3.0 FACIAL FEATURE POINT TRACKING

The face is an interface of nonverbal communication, which can represent a subject's social feeling or reveal his brain function through the use of expressions. People activates facial action mainly by controlling the individual or combined motions of four facial features: brows, eyes, nose and mouth. These are the most attractive features on the facial surface because they have high textures, and symbolize the underlying muscle activations. An observer may recognize easily and directly the messages transmitted from the movement of facial features. Optical flow in an image sequence has been used to track highly textured regions reliably for extracting the motion information of facial features to be used in further recognition process. Optical flow provides an estimate of the movement of facial feature points. Since our goal is to discriminate subtly different facial expressions and to estimate the expression intensity, the tracking algorithm must have high accuracy, be sensitive to subpixel motion, and be able to deal with relatively large facial movements.

3.1 Dot Tracking and Reliability of Feature Point Selection

Since FACS specifies that each AU corresponds to the movement of a single muscle, we design an automatic feature point tracking to extract motion information of facial feature actions based on the movement of facial feature points (which represents the underlying muscle activations) across an image sequence. This will allow realization of the AU actions of facial expressions for encoding "expression units" constructed by individual AUs or AU combinations.

It is important to make sure that the locations and movement of feature points can exactly reflect the AU activation. The following method is used to track the facial features. We attach black dots to specific points on faces of the subjects; a black dots

have the same radius equivalent to 15 pixels on a face image of the size of 490 x 640. We use the template matching (the correlation coefficient ρ) method to track dot movements during the facial action by considering the highest value of the correlation coefficient between the dot template F and the gray values in a target region $I(x)$ centered at position x,

$$\rho(x) = \left| \frac{\sum_{r\in R}[F(r)-\overline{F}][I(x+r)-\overline{I}(x)]}{\sqrt{\sum_{r\in R}[F(r)-\overline{F}]^2}\sqrt{\sum_{r\in R}[I(x+r)-\overline{I}(x)]^2}} \right| \qquad 0.0 \le \rho \le 1.0 \qquad (3\text{-}1)$$

where r denotes a position within the circular region of the dot template whose area is R (radius = 15 pixels), $I(x+r)$ denotes the gray value of the image at position $x+r$, $\overline{F}$ and $\overline{I}(x)$ are the average gray values in circle regions of the dot template and the image, respectively, and x is in a search region m of 60 x 60 pixels, which is large enough to reach the maximum dot movement but not too large to reach any neighboring dot template. If the correlation coefficient ρ is closer to 1.0, there is very strong similarity between the dot template and the target region; otherwise, they are less correlated. Utilizing the relations

$$\sum_{r\in R} F(r) = R\overline{F} \qquad (3\text{-}2)$$

and

$$\sum_{r\in R} I(x+r) = R\overline{I}(x) \qquad (3\text{-}3)$$

the above equation is simplified to achieve the efficient computation,

$$\rho(x) = \left| \frac{\sum_{r\in R} F(r)I(x+r) - \overline{I}(x)\sum_{r\in R}F(r) - \overline{F}\sum_{r\in R}I(x+r) + \sum_{r\in R}\overline{F}\overline{I}(x)}{\sqrt{\sum_{r\in R}[F(r)-\overline{F}]^2}\sqrt{\sum_{r\in R}I^2(x+r) - 2\overline{I}(x)\sum_{r\in R}I(x+r) + \sum_{r\in R}\overline{I}^2(x)}} \right|$$

$$= \left| \frac{\sum_{r\in R} F(r)I(x+r) - R\overline{F}\overline{I}(x)}{\sqrt{\sum_{r\in R}[F(r)-\overline{F}]^2}\sqrt{\sum_{r\in R}I^2(x+r) - R\overline{I}(x)\overline{I}(x)}} \right| \qquad (3\text{-}4)$$

The first term in the denominator and the average gray value of dot template $\overline{F}$ can be calculated first and assigned as constant values for quick processing.

For the first frame of each image sequence, we used the computer user interface which we designed to initially locate the center of each dot marked by a cross (+) on the screen. By observing or automatically tracking the motion of the dots in the remaining frames, we realized the AU actions of facial expressions and understood the underlying muscle activations (Figure 6). This allowed us to correctly locate controlled positions of facial feature points and measure the reliability of the selection of feature points based on which the automatic tracking using the optical flow will begin.

Using the template matching method to track dots is inaccurate for two reasons: the dots on screen may become deformed and are sometimes affected by the reflections due to lighting (especially if red or white dot templates are used) on subjects with dark skin color. The method is also much slower when compared to selected optical flow tracking discussed before, especially when the number of dots and search regions are increased.

For the optical flow tracking, a set of feature points will be initially selected on the first frame of each image sequence (from the neutral expression to the peak expression in an arbitrary length of time). We suggest to select 4 facial feature points around the contour of each brow, 4 points around each eye, 14 points around the nose contour, 10 points surround the lip contour, and 3 points along each cheek bone (below each lower eyelid) as shown in Figure 7. This can be done by an operator using a computer, mouse, and user interface as shown in Figure 2. The manually selected feature points are then automatically tracked using optical flow in the remainder of the sequence. The motion of these facial feature points simulates the facial muscular actions corresponding to AUs. The displacements of these feature points are directly proportional to the AU expression intensities.

The reliability of the feature point selection has been confirmed in experiments by two experienced operators. The first frames of 80 image sequences are independently selected by two different operators. The optical flow method was used to automatically track those selected feature points and inter-observer reliability was evaluated. Our

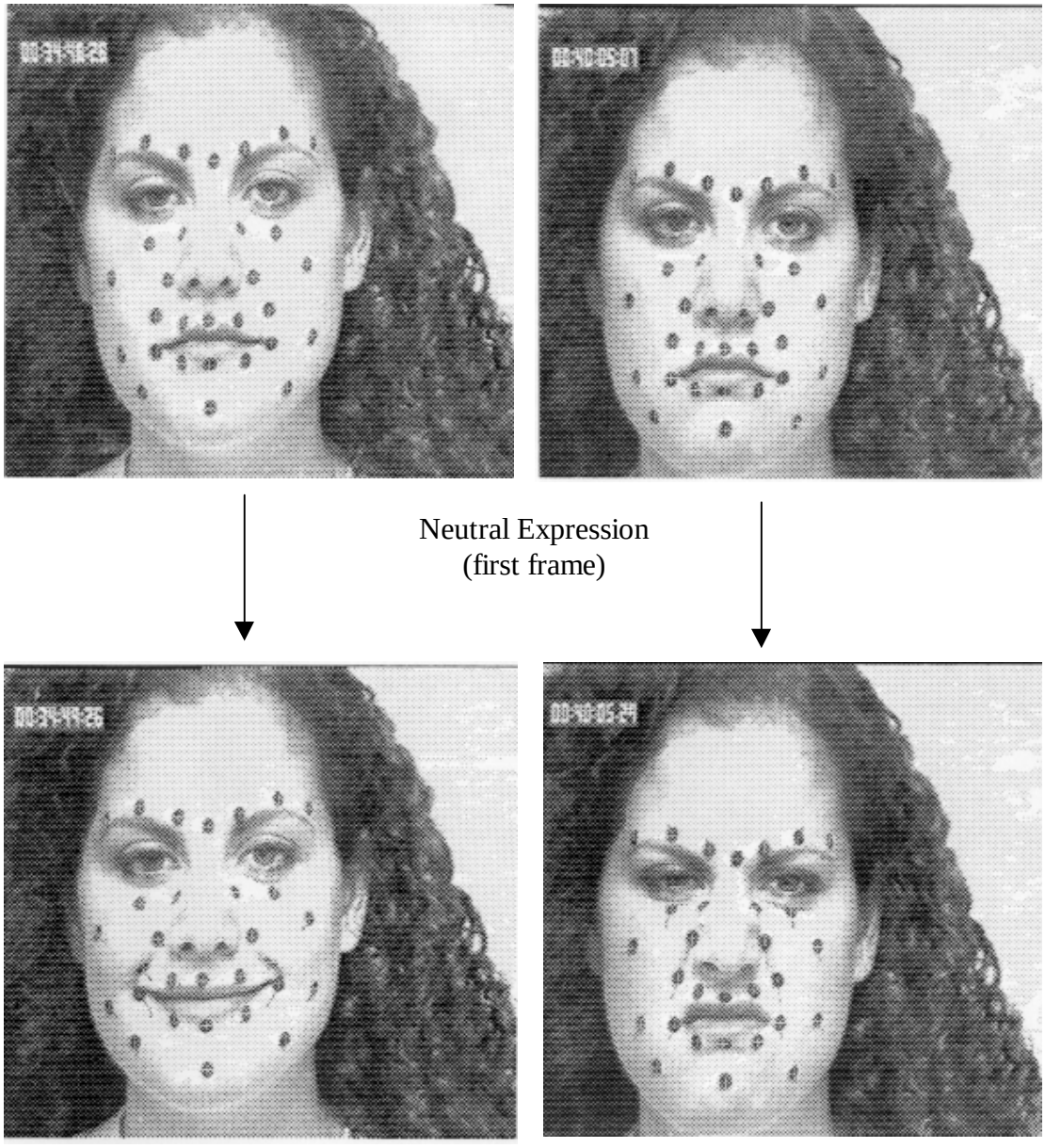


Figure 6 Dot tracking: each dot is marked by a cross (+) at its center, lines trailing from the dots represent changes in the location of dots due to facial expression.

