Real-time tracking and imitation of facial expression

Xiang Cao[*a], Baining Guo[**b]
[a]Dept. of Computer Science and Technology, Tsinghua Univ.; [b]Microsoft Research, Asia

ABSTRACT

We present a system for tracking facial expression and head pose with one camera and no special markers, and generating a face animation to imitate these expressions simultaneously. The tracking is based on Gabor wavelet coefficients (jets) of the facial feature points, a saccadic searching strategy, and a model-based adjustment. Using Principal Components Analysis (PCA), we established an expression model and use it to drive the imitation animation. The system runs in real-time and the tracking shows robustness against illumination/background variation, accumulative error and partial occlusion. The imitation animation captures both the expression and the head pose without 3D information.

Keywords: facial feature points, expression tracking, eigen-expression, imitation animation

1. INTRODUCTION

As internet chatting becomes increasingly popular, people will not be satisfied with just text or voice communication. Since facial expression plays an essential role in natural communication between people, "visual chatting", an on-line communication framework incorporating facial expression, will be more effective than traditional systems. Unfortunately, direct transfer of video of the user is not practical for the limited bandwidth of networks. This motivates the use of a virtual character imitating the user's behavior, or avatar, which is desirable also because it can protect the user's privacy, as well as produce impressive visual effects. The process determined by the application must be real-time and is expected to work well in different environments for different users.

Motivated by this application, we present a system that tracks facial feature points, acquires high-level expression parameters, and generates a cartoon animation to simultaneously imitate the user (Fig. 1). Our system runs in real-time, and the tracking shows robustness to background/illumination variations and accumulative error, as well as partial occlusion and eyeglasses. Without dealing with 3D information, the imitation animation clones not only facial expression but also head pose.

Previous work on facial feature tracking can be divided into two categories: feature detection by analysis of image properties (edges, histograms, etc.) [9, 12], and tracking sets of pre-defined feature points [2, 11]. The former approach can work automatically without initialization and can be somewhat person-independent. But usually this approach is highly empirical and not easily generalized for instances such as people wearing glasses or having a beard, or large background/illumination variations.

Figure 1: User and the imitation animation.

In the case of tracking sets of pre-defined feature points, simple or modified pattern matching approaches have low computational complexity and may achieve real-time performance, but the result is usually not satisfactory for highly deformable features like the mouth, which contains much of the expression information. This approach is also vulnerable to accumulative error since it lacks structure constraints. An alternative is to convert the tracking problem to an optimization problem under certain constraints on structure or optical flow [2]. This method can produce more precise results but its iterative computation does not allow real-time performance.

* caox00@mails.tsinghua.edu.cn; phone (8610)62777072; AAA-16099, Tsinghua University, Beijing, P. R. China, 100084
** bainguo@microsoft.com; phone (8610) 62617711~5422; Microsoft Research Asia, Sigma Center, Beijing, P.R.China, 100080

Much research has also been done on expression recognition [5, 8], but most of these works focused on giving qualitative recognition results, which is unsuitable for driving precise imitation animations in our application. On the other hand, good performance has been shown in face animation research [3, 7], but most of these methods require dense 3D information (control points, muscle forces, etc.). This kind of information is hard to capture from a single camera without special markers.

In our work, the tracking is based on the Gabor wavelet coefficients (jets) [1], which provide robustness against background /illumination variation, and an efficient saccadic searching strategy [6] simulating mechanism of human-eye is employed. A model-based adjustment is applied to the primitive tracking result of individual points to improve accuracy and robustness. Expression parameters are extracted from the tracking result using a PCA-based model, in which variation of expression and head pose is compactly represented. Using these parameters, the imitation animation is generated.

The rest of the paper is organized as follows. We address the approach for feature points tracking in Section 2. In Section 3, the expression model based on PCA is presented, and we introduce the method to generate the imitation animation using the expression parameters. Section 4 describes the experimental results. We discuss possible extensions for future work and conclude in Section 5.

2. FACIAL FEATURE POINTS TRACKING

Feature points are defined as points on the face with semantic meaning, such as the pupils, tip of the nose, corners of the mouth, and some points on the boundary of the face. They carry most of the information of facial expression, including head pose (Fig. 2). We first perform tracking for each feature point individually, then a model-based adjustment is done to achieve higher accuracy and greater robustness.

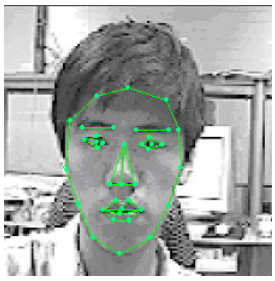


Figure 2: Facial feature points.

