when $i^* = T$ and $j^* = T'_p$.

With the optimal path through the grid in hand, we compute the normalized correlation of these time-aligned sequences, which we define to be g_p where p is the index over the set of temporal views:[1]

$$g_p = \frac{M * T \sum_{m,t} (\mathbf{s}_p)_m[t] r_m[w_p(t)] - \sum_{m,t} (\mathbf{s}_p)_m[t] \sum_{m,t} r_m[w_p(t)]}{(M * T)^2 \sigma_{\mathbf{r}} \sigma_{\mathbf{s}_p}} \ .$$

Finally then, $\mathbf{g}$ is the spatio-temporal appearance model, a vector of combined spatio-temporal view scores computed by cascading the temporal normalized correlation scores (with elastic matching) onto the spatial normalized correlation scores.

2.3 Implementation Issues and Real-time performance

The majority of the computational burden in this method involves finding the set of view model correlation scores. To compute correlation scores, we rely on a special-purpose image processing computer which is designed for quick correlation searching. The view model acquisition, evaluation, and maxima finding are implemented on a Cognex 4400 vision processor developed by the Cognex Corporation (other processing is performed on Sun-4 and SGI Indigo-2 workstations). We have found that when recognizing hand gestures where the

[1] This expression for g_p can be arrived at from the formula defined by Eq. (1), by first replacing m with p (the indexing variable), then u with m and v with t (the dimensions correlation is computed over), then $I(m,t)$ with $r_m[w_p(t)]$ and $T_p(m,t)$ with $(\mathbf{s}_p)_m[t]$ (the objects compared), and finally n_p with $M * T$ (the number of items in the comparison), so that $g_p = R_p(0,0)$.

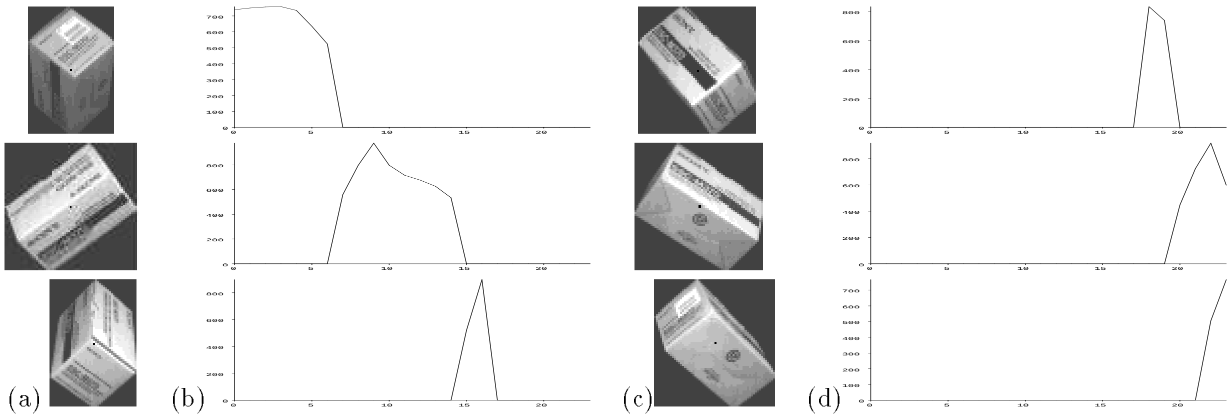


Figure 2: (a,c) Spatial view models automatically acquired from a sequence of images of a rotating box. (b,d) Normalized correlation scores for each model as a function of angle.

hand fills 1/8th to 1/4th of the video frame, it is possible to acquire these images at an input resolution of 128x120 and achieve good recognition performance. We have tested our system at this resolution on examples which used up to 40 models (on a run of the rotating box example); the time required to exhaustively search all models in this case was on the order of 200-300ms. (At this resolution our system can store up to 100 view models in memory accessible by the searching hardware.) Using the predictive search pruning mechanism described in [5], which exploits temporal correlation in the observed view scores, we were able to reduce the processing time for this example to under 100ms. The vision processing for the face interpolation and hand gesture recognition examples described at the end of this paper ran at rates in excess of 10 Hz.