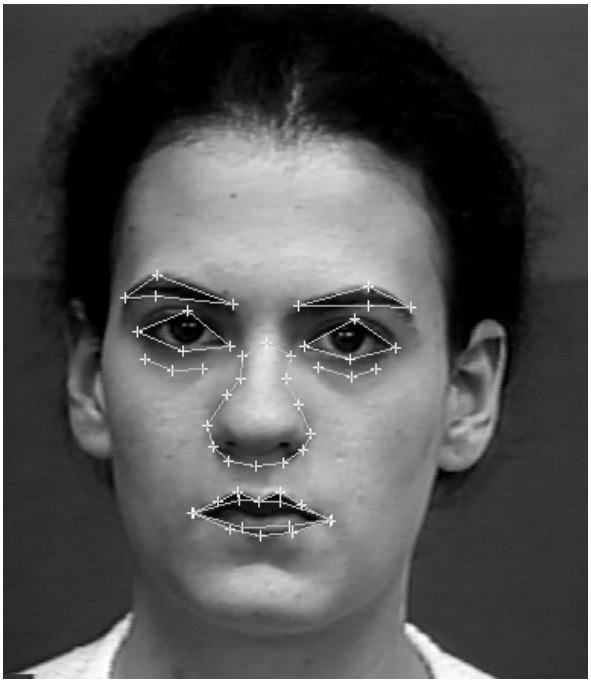


Figure 7 Locations of selected facial feature points (marked by a cross '+') which reflect the muscle motion of facial features.

results showed a very high correlation and the identical recognition performance between two operators [28]. This selection process can be easily taught to new operators. An operator can learn this after a training of about 5 minutes session, which is substantially shorter than learning the FACS (which may require 100 hours).

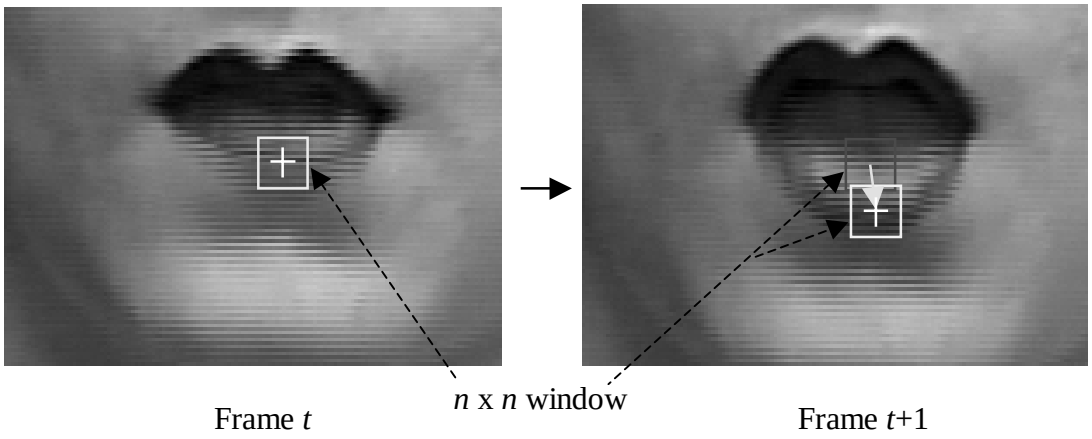


Figure 8 Feature point tracking based on tracking the movement of an *n* x *n* feature window between two consecutive frames (here, *n* is 13 pixels).

3.2 Motion Estimation and Flow Window

The motion estimation method used here is based on the optical flow algorithm developed by Lucas and Kanade [66], and implemented by including the pyramid approach used by Poelman and Kanade [77] to track large motion. This method assumes that the gray values in any image feature region (*n* x *n* feature window) do not change between two consecutive frames, but only shift from one position to another (Figure 8). We can track the motion of high gradient points, such as facial feature points, with subpixel accuracy by iterative computation. Its convergence is very fast.

Let us consider an *n* x *n* region *R* in the reference image at time *t*, where $I_t(x)$ denotes the gray value of the pixel position *x* in *R*. Let us find the best matching (registration) position of this region in the following frame at time *t*+1, where $I_{t+1}(x)$ denotes the gray value in the region, by minimizing a cost function *E* of the sum of squared differences (SSD) defined as

$$E(d(x)) = \sum_{x \in R} [I_t(x - d(x)) - I_{t+1}(x)]^2 w(x) \tag{3-5}$$

where $d(x)$ is the displacement of x of region R between two consecutive frames and $w(x)$ is a window function for weighting the squared differences in E. This minimization for finding the motion vector $d(x)$ can be done in iterations. Let

$$d(x) = d^i(x) + \Delta d(x) \tag{3-6}$$

at the ith iteration and $\Delta d(x)$ is the incremental displacement at the ith iteration. We want to robustly estimate the incremental displacement $\Delta d(x)$ with a subpixel accuracy. Let us expand the term

$$I_t(x - d) = I_t(x - (d^i + \Delta d)) \tag{3-7}$$

by the first order Taylor' expansion:

$$I_t(x - d^i - \Delta d) \approx I_t(x - d^i) - I_t'(x - d^i)^T \Delta d \tag{3-8}$$

where $I'_t(x)$ denotes the gradient of the gray value $I_t(x)$. The incremental change in the SSD cost function is given by

$$\begin{aligned} E(\Delta d) &= E(d^i + \Delta d) - E(d^i) \\ &\approx \sum_{x \in R} [I_t(x - d^i) - I_t'(x - d^i)^T \Delta d - I_{t+1}(x)]^2 w(x) - \sum_{x \in R} [I_t(x - d^i) - I_{t+1}(x)]^2 w(x) \\ &= \sum_{x \in R} [I_t'(x - d^i)^T \Delta d]^2 w(x) - 2 \sum_{x \in R} [I_t(x - d^i) - I_{t+1}(x)] I_t'(x - d^i)^T \Delta d w(x) \\ &= \Delta d^T G \Delta d - 2 e^T \Delta d \end{aligned} \tag{3-9}$$

where

$$G = \sum_x I_t'(x - d^i) I_t'(x - d^i)^T w(x) \tag{3-10}$$

is the Hessian matrix of the gradients of I_t with a window function $w(x)$, and

$$e^T = \sum_x [I_t(x - d^i) - I_{t+1}(x)] I_t'(x - d^i)^T w(x) \tag{3-11}$$

is a difference-gradient row vector which is the product of the difference (or error) between the regions in the two consecutive images and the gradient of the gray-value I_t

together with a window function $w(x)$. The maximum decrement $E(\Delta d)$ occurs when its gradient with respect to Δd is zero,

$$\frac{\partial E(\Delta d)}{\partial(\Delta d)} = G\Delta d + (\Delta d^T G)^T - 2e = 2(G\Delta d - e) = 0 \tag{3-12}$$

Hence,

$$\Delta d(x) = G^{-1}e \tag{3-13}$$

Initializing $d^{(0)}(x) = [0,0]^T$ and following equations (3-8), (3-10), (3-11) and (3-13), the optical flow $d(x)$ can be robustly estimated through iterations yielding the subpixel accuracy.