2.1 Representation of feature points

The visual appearance of the feature points is described using the Gabor wavelet transform [1, 10]. We follow the definition of *jet* given by Wiscott *et al.* [1] A jet is the set of convolution coefficients for Gabor kernels of different orientations and frequencies around one pixel. A discrete set of four frequencies and six orientations is used in our application, so the jet J of a given pixel $\vec{x} = (x, y)$ in an image $I(\vec{x})$ can be defined as:

$$J_j(\vec{x}) = \int I(\vec{x}')\psi_j(\vec{x} - \vec{x}')d^2\vec{x}' \tag{1}$$

with a family of Gabor kernels

$$\psi_j(\vec{x}) = \frac{k_j^2}{\sigma^2}\exp\left(-\frac{k_j^2 x^2}{2\sigma^2}\right)\left[\exp(i\vec{k}_j\vec{x}) - \exp\left(-\frac{\sigma^2}{2}\right)\right] \tag{2}$$

where vector $\vec{k}_j = \begin{pmatrix} k_{jx} \\ k_{jy} \end{pmatrix} = \begin{pmatrix} k_\nu \cos\varphi_\mu \\ k_\nu \sin\varphi_\mu \end{pmatrix}, k_\nu = 2^{-\frac{\nu+2}{2}}\pi, \varphi_\mu = \mu\frac{\pi}{6}$ (3)

with frequency index $\nu = 0,\ldots,3$, orientation index $\mu = 0,\ldots,5$ and . $j = \mu + 6\nu$. The family of the kernels is self-similar, generated from one mother wavelet by rotation and dilation. (Fig. 3)

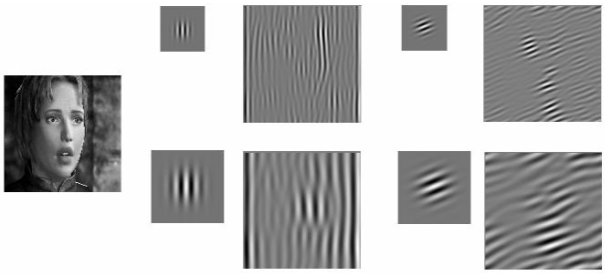


Figure 3: Gabor wavelet kernels (left) and corresponding transformed images (right) of 2 frequencies and 2 orientations (real part displayed)

Thus the jet J is defined as a set of the 24 complex coefficients computed around one pixel, which can be written as

$$J_j = R_j + i \cdot I_j \tag{4}$$

A jet describes the local texture in various sizes and orientations around the pixel, which includes information of both the appearance of the feature point itself and its relationship to other regions in the face, thus it is an efficient description of the feature point The Gabor wavelets are DC-free, so they are robust to brightness variations. They also provide some robustness to translation, rotation and scaling. In order to accelerate the computation of the Gabor wavelet

transform, we apply it in the frequency domain instead of the spatial domain.

2.2 Comparing and matching of feature points

Once the jets of the original feature points have been calculated, matching for each point can be executed for each new frame. To measure the similarity of two jets, we employ the following similarity function:

$$\boldsymbol{S}(J,J') = \frac{\sum_j J_j \cdot J'_j}{\sqrt{\sum_j \left|J_j\right|^2 \cdot \sum_j \left|J'_j\right|^2}} \tag{5}$$

where • represents the dot product $J_j \cdot J'_j = R_j R'_j + I_j I'_j$ and | | represents the module $\left|J_j\right|^2 = R_j^2 + I_j^2$.

This function represents the cosine of the angle between the two vectors, J and J', in a 24 -dimensional complex space. To find the point with the largest similarity in the new frame, with the constraint of limited motion, we apply a saccadic searching strategy with a sparse retinotopic sampling grid [6]. It is known that when humans look for something in a scene, they do not perform a raster-like search with their eyes. Instead, they make a sequence of rapid jumps, or saccades, from the original position of attention until they focus on the target. In addition, human eyes see the focus area more clearly (higher sampling rate) and more dimly (lower sampling rate) on the periphery. Thus a search strategy with the concept of "focus of attention" can be implemented, using the retinotopic sampling grid, whose sampling density decreases with the distance from the center (Fig. 4).

To begin the search, the sampling grid is centered on the previous position of the feature point. The point in the grid with the maximal similarity will be selected as the center for next saccade, i.e. the grid center will be placed to that point for the next search step. The iteration continues until the grid center no longer shifts or a threshold of iteration steps is reached. Points already visited in preceding saccades are disregarded in subsequent saccades. This strategy can be regarded as a coarse-to-fine routine since in many cases large saccades are taken first to jump rapidly to the area of interest, and smaller saccades are subsequently taken to locate the point accurately, as done by human eyes. Although this method does not ensure that the global optimum will be found, good performance is achieved in the specific problem of tracking. In most cases, the iteration converges to the correct point in a few (1-5) steps while preserving the capability to track points with larger motion, since large jumps could be made continuously.