3 Learning View and Appearance Models

A key problem for this approach is how to learn a set of view models that span the target object's range of appearance over both space and time. When objects are non-rigid, either constructed out of flexible materials or an articulated collection of rigid parts, such as hands and faces, the dimensionality of the space of possible appearances is very large. Full enumeration of the space in these cases is intractable even if a complete 3-D model is available. However, many appearances may never be encountered in a particular task due to additional constraints. These may be physical (some joints may not be completely independent), or behavioral (some poses or motions may never be used in the actual communication between user and machine). We therefore use a learning method which derives from training data an appropriate set of view models that are sufficient to characterize the entire range of target appearances.

We use a simple incremental unsupervised clustering scheme. The system begins with one spatial view model taken from an image region defined by the user in the initial frame of the image sequence. The target object is then tracked by using the similarity search function given above. When the maximum search score (r_*) falls below a certain threshold θ, a new view model is added to the appearance model (i.e., the search set) using the image at the offset associated with the current best score. To add a new model, we construct a new template $T(u, v)$ with the values of the pixels in the image in the region covered by the view in the appearance model with the maximum similarity score. In our implementation and in all the results shown in this paper, we set $\theta = 0.7$ and have empirically found this to yield good results.[2] This parameter determines the tradeoff between the number of views used (and thus storage space and computation time) and the accuracy of the appearance model representation (a higher threshold leads to a representation with more views, which can represent finer details).

For acquiring new temporal views, we use the same learning rule, but using the elastic-matching correlation function defined in the previous section. We do assume that during the learning process there is a temporal segmentation signal, so that sequences are presented to the temporal view formation process discretely. Thus, the first sequence is used to define the first temporal view model; additional sequences are added to the set of view models (the spatio-temporal appearance model) when they fail to match against the current set with a sufficient score. During run time there is no temporal segmentation signal needed.

For simple objects and transformations, this clustering method can build an appearance model which adequately covers the entire space of possible target appearances. For example, for a convex rigid body undergoing a 1-D rotation with fixed relative illumination, a small number of view models can match object across a range of the rotation transformation. Figures 2 illustrates this with an example of a box rotating about a single axis.

4 Limitations and extensions of the appearance-based method

Our iconic, appearance-based approach to representing spatio-temporal events has the advantages of being efficient to implement and having a data-driven learning method which makes few assumptions on the exact character of the

[2] This assumes that the target object is indeed present in the entire image sequence. In practice we also use a second threshold, $\theta_0 = 0.6$, and disregard any images that match with $r_* \leq \theta_0$. During the training phase, if the target object leaves the scene or is occluded and the best score falls below this threshold, we will not build spurious view models.

input signal. However, this approach also has several limitations, particularly in the ability of an iconic method to generalize across all possible gestures of a certain semantic class.

An appearance-based method using purely iconic view model templates cannot be expected to perform generalization across multiple users or when a single user performs a gesture that is not well modeled by previously-seen spatio-temporal patterns. We therefore use our method in environment where the user is known (e.g., has logged in), and where the the user is willing to use standard patterns when interacting with the system (note that the user can teach the system the patterns they want to use, they are not fixed ahead of time).

Furthermore, use of normalized correlation as the similarity metric makes our system sensitive to direction of illumination (which will effect shading) and viewing direction, although it provides for invariance to gross illumination changes. However the choice of normalized correlation was only due to hardware restrictions; with other hardware we could add a preprocessing step (such as edge extraction) and apply the method unchanged (see [11] for such an example). In practice, we have simply controlled the illumination, for instance, near-band IR illumination from LEDs is visible to standard CCD cameras but invisible to humans. Such IR illumination has the additional advantage that humans have little skin pigmentation at these wavelengths.

Even with these restrictions there are many domains in which our method can be useful. Examples include a user interacting with a workstation, a driver in an automobile[3], and camera control in teleconferencing. In each of these cases the set of users is small and/or user identity is available through other means, there are clear advantages to control via characteristic hand or face gestures, and the imaging conditions are relatively constant.