The motion estimate $d(x)$ is more accurate when the gradients of both $I_t(x)$ and $I_{t+1}(x)$ are large and nearly equal as illustrated in Figure 9 [66].

$$\begin{aligned} I_t'(x) - I_{t+1}'(x) &= I_{t+1}'(x+d) - I_{t+1}'(x) \\ &\approx [I_{t+1}'(x) + I_{t+1}''(x)d] - I_{t+1}'(x) \\ &= I_{t+1}''(x)^T d \end{aligned} \tag{3-14}$$

and $I_{t+1}''(x)$ denotes the second derivatives. The first order Taylor's linear approximation is more likely to give an accurate estimate d when both the difference of the gradients I_t' and I_{t+1}', and the second derivatives of I_{t+1} are small [66]. Otherwise, it is prove to have a large error. Thus the window function $w(x)$ should be small when the difference of the I_t' and I_{t+1}' is large, and large when the difference is small (Figure 9) [66]. If the window function is unity, $w(x) = 1$, over the n x n region R such as used in Lucas and Kanade's flow estimation [66], the local minimum of SSD is considered which maintains the high frequency information in the region but may yield a noisy result. This optical flow can accurately track the highly textured local region and the computation converges very fast. It is good for use in the facial feature point tracking and for real time processing. But it will be less accurate when used to track a less textured region or a region with high reflection where there is no high gradient pixels for tracking, *i.e.*, the region is not

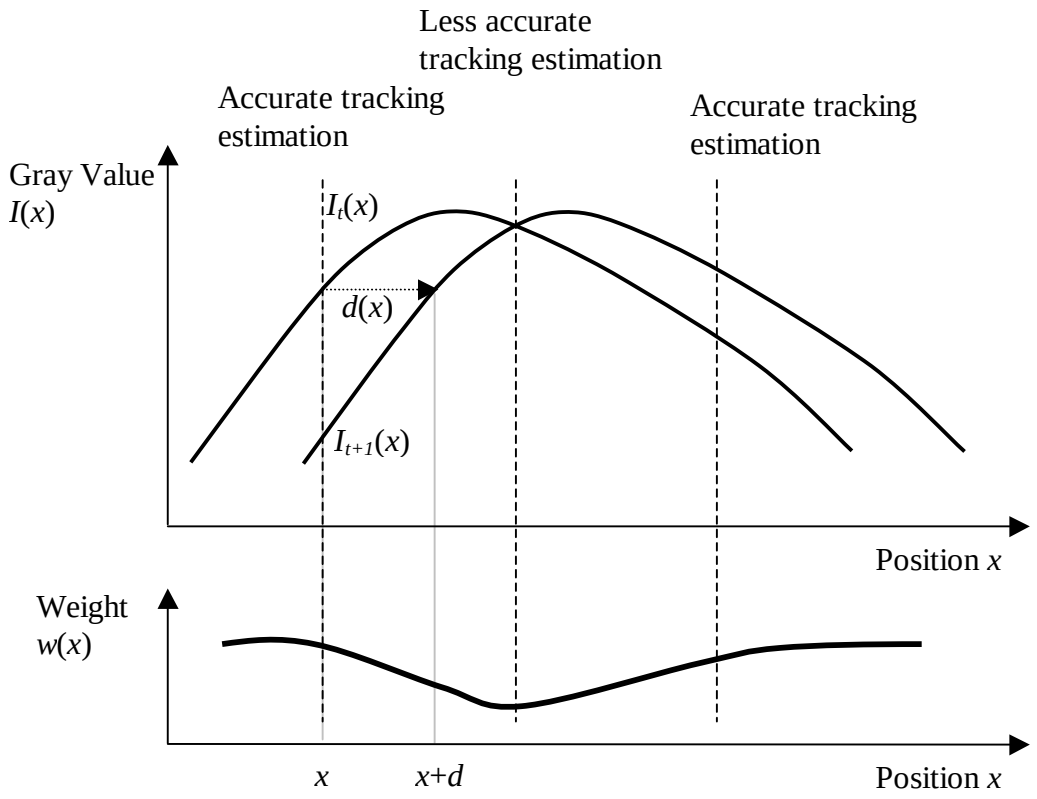
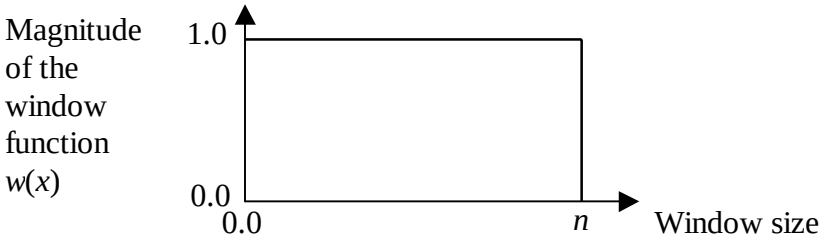
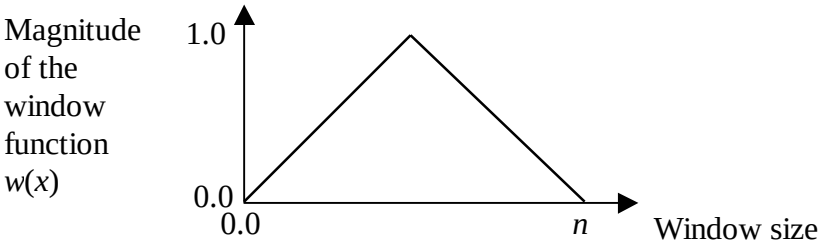


Figure 9 Window (weight) function $w(x)$ can be used to control the accuracy of motion estimation based on the gradient varying from point to point [66].

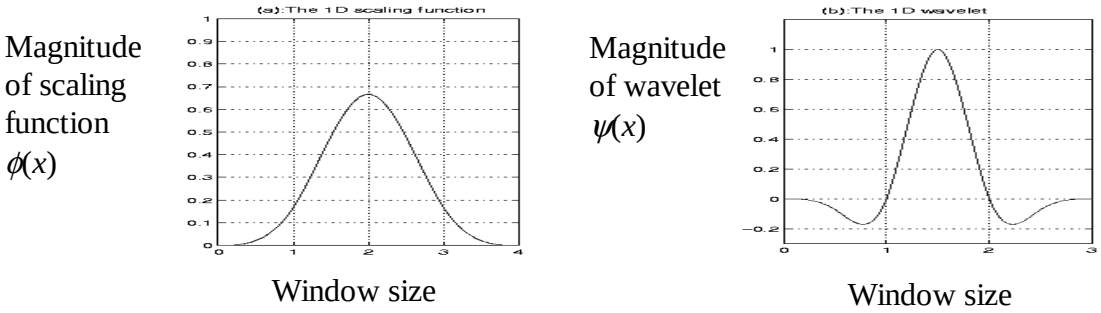
trackable. In such a case, more global information instead of local minimization may be needed, such as using the regularization-based or global smoothness approach [11,47] to estimate the optical flow in textureless region, but it is more time consuming since a large window is involved and it may also smooth out high frequency components and thus reduce the tracking accuracy. Two methods have been recently developed to overcome these local noise and global smoothness problems: the spline base method [90] and the wavelet base method [101], both use the pyramid approach and multiple flow windows (Figure 10).



Window function $w(x)$ used in the Lucas-Kanade's method [66].



Window function $w(x)$ used in Szeliski's spline-based method [90].



Low-pass scaling function $\phi(x)$ and high-pass wavelet $\psi(x)$ used in the wavelet-based method of Wu [101].

Figure 10 Comparison of Lucas-Kanade, spline-based and wavelet-based window functions.

3.3 Motion Confidence Estimation

Tomasi and Kanade [92]; and Poelman and Kanade [77] have shown that the eigenvalues of the Hessian matrix G can be used to estimate the confidence of whether the n x n feature region R is trackable or not. We illustrate it by four features regions shown in Figure 11 with window function $w(x) = 1$ in each region, thus

$$G = \begin{bmatrix} G_{11} & G_{12} \\ G_{12} & G_{22} \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \left(\frac{\partial I_t}{\partial x_1} \right)^2 \Big|_{x - d^i} & \sum_{x \in R} \left(\frac{\partial I_t}{\partial x_1} \right) \left(\frac{\partial I_t}{\partial x_2} \right) \Big|_{x - d^i} \\ \sum_{x \in R} \left(\frac{\partial I_t}{\partial x_1} \right) \left(\frac{\partial I_t}{\partial x_2} \right) \Big|_{x - d^i} & \sum_{x \in R} \left(\frac{\partial I_t}{\partial x_2} \right)^2 \Big|_{x - d^i} \end{bmatrix} \tag{3-15}$$

$$e = \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \sum_{x \in R} \left(\frac{\partial I_t}{\partial x_1} \right) \Big|_{x - d^i} [I_t(x - d^i) - I_{t+1}(x)] \\ \sum_{x \in R} \left(\frac{\partial I_t}{\partial x_2} \right) \Big|_{x - d^i} [I_t(x - d^i) - I_{t+1}(x)] \end{bmatrix} \tag{3-16}$$

In Figure 11.a, the feature region R is textureless or very smooth. This region will be difficult to track since it has zero gradient in all directions and, hence, both eigenvalues of the Hessian matrix G are equal to zero. If a feature region R contains a line or edge which has high spatial gradient values in at least one direction as shown in Figure 11.b or 11.c, or which has highly correlated spatial gradients in both horizontal and vertical directions as in Figure 11.c (the Hessian matrix G has one large eigenvalue and one small eigenvalue), then this region will be difficult to track. Only when the feature region R has high spatial gradients in two orthogonal directions (horizontal- and vertical-gradients are weakly correlated), and hence both eigenvalues of G are large, can this feature region be localized and easily tracked as shown in Figure 11.d. From the mathematical point of view, if the Hessian matrix G has one or more small eigenvalues, then computation of its inverse is an ill-conditioned problem because it is close to be singular.