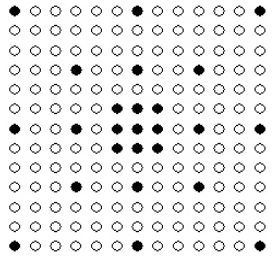


Figure 4: The retinotopic sampling grid. Filled circles denote the sampling points

Since local texture varies substantially near the mouth, some color information is also employed to assist the mouth tracking. Two quartic curves are fitted respectively to the upper and lower boundaries of the lips, and the mouth points are adjusted according to these curves.

2.3 Model-based adjustment

Since the feature points are tracked individually, they can be subject to stochastic noises and errors. Some of the points may occasionally drift to unreasonable positions, due to accidental noises. From analysis of the tracking data, an expression model was established and can be used to correct the result. The accuracy and robustness of tracking are significantly improved by this. These details are discussed in Section 3.1 - 3.2.

3. EXPRESSION MODELING AND IMITATION

Coordinates of the feature points contain the expression information, but the collection of these values is too high-dimensional and unstructured. This raw data is unsuitable for directly driving an imitation animation, mainly for two reasons: *1*. The user's face and the imitating face can be different in size, proportion, and even shape or structure (consider using the tracking data to drive the face of an animal or a robot); direct application of the feature point motion vectors would not only be improper but also sometimes impossible. *2*. Tracking error of a few feature points may lead to an expression impossible for a human to make. For these reasons, expression information should be represented on a higher level, so we employ PCA to extract semantic features of the data.

3.1 Expression model

The Point Distribution Model (PDM) [4] is defined as a set of feature points describing a deformable shape. It describes

both the mean shape and typical variations of a given class of objects. We employ the idea of PDM to describe the structure of the feature points, and use its parameters as a representation of expression.

The aggregation of the tracked feature point positions in a given frame is considered as a sample of the model, which is represented by the feature point vector: $\mathbf{v} = (x_1, y_1, x_2, y_2, \cdots, x_n, y_n)^T$ (6)

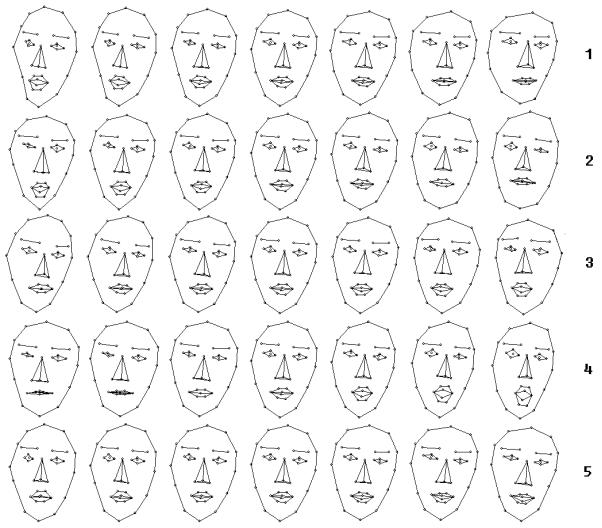


Figure 5: Mean-expressions (center column) and the first 5 groups of eigen-expressions. The right indices indicate the eigenvector number, and $\sigma_i = \sqrt{\lambda_i}$ is the standard deviation for the *i* th eigenvector. (The eigen-expressions are actually motion vectors; here we add them to the mean for display.)

where (x_i, y_i) is the 2D coordinate of the *i* th feature point, and *n* is the number of feature points (in our case *n*=38, thus **v** is a vector of 76 dimensions). In order to eliminate variations in translation, rotation and scaling, the samples are first aligned to their mean. The algorithm for aligning and inverse-aligning two feature point vectors can be found in the appendix of this paper.

We collected 210 samples from the tracking results of three different persons, covering various possible expressions and a considerable range of head poses. A few tracking errors are corrected manually. After aligning all these samples, PCA is applied to them. Similar to mean-faces and eigen-faces in face recognition [13], a "mean-expression" and several "eigen-expressions" are acquired, corresponding to the mean vector $\overline{\mathbf{v}}$ and eigen-vectors $\widetilde{\mathbf{v}}_i$, respectively (Fig. 5).