5 Interpolation from Visual Input to Task Control

We use a task-dependent interpolation method to map from the set of view model scores to a result vector used for recognition or control. Interpolation is done using a supervised learning paradigm, based on a set of training examples which define the desired result for a particular set of view model outputs. Using the Radial Basis Function (RBF) method presented in [13], we compute a result vector $\hat{\mathbf{y}}$ to be a weighted sum of radial functions centered at an exemplar value:

$$\hat{\mathbf{y}}(\mathrm{g}) = \sum_{i=1}^{n} c_i \mathcal{F}(||\mathbf{g} - \mathbf{g}^{(i)}||) \quad , \tag{2}$$

[3]Plans for selling cars that have cameras pointed at the driver, for drowsiness detection, have already been announced by some manufacturers.

where

$$\mathbf{c} = \mathbf{F}^{-1}\mathbf{y}, \quad (\mathbf{F})_{ij} = \mathcal{F}(||\mathbf{g}^{(\mathrm{i})} - \mathbf{g}^{(\mathrm{j})}||), \quad \mathbf{y} = [\mathbf{y}^{(1)}, ..., \mathbf{y}^{(\mathrm{n})}]^{\mathbf{T}} \quad , \tag{3}$$

$\mathbf{g}$ are the computed spatio-temporal view-model scores, and $\{(\mathbf{y}^{(\mathrm{i})}, \mathbf{g}^{(\mathrm{i})})\}$ are a set of exemplar result and view-model score pairs (which may be scalar or vector valued). $\mathcal{F}$ is the RBF, which in our implementation was simply $\mathcal{F}(\S) = \S$.

We use the interpolation stage to map the observed view model scores into a quantity which is directly useful for a particular task. For example, if we wanted to estimate the eye gaze angle for the example in Figure 1, we could use an RBF interpolator with a one dimensional output space corresponding to gaze angle and three exemplars, containing the view model scores corresponding to each view model angle:

$$\{(y^{(i)}, \mathbf{g}^{(i)})\} = \{(-30, [1.0, 0.3, 0.3]^T), (0, [0.3, 1.0, 0.3]^T), (30, [0.3, 0.3, 1.0]^T)\}$$

Using this RBF configuration, it is straightforward to recover an estimate of the underlying eye gaze angle from the three spatial view model outputs. The interpolated gaze angle is shown in Figure 1(c).

6 Real-time Applications

6.1 Mimicking Facial Expressions

The modeling and tracking of expressions and faces has been a topic of increasing interest recently. Facial animation is a difficult problem due to the sheer complexity of realistic facial models: dozens of degrees of freedom are present in a face, and to control them for computer animation using conventional keyframe or motor control techniques is quite difficult. A much more natural approach is to use one's own face to control the model face parameters. We present a system which tracks facial expressions in real-time without mechanical actuators or make-up, using our interpolated appearance-based vision methods.

This approach follows in the tradition of others who have explored techniques for visual analysis of facial expression [20, 16, 2, 7, 10]. In our method spatial view outputs are interpolated to control the motor states of a 3-D computer graphics face, using the face model employed in Essa and Pentland [7]. The result vector $\hat{\mathbf{y}}$ is defined to be the motor state of the animated face. Training examples are acquired by setting the model face to generate a particular expression, asking the user to mimic the expression, and recording the pair of vision scores and muscle parameters.

We have implemented real-time facial expression tracking, for use in interactive animation or telepresence. Figure 3(a) shows 5 frames of a 125 frame video sequence of a user making a smile and surprise expression are shown. Spatial view models covering the entire face were acquired from a separate training sequence containing the same user making these expressions. The unsupervised