To estimate the confidence of whether a selected n x n feature region R is trackable or not, we use the confidence value defined below.

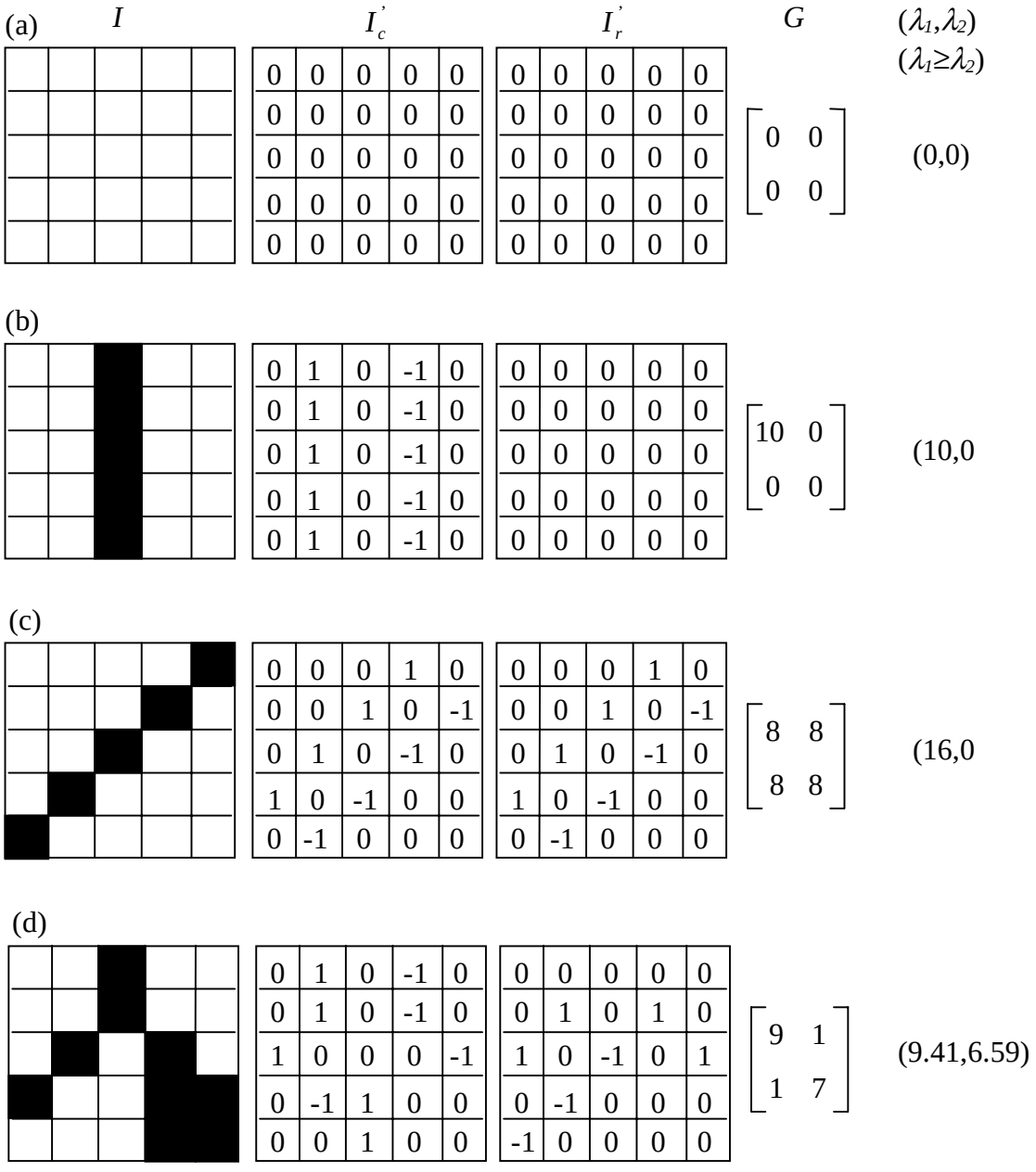


Figure 11 Trackability of various feature regions in a binary image: (a) contains no image texture so it would make a poor feature; (b) and (c) have high gradients locally either in one direction, or the horizontal- (I_c'-) and vertical- (I_r'-) gradients are highly correlated with each other as in (c), they also are not trackable; only feature (d) can be used as a trackable feature [77].

$$C = \frac{\lambda_{\min} * (n * n)}{1 + \sum_{x \in R} [I_t(x - d) - I_{t+1}(x)]^2} \tag{3-17}$$

Here, λ_{min} is the minimum eigenvalue of the Hessian matrix G. The constant 1 is used to avoid a zero value in the denominator in case of a perfect matching. The trackability is proportional to the confidence value, (or is proportional to λ_{min}), and inversely proportional to the difference (or residue) between two matching regions. A feature region with high λ_{min} contains high-frequency textured patterns and can be localized accurately even though there is noise in the image. A feature region with a very low value of λ_{min} contains smooth areas, so image noise is more likely to cause shifts along the lower gradient direction, such as along the line or edge in Figure 11.b and 11.c.

3.4 Tracking Subpixel and Large Motion

Since we want to discriminate subtly different facial expressions by extracting the movement of feature points, it is necessary to use optical flow to accurately track motions of feature points in subpixel accuracy. Initially a 5 x 5 Gaussian filter is used to smooth out the noise in order to enhance the flow computation convergence. Selecting the size of the feature region is an important trade-off. It should be large enough to include sufficient texture in it, while small enough so that the computation of the inverse Hessian matrix G will not become ill-conditional. Also, a larger region will require more computation in order to perform feature tracking. We choose 13 x 13 pixels to be the feature region. That is, each selected feature point in the first frame of each image sequence (image size 490 x 640 pixels but cropped to 417 x 385 pixels) is the center of a 13 x 13 flow region. A window function $w(x)$ = 1 over the feature region is chosen because the area surrounding each feature point is full of texture. The movement of facial feature points is then automatically tracked with subpixel accuracy in translation [3,102] via optical flow in the remaining frames of the image sequence.

The selected feature point location x is integer-valued, but $x+d$ is generally not integer-valued in order to accommodate for subpixel flows. We estimate the image gray value at an non-integer-valued pixel by using the bilinear interpolation,

$$\begin{aligned} I(x+d) &= I(x_1, x_2) \\ &= I(i_1, i_2) * (i_1 + 1 - x_1, i_2 + 1 - x_2) + I(i_1, i_2 + 1) * (i_1 + 1 - x_1, x_2 - i_2) + \\ &\quad I(i_1 + 1, i_2) * (x_1 - i_1, i_2 + 1 - x_2) + I(i_1+1, i_2 + 1) * (x_1 - i_1, x_2 - i_2) \end{aligned} \quad (3\text{-}18)$$

where $i_1 = \lfloor x_1 \rfloor$ and $i_2 = \lfloor x_2 \rfloor$, and $\lfloor x \rfloor$ represents the largest integer smaller than or equal to x. Since the movement of feature points will be tracked for an entire sequence in subpixel accuracy, the ending position of each tracked feature point in the first pair of frames (I_t,I_{t+1}) will be used as the starting position of the tracked point for the next pair of frames (I_{t+1},I_{t+2}), and so on. The bilinear interpolation method is applied to interpolate gray values of the non-integer-valued pixels of both the starting and ending positions for each tracked feature point in each consecutive pair of frames in the sequence.

Consecutive frames of an image sequence may contain large feature-point motion caused by gross movement of the subject between frames, such as sudden head movements, brow raised or mouth opening of the surprise expression, which may cause missing or lost tracking (Figure 12). In Figure 12, lines trailing along feature points denote their movements across image frames in the sequence. In order to recover these large motions without losing subpixel accuracy, we use a pyramid method with reduced resolution (spatial smoothing) [(77)]. Each image is decomposed into 5 levels from level 0 (the original finest resolution image) to level 4 (the coarsest resolution image). The image sizes are 490 x 640 (row x column), 125 x 160, 62 x 80, 31 x 40, and 15 x 20 pixels, respectively (Figure 13). We use 13 x 13-pixel window regions for each level. From level 4 to level 1 (from coarse to fine levels), we consider window-wise 1, 4, 16 and 64 flow regions for the whole image at each level, respectively. Each window center in the first frame is used as the starting position for motion estimation at that level. From level 1 to level 0, we only consider those 13 x 13 feature regions whose centroids are the locations of the previously tracked feature points. Flow computation proceeds from the

Figure 12 Feature point tracking excluding the pyramid method: it is sensitive to subtle motion such as eye blinking, but it loses tracking for large motion such as mouth opening and suddenly raising eye brows. Lines trailing along feature points (marked by a cross '+') denote their movements across image frames in the sequence.