The mean-expression is neutral in both expression and head pose, while each group of eigen-expressions represents a dimension of expression variation, such as the 1st and 2nd groups modeling the head pose in two perpendicular orientations, and the 4th group representing a gradual transition from "calmness" to "astonishment". All reasonable or valid expressions should be linear combinations of the mean-expression and some eigen-expressions. The primary variations in expressions are concentrated in the first few eigen-expressions, and the variation of the subsequent ones are relatively trivial and consist mainly of stochastic noises. In this sense, the first few eigen-expressions constitute the space of valid expressions, and the residual could be discarded. In our case, we preserve the first ten eigen-expressions, which contain 97% of the total variation of the expressions. After aligning it to the mean vector $\overline{\mathbf{v}}$, any sample vector **v** can be projected to the valid expression space, generating a 10-dimensional expression vector $\mathbf{e} = (e_1, e_2, \cdots, e_m)^T$ (in our case *m*=10). To eliminate over-exaggerated expressions due to errors in tracking, a threshold is applied to each element of **e** :

$$e_i = \begin{cases} 3\sigma_i, (e_i > 3\sigma_i) \\ -3\sigma_i, (e_i < -3\sigma_i) \\ e_i (elsewise) \end{cases} \quad (7)$$

where $\sigma_i = \sqrt{\lambda_i}$ is the standard deviation of the *i* th eigenvector. We use the expression vector as the representation of expressions.

3.2 Model-based tracking adjustment

As mentioned in Section 2.4, the expression model can be used to adjust the tracking result of each frame. It is straightforward to reconstruct the feature point vector $\mathbf{v}'$ from the expression vector **e**.

$$\mathbf{v}' = \overline{\mathbf{v}} + \sum_{i=1}^{m} e_i \widetilde{\mathbf{v}}_i \quad (8)$$

Since the sample vector **v** had been aligned to $\overline{\mathbf{v}}$ before the PCA decomposition, an inverse-alignment should be done to $\mathbf{v}'$ (see *Appendix*) to recover its variation in translation, rotation and scaling.

As long as not too many feature points are tracked in error (usually under 1/3 of the total number), the adjustment gives a satisfactory result, bringing the wrong points to reasonable positions. This is the result of preserving only the

components in the valid expression space of **v**. Tracking does not fail even when a considerable part of the face is occluded (Fig. 11).

3.3 Generation of imitation animation

Using the expression vector, the imitation animation can be generated. The model of the imitation face is also in the form of a PDM, with its own mean-expression $\overline{\mathbf{u}}$ and eigen-expressions $\widetilde{\mathbf{u}}_i$ (i=1,…,*m*) (Fig. 6). The mean-expressions $\overline{\mathbf{v}}$ and $\overline{\mathbf{u}}$ both represent a neutral expression and pose, and the eigen-expressions $\widetilde{\mathbf{v}}_i$ and $\widetilde{\mathbf{u}}_i$ should be correspondent in semantics. For example, $\widetilde{\mathbf{v}}_2$ and $\widetilde{\mathbf{u}}_2$ both indicate the pitch of the head, although they may consist of different numbers of points and have different structures. Thus the imitation model **U** represents the same expression space as the original model **V**, in the sense of semantics. So the generation of the imitation is described as:

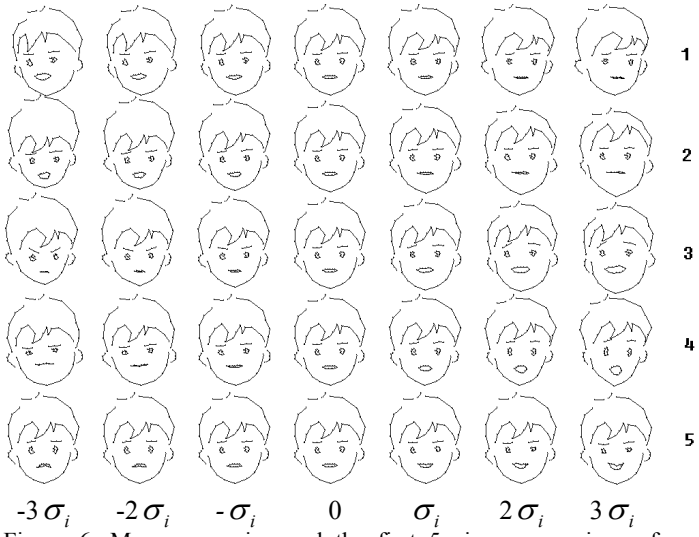


Figure 6: Mean-expression and the first 5 eigen-expressions of an imitation model that corresponds to those of the original model.

1. Align the tracked feature vector **v** to $\overline{\mathbf{v}}$;
2. Decompose **v** to get expression vector **e** under model **V**;
3. Reconstruct **e** to get **u** under model **U**;
4. Inverse-align **u**, using the alignment parameters in Step 1 (see *Appendix*).

In this way, a mapping is established between the original model **V** and the imitation model **U**, in other words, between the tracking data and the imitation animation. The imitation image is then generated simply by connecting the points in **u** by strokes. (Fig. 7)

Without explicit reconstruction of the 3D information, the imitation animation handles precisely not only the facial expression but also the head pose. The imitation model **U** ensures the generation of natural-looking expressions. See Fig. 10 for some examples.