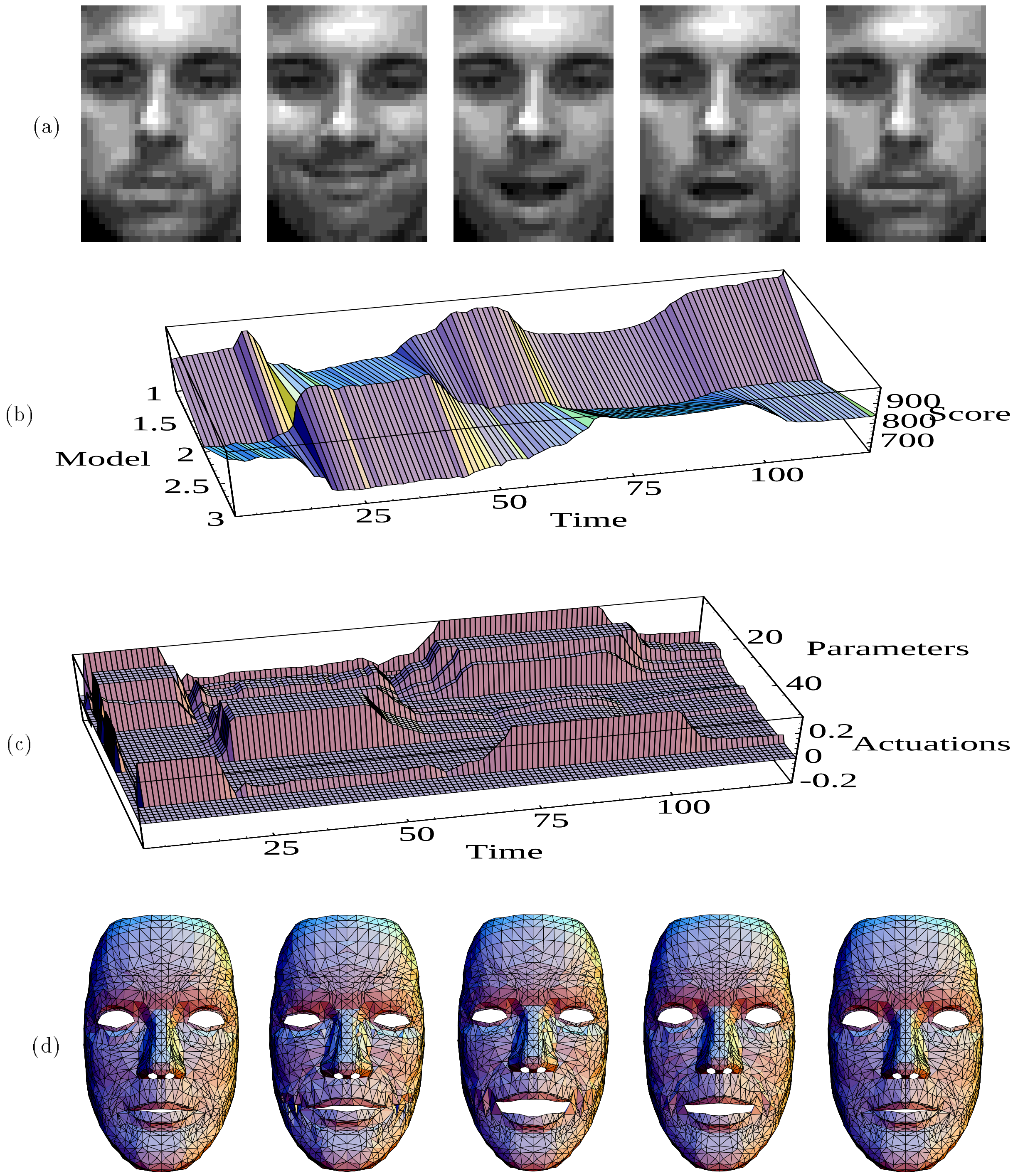


Figure 3: Interactive facial expression tracking in real time. A set of normalized correlation view models are used to characterize facial state, and then used to interpolate a set of motor control parameters for a physically based face model. View models are acquired using unsupervised clustering while the interpolation is trained using supervised learning. See text for details.