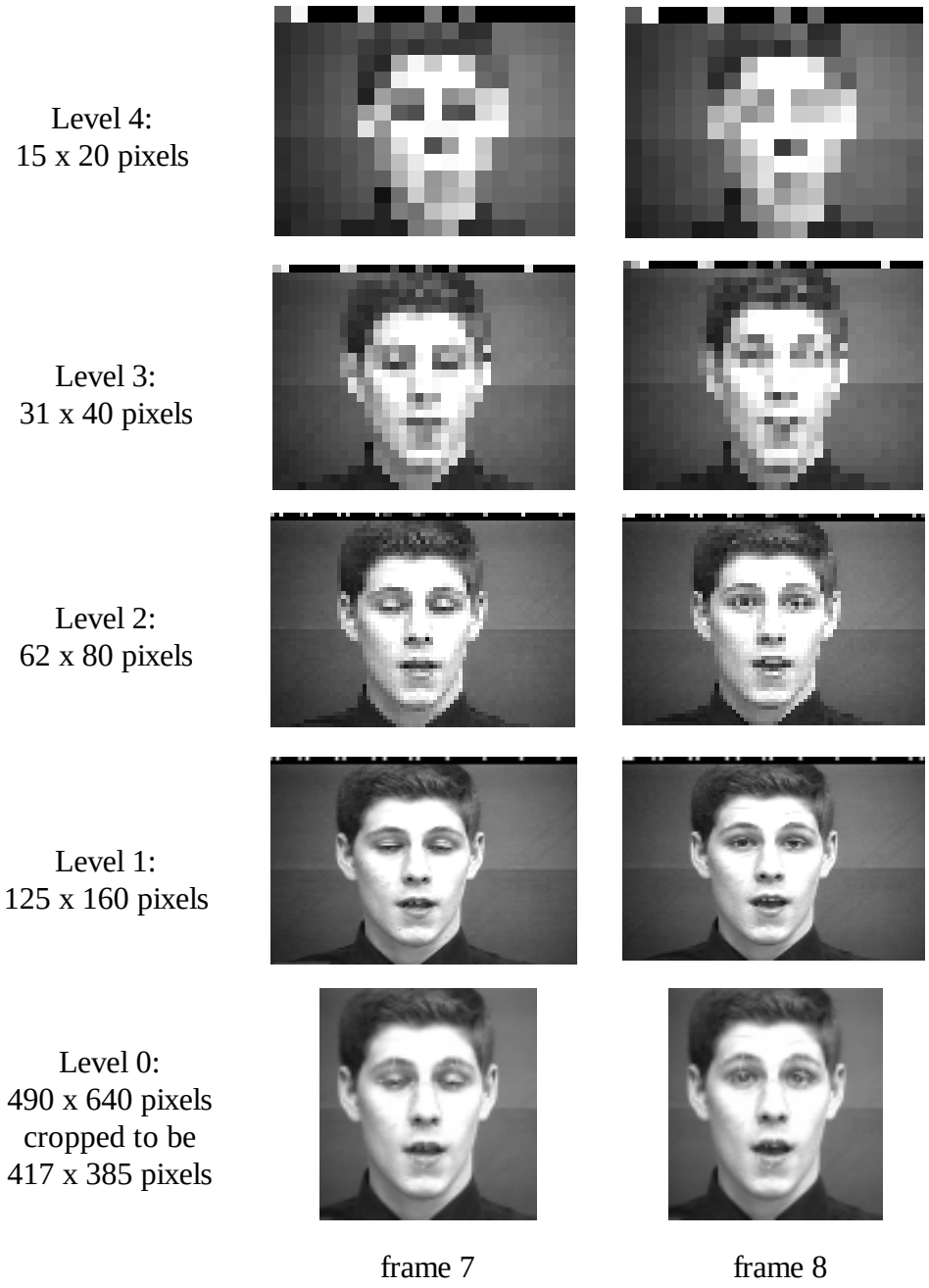


Figure 13 A 5-level pyramid for feature point tracking.

lowest resolution level (level 4) to the highest level (level 0) of the pyramid.

For the iterative computation at level l, we use the estimated motion vector d_{l+1} and confidence value C_{l+1} obtained at the coarser level l+1. At the coarsest resolution level 4, the motion vector is initialized at $(0,0)^T$ and iterated to obtain a solution d_4; confidence value C_4 is then computed. There are four feature regions at level 3, $2*d_4$ is taken as the initial motion vector of each region center and iteration proceeds to obtain a motion vector $\tilde{d}_3$ in each region with confidence value $\tilde{C}_3$. $\tilde{d}_3$ and d_4 are weighted as described below to give a new estimate d_3; so is a new estimate of C_3 obtained by weighting $\tilde{C}_3$ and C_4. In general,

$$d_l = \frac{C_{l+1} * (2 * d_{l+1}) * K + \tilde{C}_l * d_l * (1-K)}{C_{l+1} * K + \tilde{C}_l * (1-K)} \tag{3-19}$$

$$C_l = \frac{C_{l+1} * C_{l+1} * K + \tilde{C}_l * \tilde{C}_l * (1-K)}{C_{l+1} * K + \tilde{C}_l * (1-K)} \tag{3-20}$$

where $0.0 \le K \le 1.0$ *and* $1 \le l \le 3$

where K is a constant weighting factor that determines how much confidence is given to that obtained from the coarser level in the pyramid. $2*d_l$ will be used as the initial motion vector and C_l as new confidence value for iterative estimation at the next finer resolution level l-1. If K = 1.0, all flow estimates at the current level (l) are derived from the previous level (l+1) without regard to the flow estimate computed at the current level. If K = 0.0, it provokes each level's computation to use the previous level's flow estimate only as an initial value. This inter-level confidence base is used for combining the effect of spatial smoothing. In order to have a reliable flow estimation when the tracking region contains insufficient texture, the flow at the high-resolution level is mainly inherited from the flow in the previous coarse level of the pyramid, that is, K is close to but not equal to 1.0. If the tracking region has high texture, then it can be reliably tracked so as to be less involved with the flow estimated at the previous level, that is, K is close to but not equal to 0.0. We choose K=0.5 for confidence estimation. From level 1 to level 0, we first identify the locations at level 1 corresponding to the feature points at level 0, which in

general will not coincide with the region centers at level 1. At each of these locations, its motion vector will be estimated by the bilinear transformation from its four neighboring centers. This estimated vector multiplied by 4 will be used as the initial motion vector at the corresponding feature point for iterative estimation at level 0 to obtain the estimate d_0 in the feature point tracking. Then the process repeats for the next consecutive point of frames.

Using this pyramid method for optical flow computation, we initially enable the gradient descent method at low resolution levels to avoid local minima in the search for the optimal solution of feature point displacement. This allows us to recover any large motion (up to 100-pixel displacement) of the feature point while maintaining its sensitivity to subtle (subpixel) facial motion (as shown in Figure 14), and the flow computation converges quickly (less than 20 seconds for tracking 70 feature points between two consecutive frames using the interface under SUN Sparc 5). Point tracking method deals very well with large feature point movement between two 490 x 640-pixel frames.

3.5 Analysis of Feature Point Tracking Problems

It has been noted in our experiments that an error may occurred when some facial feature points located at the edge of the brows or mouth had large movements between two consecutive frames. Those selected feature points were tracked along the edge direction of the brows or mouth (Figure 15). This is due to the fact that the tracking was sensitive along the low gradient direction when this region contains a high gradient line.

There are three ways to correct these errors. One way is to locate those feature points away from edges instead of along the edges. The error is reduced for brows raised but still occurs along edges when mouth opens larger (Figure 16), because the mouth motion causes more facial deformation than that by brow motion. Another solution is to have a larger n x n feature region R to include more motion information (Figure 17), but it requires more computation time for tracking. Still another method, which we use, is to

increase the value of the weighting factor *K* of the inter-level confidence base so as to include more global information from the previous low-resolution level processing in the pyramid method (Figure 18).