To set up the PDM of the imitation model, 2*m*+1 line-sketches are drawn by hand. Feature points, or landmarks (not necessarily the same as those of the original model **V**, both in number and in semantics) are marked manually on each sketch. One sketch corresponds to the mean expression with $\mathbf{e}=(0,0,\cdots,0)^T$, thus generating a feature point vector $\mathbf{x}^0=(x_1,y_1,x_2,y_2,\cdots,x_l,y_l)^T$, where (x_j,y_j) is the coordinate of the *i* th feature point, and *l* is the number of points. The other 2*m* sketches respectively correspond to the eigen-expressions with $\mathbf{e}=(0,\cdots,0,\pm 3\sigma_i,0,\cdots,0)^T$, *i*=1,…,*m*, which generate vectors $\mathbf{x}^{i+}$ and $\mathbf{x}^{i-}$ (after alignment to $\mathbf{x}^0$). $\overline{\mathbf{u}}$ can be directly derived as $\overline{\mathbf{u}}=\mathbf{x}^0$, yet $\widetilde{\mathbf{u}}_i$ and σ_i need to be calculated from $\mathbf{x}^0$, $\mathbf{x}^{i+}$ and $\mathbf{x}^{i-}$. Ideally,

$$\begin{aligned}\mathbf{x}^{i+}&=\mathbf{x}^0+3\sigma_i\widetilde{\mathbf{u}}_i,\\ \mathbf{x}^{i-}&=\mathbf{x}^0-3\sigma_i\widetilde{\mathbf{u}}_i\end{aligned}\tag{9}$$

but the points marked by hand may not satisfy this linear constraint precisely. It becomes a simple problem of optimization, that is to find the $\widetilde{\mathbf{u}}_i$ and σ_i that minimizes the cost

$$\begin{aligned}c_i&=\left|\mathbf{x}^{i+}-(\mathbf{x}^0+3\sigma_i\widetilde{\mathbf{u}}_i)\right|^2+\left|\mathbf{x}^{i-}-(\mathbf{x}^0-3\sigma_i\widetilde{\mathbf{u}}_i)\right|^2\\&=(\mathbf{x}^{i+}-\mathbf{x}^0-3\sigma_i\widetilde{\mathbf{u}}_i)^T(\mathbf{x}^{i+}-\mathbf{x}^0-3\sigma_i\widetilde{\mathbf{u}}_i)\\&+(\mathbf{x}^{i-}-\mathbf{x}^0+3\sigma_i\widetilde{\mathbf{u}}_i)^T(\mathbf{x}^{i-}-\mathbf{x}^0+3\sigma_i\widetilde{\mathbf{u}}_i)\end{aligned}\tag{10}$$

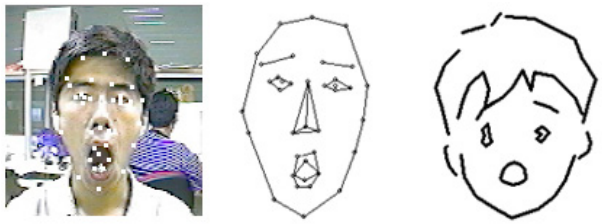

original frame (tracked points displayed) tracked points imitation

Figure 7: Correspondence between tracked points and imitation.

when $\frac{\partial c_i}{\partial \tilde{\mathbf{u}}_i} = -6\sigma_i(\mathbf{x}^{i+} - \mathbf{x}^0 - 3\sigma_i\tilde{\mathbf{u}}_i) + 6\sigma_i(\mathbf{x}^{i-} - \mathbf{x}^0 + 3\sigma_i\tilde{\mathbf{u}}_i) = 0$ i.e. $\tilde{\mathbf{u}}_i = \frac{\mathbf{x}^{i+} - \mathbf{x}^{i-}}{6\sigma_i}$ (11)

the cost is minimized. Note that $\tilde{\mathbf{u}}_i$ is the eigenvector corresponding to the *i* th eigenvalue and should be a normalized vector, so

$$\sigma_i = \frac{\left|\mathbf{x}^{i+} - \mathbf{x}^{i-}\right|}{6} \tag{12}$$

3.4 Imitation by tracking fewer points

By observing the eigen-expressions of the original model, we found that a few feature points indicate most of the information, such as the tip of the brow and some points on the face boundary. Other points, such as points on the nosewings, usually move accordingly to these major points. So we may acquire the expression parameters by tracking only a few of the feature points. In mathematics, this means estimating **e** using only some of the elements of **v**.