(a) (b)

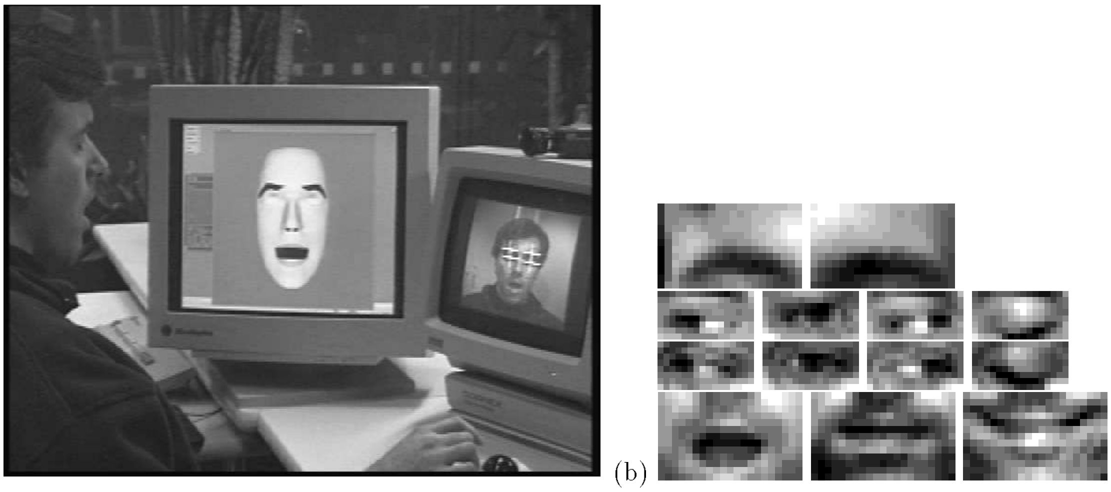


Figure 4: (a) Overview shot of full system. User is on left, vision system and camera is on right, and animated face is in the center of the scene. The animated face matches the state of the users face in real-time, including eye-blinks (as is the case in this shot.) (b) Spatial view models learned by unsupervised clustering method for region-based face mimicking example; independent view models were found for eyes, eye-brows, and mouth regions.

clustering method returned view models corresponding to a neutral, smile, and surprise expression. (No temporal views were used in this example, since we were interested in tracking static pose.) An RBF interpolator was trained using perceptual/motor state pairs for these three expressions; the resulting (interpolated) motor control values for the entire sequence are shown in Figure 3(c), and the rendered facial mesh for five frames of these motor control values is shown in Figure 3(d). Note that the unsupervised clustering method will often return more view models than the actual number of expressions; the RBF method makes no assumptions in this regard.

When there are only a few canonical expressions that need be tracked/matched, this full-face view-based approach is robust and simple. However if the user wishes to exercise independent control of the various regions of the face, then use of full-face models will be overly restrictive. For example, if the user trains two expressions, eyes closed and eyes open, and then runs the system and attempts to blink only one eye, the rendered face will be unable to match it, instead half-closing both eyes. A solution is to decouple the regions of the face which are independent geometrically (and to some degree, independent in terms of muscle effect.) In this approach, separate appearance models are computed for each facial region, and multiple RBF interpolations are performed. Each interpolator drives a distinct subset of the motor state vector.

Figure 4 shows a picture of the set-up of the system as it is being run in an interactive setting, the regions used for the decoupled view models, and the actual view models acquired for the smile/surprise sequence. During run-time, the animated face mimics the facial state of the user, matching in real time the position of the eyes, eyelids, eyebrows and mouth of the user. In the example shown in this picture, the users eyes are closed, so the animated face's eyes are similarly closed.

Realistic real-time performance of animated facial expressions has been achieved with this method: our prototype system combining vision and graphics processing runs in excess of 8 frames/sec with approximately 0.5 sec lag. Typical users can use the system for periods of approximately 15 minutes without having to retrain the view models. In addition, our system offers an extremely low bitrate mechanism for facial teleconferencing. In the above example, vision scores can be encoded in approximately 64 bits per frame; at 8 Hz, only 512 bytes/sec of bandwidth is required. In the domains where the user is known and the imaging conditions relatively controlled, our method can provide a real-time solution to low-bitrate facial coding for teleconferencing and telepresence applications.