3.6 Data Quantization and Conversion for the Recognition System

Three upper facial expressions are to be recognized based on displacements of 6 feature points at the upper boundaries of both brows, and six lower face expressions are to be recognized based on displacements of 10 feature points around the mouth. Feature points are numbered from left to right for brows region, and from the left corner point of lip clockwise around the mouth. The displacement of each feature point is calculated by subtracting its normalized position in the first frame from its current normalized position. Each feature point has the horizontal displacement component and vertical displacement component. The displacement vector is 12-dimensional in the upper face and 20-dimensional in the lower face (Figure 19). Facial expressions are characterized by these two vector sequences. These displacement vectors in upper and lower facial regions are vector-quantized separately into 16 and 32 symbols, respectively, as discussed in section 6.1 and section 8.3.1. Table 4 shows sample symbol sequences for nine facial expressions under consideration. Such symbol sequences are used as inputs to the HMMs of upper facial expressions and lower facial expressions, respectively, for automatic recognition.

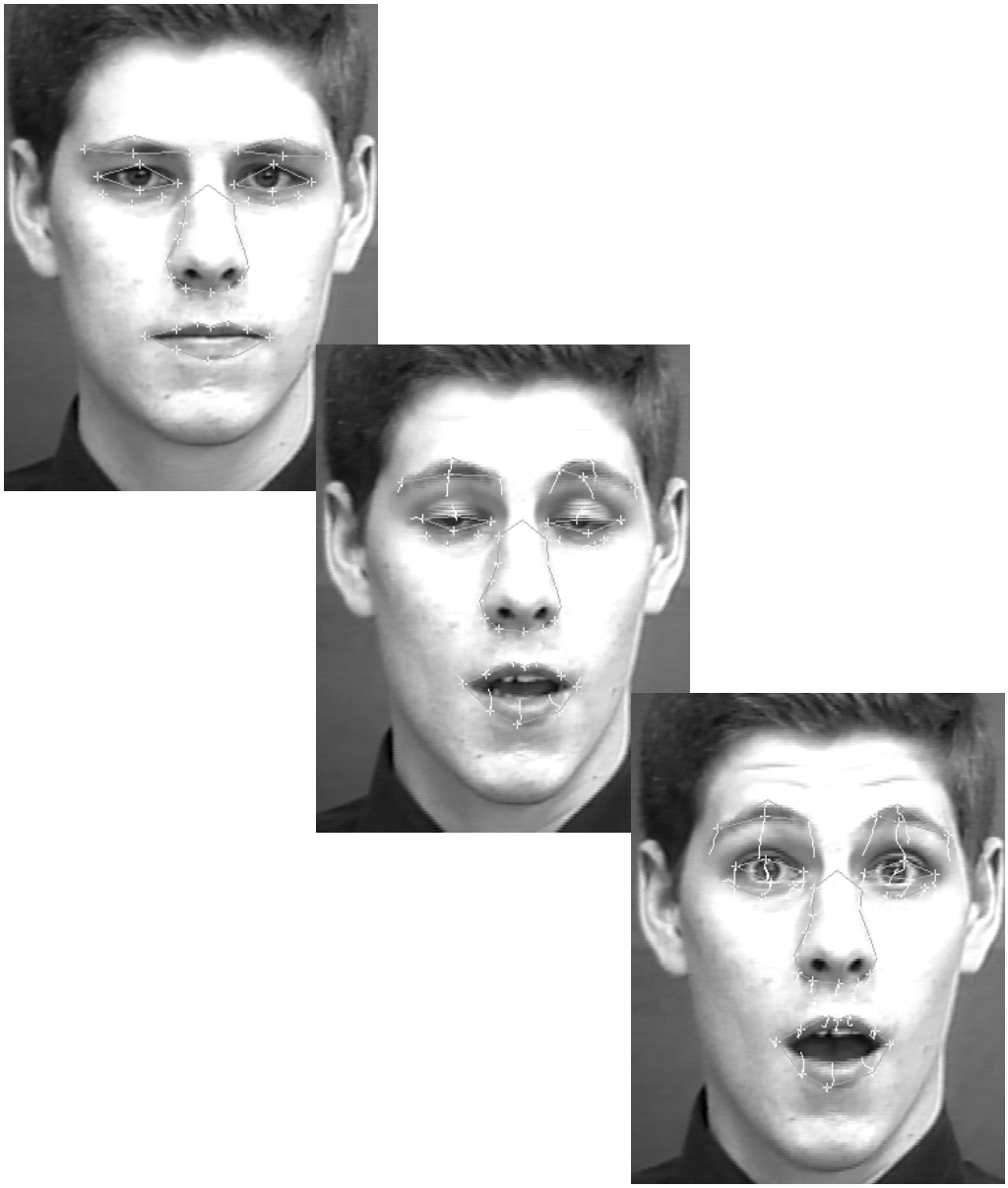


Figure 14 Feature point tracking including the pyramid method: it is sensitive to subtle motion such as eye blinking and also tracks accurately for large motion such as mouth opening and suddenly raising eye brows.

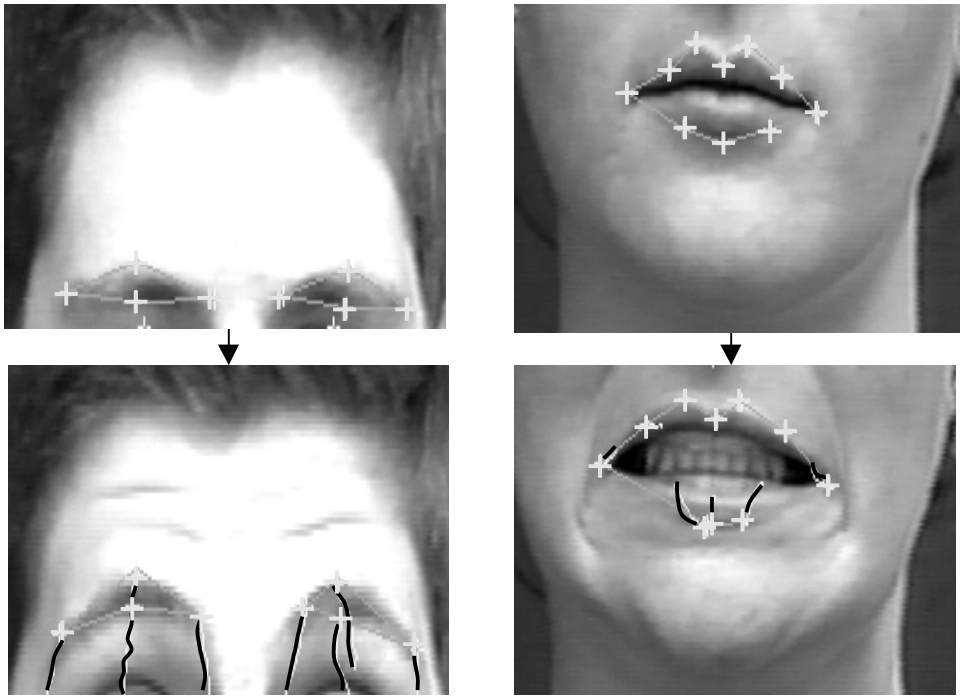


Figure 15 The feature point tracking error due to violation of the feature region's trackability condition: tracking along the edge direction at both brows and mouth regions with deformed shapes.

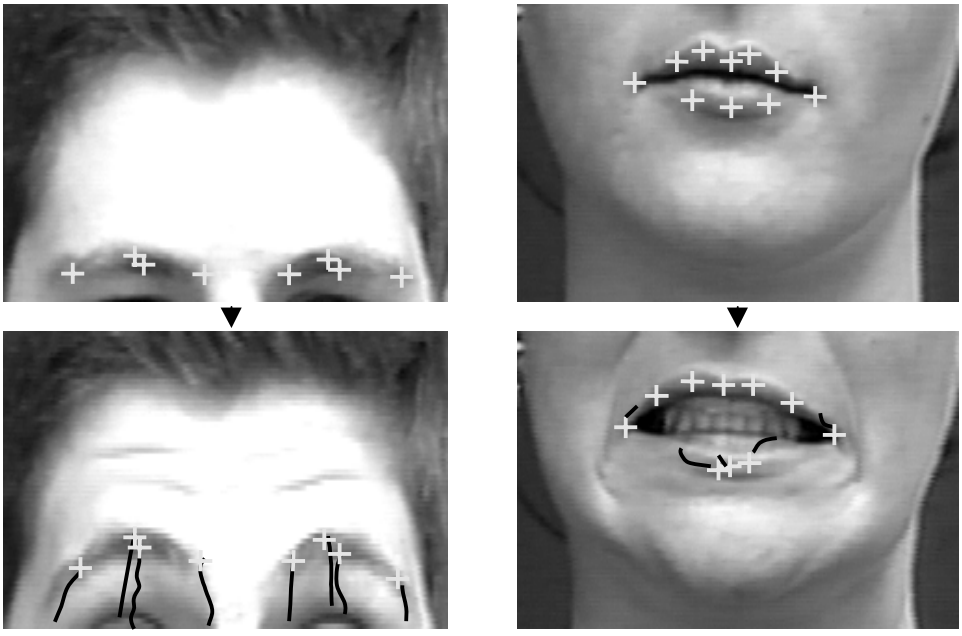


Figure 16 Reducing the tracking error by locating feature points away from edges of facial features: the tracking for brow region is improved, but is still erroneous in mouth region with large mouth opening.