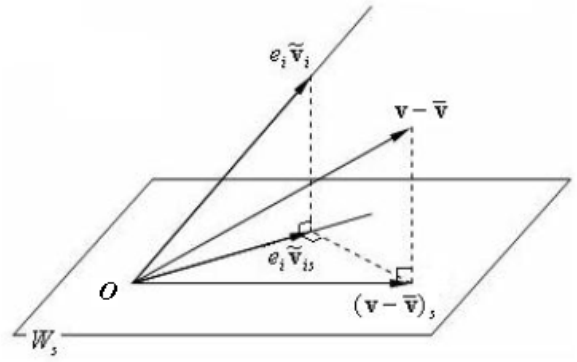


Figure 8: Estimation of e_i

For feature point vector $\mathbf{v} = (v_1, v_2, \ldots, v_n)^T$, mean vector $\bar{\mathbf{v}} = (\bar{v}_1, \bar{v}_2, \ldots, \bar{v}_n)^T$ and eigenvector $\tilde{\mathbf{v}}_i = (\tilde{v}_{i1}, \tilde{v}_{i2}, \ldots, \tilde{v}_{in})^T$, the original criterion for calculating e_i is to minimize

$$d = \left|(\mathbf{v} - \bar{\mathbf{v}}) - e_i\tilde{\mathbf{v}}_i\right| = \sum_{j=1}^{n}(v_j - \bar{v}_j - e_i\tilde{v}_{ij})^2 \tag{13}$$

Yet if we know only *k* (*k*<*n*) elements of **v**: $v_1, v_2, \ldots v_k$, (without loss of generality, the first *k* elements), the minimization condition of *d* cannot be calculated. In this case the most reasonable criterion for computing e_i is to minimize

$$d_s = \left|(\mathbf{v} - \bar{\mathbf{v}})_s - e_i\tilde{\mathbf{v}}_{is}\right| = \sum_{j=1}^{k}(v_j - \bar{v}_j - e_i\tilde{v}_{ij})^2 \tag{14}$$

where $(\mathbf{v}-\bar{\mathbf{v}})_s = (v_1 - \bar{v}_1, v_2 - \bar{v}_2, \ldots, v_k - \bar{v}_k, 0, \ldots 0)^T$ and $\tilde{\mathbf{v}}_{is} = (\tilde{v}_{i1}, \tilde{v}_{i2}, \ldots, \tilde{v}_{ik}, 0, \ldots 0)^T$ are respectively the projections of $(\mathbf{v}-\bar{\mathbf{v}})$ and $\tilde{\mathbf{v}}_i$ in the subspace W_s, which consists of the first *k* dimensions of the whole space. To achieve this minimization, let

$$\frac{\partial d_s}{\partial e_i} = -2\sum_{j=1}^{k}(v_j - \bar{v}_j - e_i\tilde{v}_{ij})\tilde{v}_{ij} = 0 \quad \text{, then we get} \quad e_i = \frac{\sum_{j=1}^{k}(v_j - \bar{v}_j)\tilde{v}_{ij}}{\sum_{j=1}^{k}\tilde{v}_{ij}^2} \tag{15}$$

The estimated expression vector **e** can be used both for generating the imitation animation and for interpolating the feature points that have not been tracked (Fig. 13). When more feature points tracked, the redundancy contributes to robustness.

4. EXPERIMENTS

We ran the system on a Pentium III 1GHz PC, using a 3Com USB camera. The size of each frame is 256×256 pixels. The whole process of tracking and imitation is done at the rate of 12.5 fps. Most of the computational cost is spent on Gabor wavelet filtering. No camera calibration is needed to set up the system.

We track 38 feature points on the human face (Fig. 1). To initialize the jets of the facial feature points of a certain user, an approximately frontal image is captured, and feature points are manually marked on that image. Once the jets for a certain person are calculated and saved, no additional initialization is needed for this person in later tracking.

The tracking shows robustness to translation, rotation, and scaling of the face, and also lighting variation, because of the Gabor wavelet transform. Moreover, variations of background show little effect on tracking. Head rotations are allowed in the range of about ±35° in the three axis directions (see Fig. 10 for some examples). The system also shows robustness to accumulative error. Even if in some frames the tracking gets lost (in the extreme situation the user hides himself entirely from the camera), it can recover in future frames. Because of the model-based adjustment, the tracking does not fail even when a considerable part of the face is occluded (Fig. 11). In experiments we also find that the jets do

not discriminate greatly among different people. Jets recorded for one person often work well for another, since feature points of different people are similar in nature (Fig. 12).

As a demonstration, we implement a simple imitation line-sketch of a boy's face, made up of 83 feature points. According to the ten eigen-expressions used, 21 sketches are drawn and their corresponding feature points are marked manually to set up the imitation model. Animation of any other characters can be generated in the same manner. By including more points in the imitation model, more refined animation can easily be created. The facial expression, head pose, as well as translation of the user are incorporated well in the animation.

As described in Section 3.4, we may track fewer feature points to get the expression parameters. In an experiment we track nine points, which contain most information of facial expression and head pose. The imitation appeared good. Not much difference can be observed between it and the result that uses all the feature points. But tracking lost part of its robustness in this case, since little redundancy is included in so few feature points (Fig. 13).