6.2 Recognition of Hand Gestures

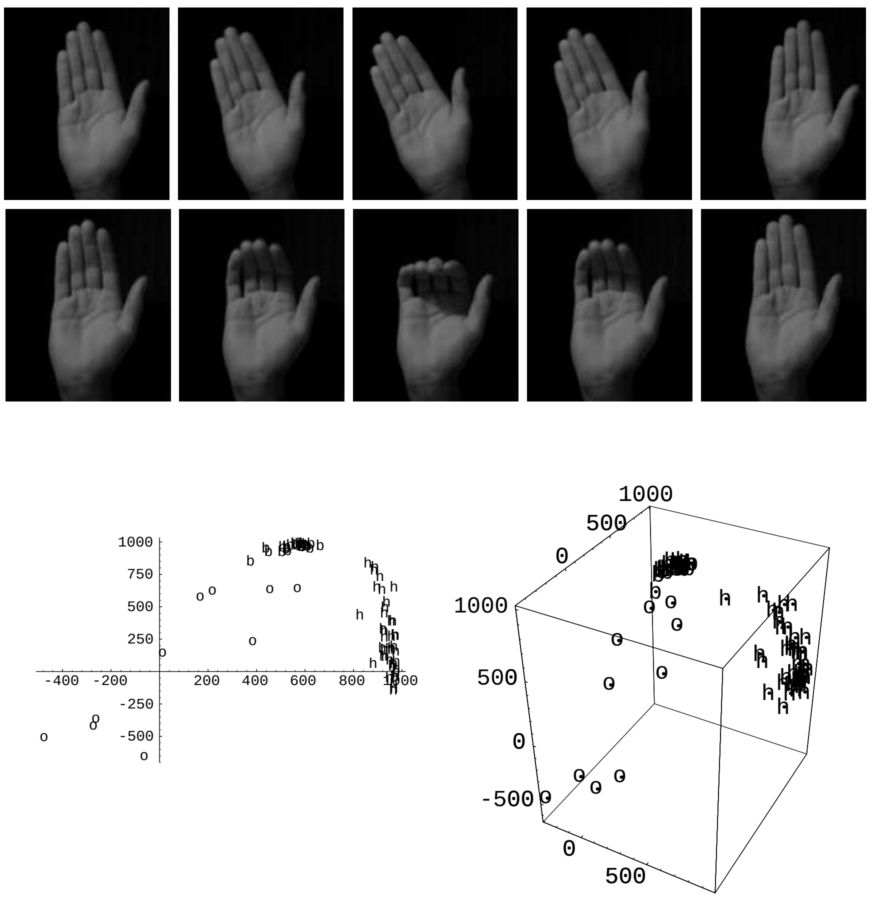
Another application of our appearance-based interpolation framework is the recognition of hand gestures. An RBF-based classifier can be defined by interpolating the spatio-temporal similarity scores into a space whose axes correspond to the recent performance of the patterns we wish to detect. In contrast to the previous example, this classification task calls for a discrete decision rather than a continuous modulation of an input signal. We achieve this by selecting only the maximal component of the interpolated result.

We tested our system on a recognition task with two target gestures. These patterns were selected to be a simple interface to a video conference control system, and are arbitrary. Fourty-two examples of a "hello" gesture were collected, twenty-six examples of "good-bye" and ten examples of other gestures intended to generate false alarms in the classifier. Each user is free to define "hello" or "goodbye" to be any repeatable spatio-temporal pattern. We assume a training phase where the gesture is performed in front of a known background (and we set $d_m(i,j)$ accordingly). Users performed the example gestures discretely, so that temporal segmentation was provided in the training phase.

Figure 5(a,b) shows a representative example of a hello and a goodbye gesture produced by one user. For each trial we randomly selected a subset of gestures to train the classifier, and tested on the remaining gestures. "Hello" and "good-bye" gestures in the training set were input to the unsupervised clustering procedure, which computed spatial and temporal views as described above. In each run five spatial views and two or three temporal views were found by the clustering procedure. The resulting spatio-temporal appearance model scores of these view models proved to be a good classification mechanism. Figure

(a)

(b)



(c) (d)

Figure 5: Formation of spatial and temporal view models for hand gesture analysis. Example of (a) "hello" and (b) "good-bye" gesture are shown. Spatial and temporal view models were found for these gestures, as described in the text. (c,d) Scatter plot of spatio-temporal correlation feature vector on gestures in test set (h=hello, b=bye, o=other). (c) shows case where 2 temporal models were found, (d) shows case with three temporal models.