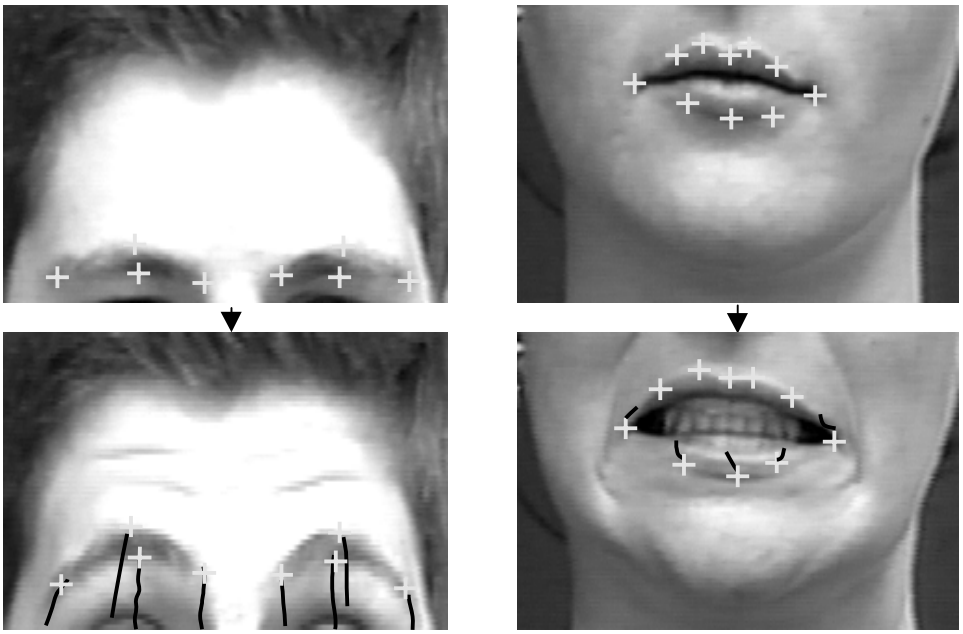


Figure 17 Reducing the tracking error by using a large window size which requires more processing time.

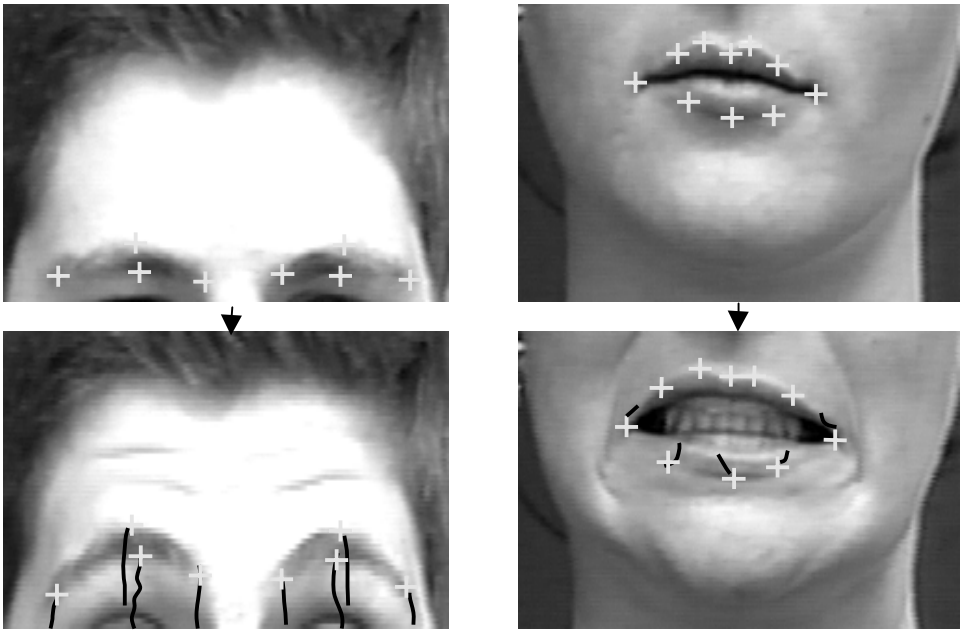
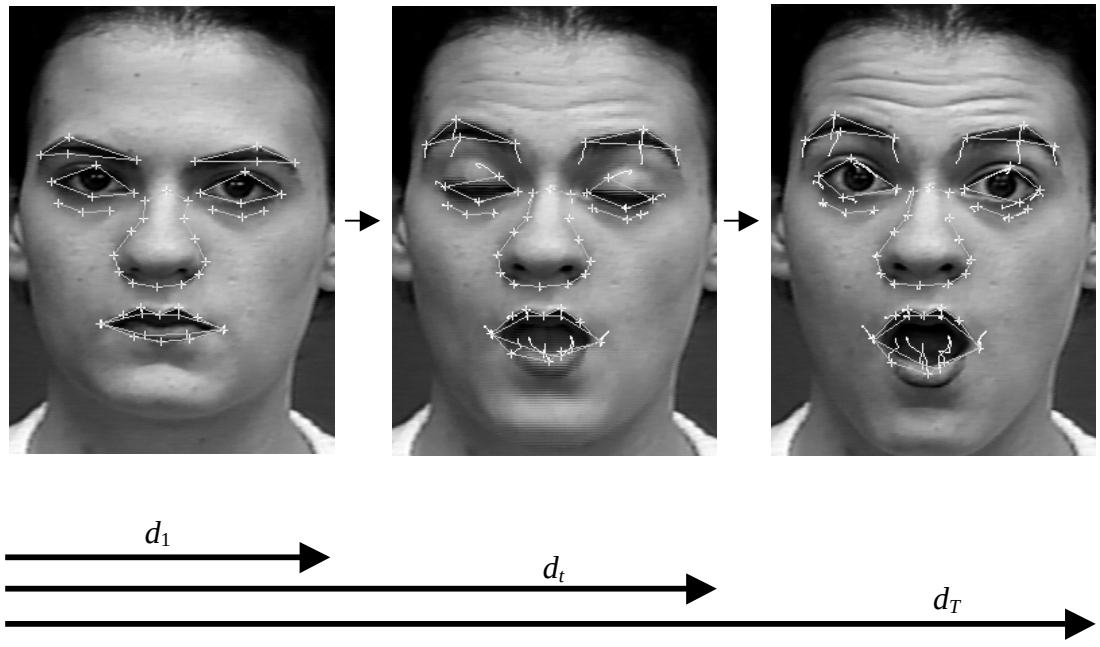


Figure 18 Reducing the tracking error by increasing the value of the weighting factor K (K=0.5) of the inter-level confidence base in order to include more global information from the previous low-resolution level processing in the pyramid method.



Displacement vector $d_t = (d_{t,1},d_{t,2},\ldots, d_{t,j},\ldots,d_{t,i})$
where $d_{t,j} = (d_{t,j,horizontal}, d_{t,j,vetrical})$ is the pair of horizontal and vertical displacements for feature point j at frame t,
$i = 6$ for brow region (upper facial expression)
$i = 10$ for mouth region (lower facial expression)

and feature points are numbered from left to right for brow region, and from the left corner point of lip clockwise around the mouth.

Displacement vector sequence $D = (d_1,d_2,\ldots,d_t,\ldots,d_T)$ for each of the upper and lower facial expressions where T is the length of an image sequence.

Figure 19 Displacement vector of the facial feature point tracking to be encoded for input to a Hidden Markov Model.

Table 4 Sample symbol sequences for three upper facial expressions and six lower facial expressions under consideration.