5. CONCLUSION AND FUTURE WORK

In this work, we implemented a real-time system for tracking and imitating facial expressions. A real-time, robust feature tracking approach customized for expression imitation, and a framework for generating imitation animation without 3D information are presented.

Since we have extracted the high-level expression parameters, it is a natural extension of our work to do expression recognition based on this, for either individual frames or sequences, by establishing a classifier in the expression space. Head pose estimation can also be achieved by analyzing the first a few elements of the expression vector. Finer imitation models of various characters, incorporating color, texture, and 3D structure can be fulfilled by trivial modification of current models. Finally, a network version of our system can be realized to illustrate its application of visual chatting.

APPENDIX: ALIGNING AND INVERSE-ALIGNING TWO FEATURE POINT VECTORS

The positions of feature points acquired in the tracking can have variations in translation, rotation, and scaling. These variations should be eliminated when we analyze the expression parameters from them. To achieve this, we align the feature point vector to a reference vector (usually the mean vector of the expression model) before its decomposition to expression space.

For feature point vector $\mathbf{v} = (x_1, y_1, x_2, y_2, \cdots, x_n, y_n)^T$, an Alignment Transform $\boldsymbol{A}$ is defined as: for each point $(x_i, y_i)^T$ in $\mathbf{v}$,

$$\boldsymbol{A}(s,\theta,t_x,t_y)\begin{bmatrix} x_i \\ y_i \end{bmatrix} = \begin{bmatrix} (s\cos\theta)x_i - (s\sin\theta)y_i + t_x \\ (s\sin\theta)x_i + (s\cos\theta)y_i + t_y \end{bmatrix} \tag{16}$$

which includes a scaling by s, a rotation byθ, and a translation by $(t_x, t_y)^T$. The optimal alignment for vector $\mathbf{v}_2$ to vector $\mathbf{v}_1$ minimizes the weighted sum

$$E = (\mathbf{v}_1 - \boldsymbol{A}(\mathbf{v}_2))^T \mathbf{W}(\mathbf{v}_1 - \boldsymbol{A}(\mathbf{v}_2)) \tag{17}$$

where $\mathbf{W}$ is a diagonal matrix of weights for each point. The parameters of $\boldsymbol{A}$ can be computed by a least-squares approach, which is described in [4].

In the process of model-based adjustment and generating imitation, we need to use inverse-alignment to restore the variation of rotation, translation and scaling of feature points after the reconstruction from the expression space. Knowing the alignment transform $\boldsymbol{A}(s,\theta,t_x,t_y)$, the corresponding inverse-alignment transform can be derived as:

$$\begin{aligned} \boldsymbol{A}^{-1}(s,\theta,t_x,t_y)\begin{bmatrix} x_i \\ y_i \end{bmatrix} &= \boldsymbol{A}(s^{-1},-\theta,-\frac{t_x\cos\theta + t_y\sin\theta}{s},\frac{t_x\sin\theta - t_y\cos\theta}{s})\begin{bmatrix} x_i \\ y_i \end{bmatrix} \\ &= \begin{bmatrix} (\cos\theta/s)x_i + (\sin\theta/s)y_i - \dfrac{t_x\cos\theta + t_y\sin\theta}{s} \\ -(\sin\theta/s)x_i + (\cos\theta/s)y_i + \dfrac{t_x\sin\theta - t_y\cos\theta}{s} \end{bmatrix} \end{aligned} \tag{18}$$

The process of alignment and inverse-alignment for generating imitation is illustrated in Fig. 9.

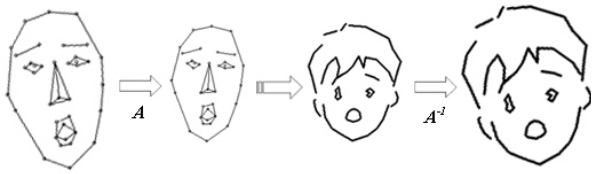
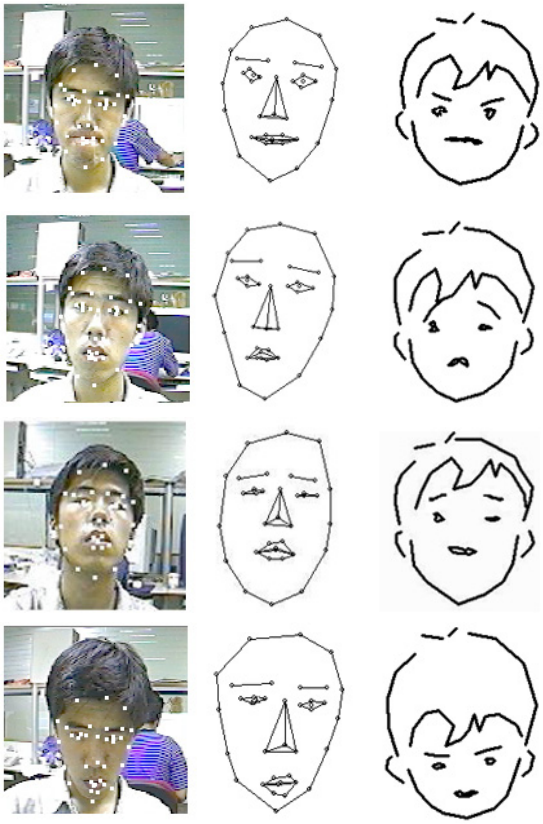