(a)

n_{train}=3	(actual) hello	(actual) bye	(actual) other
(predicted) hello	3928	104	68
(predicted) bye	0	2492	8
(predicted) other	68	239	593

(b)

n_{train}=39	(actual) hello	(actual) bye	(actual) other
(predicted) hello	2069	18	13
(predicted) bye	0	1297	3
(predicted) other	2	36	462

(c)

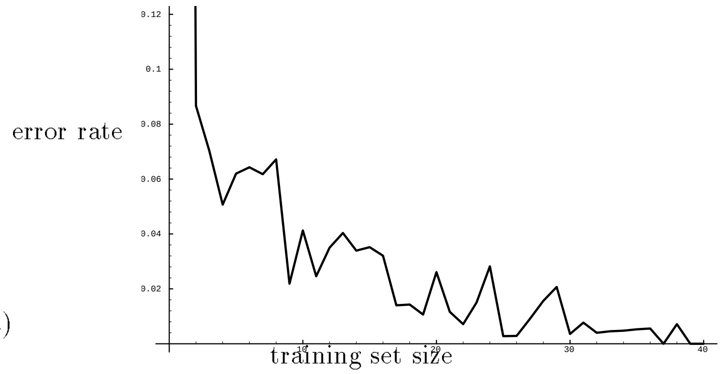


Figure 6: Classification of spatio-temporal view model scores using RBF-based method described in text. (a) Confusion matrix obtained using RBF classification method with a training set containing one example of each gesture; rows indicate predicted gesture, columns indicate actual gesture ($n_{train} = 3$, $n_{test} = 75$, results summed over 100 runs, for a total of 7500 test trials). (b) Confusion matrix obtained using training set containing half of all gestures (n_{train}=39, n_{test}=39, results summed over 100 runs, for a total of 3900 test trials). (c) Plot of error rate as a function of training set size. Error rate ranged from 6.5% for $n_{train} = 3$ to 1.8% for $n_{train} = 39$.

5(c,d) shows example scatter plots of temporal appearance vectors ($\mathbf{g}$), labeled with the actual gesture (Hello, Bye, Other). Figure 5(c) shows a case where two temporal view models were found, this leading to a 2-D spatio-temporal appearance vector. Figure 5(d) shows a case where 3 temporal view models were found for the data resulting in 3-D spatio-temporal appearance vector. In both cases a clear separation of the target patterns is present in the spatio-temporal appearance vectors.

To perform classification, we configured an RBF interpolator with a three dimensional output space and an input space corresponding to the dimensionality of the spatio-temporal appearance model. The RBF was defined with exemplars of the form $(\mathbf{y}^{(i)}, \mathbf{g}^{(i)})$, where $\mathbf{g}^{(i)}$ are the spatio-temporal appearance vectors in the training set, $\mathbf{y}^{(i)}$ is set to $[1,0,0]^T$ if the i-th gesture in the training set is a "hello" gesture, $[0,1,0]^T$ if "good-bye", and $[0,0,1]^T$ if it is a conflictor or no gesture ("other"). We classified all of the gestures in the test set, defining the predicted class based on the index of the largest interpolated score in the result vector. We classified the gesture "hello" if the first element was largest, "good-bye" if the second was largest, and "other" if the third was largest.

Figures 6 shows the classification results for different training set sizes, The confusion matrices for a training set of 3 and 39 gestures from the full set of 78, are shown, based on data summed over 100 trial runs. (The graph is estimated from data summed over 10 trial runs per training set size. With only three training examples (one of each type, randomly selected) we obtained a remarkable success rate (ratio of the number of correct trials to the total number of trials) of 93.5%. When the training set size was allowed to increase to be half the size of the data set, performance increased to 98.2%.