AUs	**Feature Point Tracking: Upper Facial Expressions (Symbol Sequence)**
4	3 3 3 3 *5* 5 5 5 5 *5* 5 5
1+4	3 3 3 3 *3* 3 3 1 1 *1* 1 1 1 1 *1* 1 1
1+2	3 3 3 3 *3* 6 6 6 6 *6* 6 6
AUs	**Feature Point Tracking: Lower Facial Expressions (Symbol Sequence)**
12	2 2 2 2 *2* 2 2 9 1 *1* 1
6+12+25	2 2 2 2 *10* 5 5 5 11 *11* 11 11 11 11 *11*
20+25	2 10 10 10 *13* 13 13 13 13 *13* 13
9+17	2 2 3 3 *3* 14 14 14 14 *14*
15+17	2 2 2 2 *6* 6 6 6 6 *6* 6 6 6
17+23+24	2 2 2 2 *4* 4 4 4 4

d	0	1	3	5	7	10	14	20
η	0.00	0.14	0.39	0.58	0.71	0.82	0.89	0.94

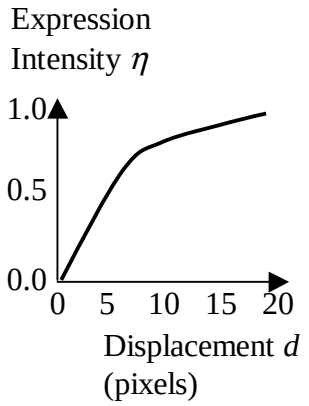


Figure 20 An example illustrating the expression intensity estimation by following the non-linear mapping (k = 7) of the constrained feature point displacement for the case of AU2.

3.7 Expression Intensity Estimation

Since displacements of feature points correspond to AU intensities of facial expressions and indicate the underlying muscle motion, we can quantify the expression intensity based on the displacements of feature points. Figure 20 and Table 5 show the logic of expression intensity estimation and the specified constraints for displacement measurement of each feature point which corresponds to the expression intensity of individual AU. We propose to estimate the expression intensity η by using following non-linear mapping of the feature point displacement

$$\eta = \frac{d}{\sqrt{d^2 + k^2}} \qquad \textit{where } 0 \le \eta \le 1.0 \tag{3-21}$$

where d is the measured normalized displacement of a feature point under constraint for a AU as specified in Table 5. Since the motion of a feature point during a facial expression is described, in general, initial as gradual motion (starting from the neutral expression) followed by quick motion then gradually slowing down until the peak expression is reached, the value of constant k is determined empirically according to individual AU

Table 5 The dictionary for expression intensity estimation (image: 417 x 385 pixels).

Action Unit	**Constraint: direction** 	**Measured Displacement *d***	***k*** **(Pixel)**
	Brows		
1	Inner point of brow moves vertical up	Vertical displacement	7
2	Outer point of brow moves vertical up	Vertical displacement	7
4	1. Inner point of brow moves vertically down 2. Inner points of both brows move horizontally toward one another	1.Vertical displacement 2.Horizontal distance between both points	4
	Eyes		
5	Middle point of upper eyelid moves up	Vertical displacement	2
7	Middle point of lower eyelid moves up	Vertical displacement	2
41~46	Middle points of eyelids move vertically together (narrow eyes or blinking)	Vertical displacement between both points	4
	Nose		
9	Side point at nostril moves up	Vertical displacement	5
	Mouth		
12	1. Left corner point of lip moves up between 100 and 170 degrees 2. Right corner point of lip moves up between 10 and 80 degrees	1. Euclidean distance 2. Euclidean distance	6
20	1. Left corner point of lip moves horizontally between 170~180 and -170~-180 degrees 2. Right corner point of lip moves horizontally between 0~10 and 0~-10 degrees	1. Euclidean distance 2. Euclidean distance	6
15	1. Left corner point of lip moves down between -90 and -170 degrees 2. Right corner point of lip moves down between -10 and -90 degrees	1. Euclidean distance 2. Euclidean distance	3
18	Both corner points of lip move horizontally toward one another	Horizontal distance between both lip corners	8
23	Two points on upper lip move horizontally together	Horizontal distance between both points	2
24	Both center points on lips move vertically together	Vertical distance between both points	5
25~27	Both center points on lips move vertically away from baseline	Vertical distance between both points	20

Table 5 (Continued)
The dictionary for expression intensity estimation (image: 417 x 385 pixels).

Action Unit	**Constraint: direction** 180^0 -180^0 0^0	**Measured Displacement *d***	***k*** **(Pixel)**
	Mouth		
17	Existence of furrow or wrinkle on the chin	Not measured	-

(listed in Table 5) to give a genuine expression intensity time course fit to the velocity of the facial motion (like the oscillation of a spring between compression and release) as shown in Figure 21.

The expression intensity estimation given above is one first attempt to quantify AU expression measurement. In reality, it is difficult to measure the expression of an individual AU by tracking a single feature point which indicates a single muscle movement. For example, both AU15 and AU20 involve downward motions of feature points at lip corners. Although the FACS system assumes that there is a one-to-one mapping between an AU and a single muscle motion, the expression intensity of an individual AU may be composed of the coordinated movements of multiple muscles or movements of multiple feature points. It would be desirable to consider a set of feature points corresponding to an "expression unit," and measure their simultaneous motion for quantifying the expression intensity. This will be a challenging aspect for future research.

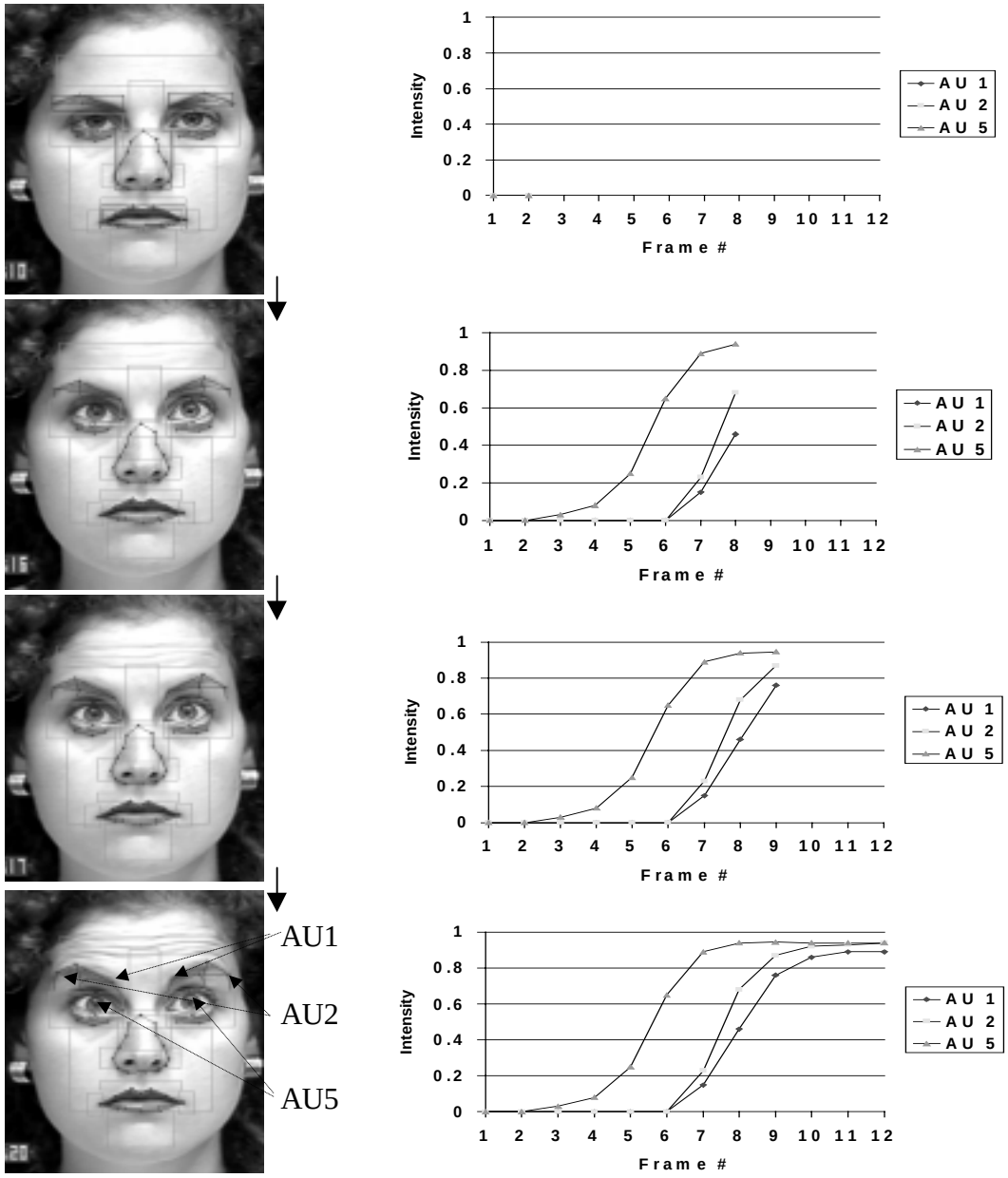


Figure 21 Expression intensity time course of AU1, 2 and 5 fit to the displacement changes of facial feature points based on the non-linear mapping [107].