Figure 9: Alignment and inverse-alignment

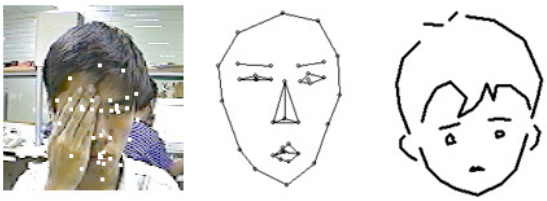
REFERENCES

1. L. Wiskott, J.M. Fellous *et al*., "Face recognition by elastic bunch graph matching", *Proc Intl. Conf. on Image Processing, ICIP'97*, pp. I 129-132, Santa Barbara, 1997.
2. S. B. Gokturk, J. Y. Bouguet, and R.Grzeszczuk. "A data-driven model for monocular face tracking", *Proc. Intl. Conf. on Computer Vision, ICCV 2001*, Vancouver, 2001.
3. J. Noh and U. Neumann. "Expression cloning", *Proc. ACM SIGGRAPH '01,* pp 277-288, Los Angeles, 2001
4. T. F. Cootes, C. J. Taylor *el al.*, "Active shape models – their training and application", *Computer Vision and Image Understanging,* 61(1) pp38-59, 1995
5. Y. Yacoob and L. Davis, "Recognizing facial expressions by spatio-temporal analysis", *Proc. 12th Intl. Conf. on Pattern Recognition, ICPR,* pp 747-749, Jerusalem, 1994.
6. F. Smeraldi, O. Carmona, and J. Bigün, "Saccadic search with Gabor features applied to eye detection and real-time head tracking", *Image and Vision Computing* 18 (2000) pp 323–329, 2000
7. Y. Lee, D. Terzopoulos, and K. Waters, "Realistic modeling for facial animation", *Proc. ACM SIGGRAPH'95*, pp.55-62, Los Angeles, 1995
8. H. Hong, H. Neven, and C. von der Malsburg, "Online facial expression recognition based on personalized galleries", *Proc. 3rd Int. Conf. on Automatic Face and Gesture Recognition, FG'98*, pp.354-359, Nara, 1998
9. Y. Tian, T. Kanade, and Jeffrey F. Cohn, "Multi-State Based Facial Feature Tracking and Detection", *tech. report CMURITR99-18,* Robotics Institute, Carnegie Mellon University, 1999.
10. M. Potzsch, T. Maurer, *et al.* "Reconstruction from graphs labeled with responses of Gabor filters", *Proc. Intl. Conf. of Artificial Neural Networks,* pp. 845--850, Bochum, 1996.
11. J. Cohn, A. Zlochower *et al.,* "Feature-point tracking by optical flow discriminates subtle differences in facial expression"**,** *Proc. 3rd Intl. Conf. on Automatic Face and Gesture Recognition, FG '98,* pp. 396 – 401, Nara, 1998
12. A**.** Colmenarez, B. Frey, and T. Huang, "Detection and tracking of faces and facial features", *Proc Intl. Conf. on Image Processing, ICIP'99***,** Kobe, 1999
13. M. Turk and A. P. Pentland, "Eigenfaces for recognition", *Journal of Cognitive Neuroscience,* 3(1) pp.71-86, 1991.



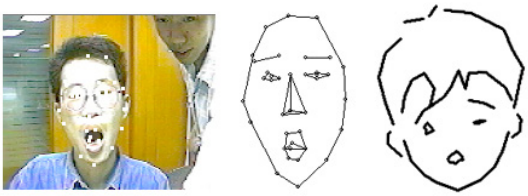
original frame tracked points imitation

Figure 10: Results for various expressions and head



original frame tracked points imitation

Figure 11: Result when part of the face is occluded.



original frame tracked points imitation

Figure 12: Using the jets of one person to track another one. Robustness against the affect of eyeglasses and variation of background and illumination is also illustrated.

Points to be tracked original frame recovered points imitation

Figure 13: Tracking fewer feature points to capture the expression. The green points show the tracked feature points.