7 Summary and Conclusions

We have developed and implemented a real-time system for learning, tracking, and recognizing complex objects and gestures. The use of an appearance-based representation allows us to model (and search) only the portion of an object's appearance-space which is actually used by a user in the gestures to be analyzed. Our appearance-based approach also allow analysis without having to recover exactly the underlying object pose parameters. Using task-dependent interpolation based on the Radial Basis function method, we have been able to achieve fast and robust analysis and synthesis of facial expressions, and accurately recognize hand gestures using only conventional video camera images as input. This system has promise as a new approach in the interactive animation, video tele-conferencing, and personalized interface domains.

References

[1] Bellman, R. E., *Dynamic Programming.* Princeton, NJ: Princeton Univ. Press, 1957.

[2] Beymer, D.J., "Face recognition under varying pose", *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 756-761, Seattle, WA, 1994.

[3] Breuel, T., "View-based Recognition", *IAPR Workshop on Machine Vision Applications*, Tokyo, Japan, 1992.

[4] Cipolla, R., Okamotot, Y., and Kuno, Y., "Qualitative visual interpretation of 3D hand gestures using motion parallax", *IAPR Workshop on Machine Vision Applications*, Tokyo, Japan, 1992.

[5] Darrell, T. and Pentland, A., "Space-Time Gestures" Proceedings IEEE CVPR-93, New York, 1993.

[6] Darrell, T., and Pentland, A., "Attention-driven Expression and Gesture Analysis in an Interactive Environment," Int'l Workshop on Face and Gesture Recognition, Zurich, Switzerland, June 26-28., 1995.

[7] Essa, I., and Pentland, A. P., "A vision system for observing and extracting facial action parameters", In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, 1994.

[8] Fukumoto, M., Mase, K., and Suenaga, Y., "Real-Time Detection of Pointing Actions for a Glove-Free Interface", *IAPR Workshop on Machine Vision Applications.* Tokyo, Japan, 1992.

[9] Ishibuchi, K., Takemura, H., and Kishino, F., "Real-Time Hand Shape Recognition using Pipe-line Image Processor", *IEEE Workshop on Robot and Human Communication*, pp. 111-116, 1992.

[10] Mase, K., "Recognition of facial expressions for optical flow", *IEICE Transactions, Special Issue on Computer Vision and its Applications*, E 74(10), 1991.

[11] Moghaddam, B., and Pentland, A., "Probabilistic Visual Learning for Object Detection", *Proceedings of the International Conference on Computer Vision 1995 (ICCV'95)*, Cambridge, MA, June 1995.

[12] Murase, H.,and Nayar, S. K., "Learning and Recognition of 3D Objects from Appearance", *Proc. IEEE Qualitative Vision Workshop*, New York City, pp. 39-49, 1993.

[13] Poggio, T., and Girosi, F., "A theory of networks for approximation and learning", MIT AI Lab TR-1140, 1989.

[14] Poggio, T., and Edelman, S., "A Network that Learns to Recognize Three Dimensional Objects", *Nature*, Vol. 343, No. 6255, pp. 263-266, 1990.

[15] Sakoe, H., and Chiba, S., "Dynamic Programming optimization for spoken word recognition", *IEEE Trans. ASSP*, Vol. 26, pp. 623-625, 1980.

[16] Terzopoulus, D., and Waters, K., "Analysis and synthesis of facial image sequences using physical and anatomical models", *IEEE Trans. PAMI*, 15(6):569–579, 1993.

[17] Turk, M., and Pentland, A. P., "Eigenfaces for Recognition", *Journal of Cognitive Neuroscience*, vol. 3, pp. 71-89, 1991.

[18] Ullman, S., and Basri, R., "Recognition by Linear Combinations of Models," *IEEE PAMI*, Vol. 13, No. 10, pp. 992-1007, 1991.

[19] K. Waters and D. Terzopoulos. "Modeling and animating faces using scanned data", *The Journal of Visualization and Computer Animation*, 2:123–128, 1991.

[20] L. Williams. "Performance-driven facial animation", *ACM SIGGRAPH Conference Proceedings*, 24(4):235–242, 1990.

[21] Y. Yacoob and L. Davis, "Computing spatio-temporal representations of human faces", *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, pp. 70-75, Seattle, WA, 1994.