4.0 DENSE FLOW TRACKING AND EIGENFLOW COMPUTATION

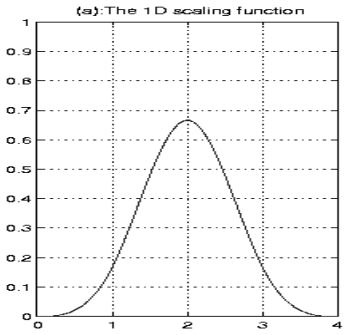
Feature point tracking of eyebrows and mouth presented in Chapter 3 is sensitive to subtle feature motions and is capable to track large displacements. The forehead, cheek and chin regions also contribute important facial expression information. Since the actions of individual facial muscles are interdependent, and the activation of fibers in one muscle may influence movements of adjacent muscles. Single facial expression may be the result of movements and deformation of not only facial features but also facial skins which are caused by facial muscle actions triggered by nerve impulses. To enhance the realism of an automatic recognition system, it is desirable to capture more detailed motion information. This leads to the consideration dense flow which describes the motion of each pixel on the entire face image.

4.1 Wavelet-Based Motion Estimation

Wu's approach of dense flow estimation [101] is employed to estimate the entire facial motion. The flow window functions are based on scaling function and wavelet of Cai and Wang [18] (Figure 22.a), which provide wavelet coefficients from the coarse-to-fine resolution levels. This approach estimates image motion in large regions with large window support at the coarse resolution level and motions in small regions with small window support at the fine resolution level. It is different from the traditional coarse-to-fine pyramid method by taking into account the detail information decomposing image resolutions at various resolution levels. Both coarser and finer level motions can be simultaneously estimated to correct the error produced by the previous motion estimation at the coarse level. It will yield more accurate motion estimation.

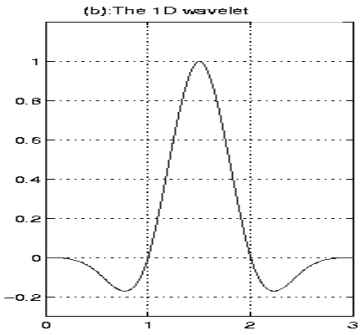
Let us consider the one-dimensional case first. The scaling function ϕ is the fourth-

Scaling function $\phi(x)$



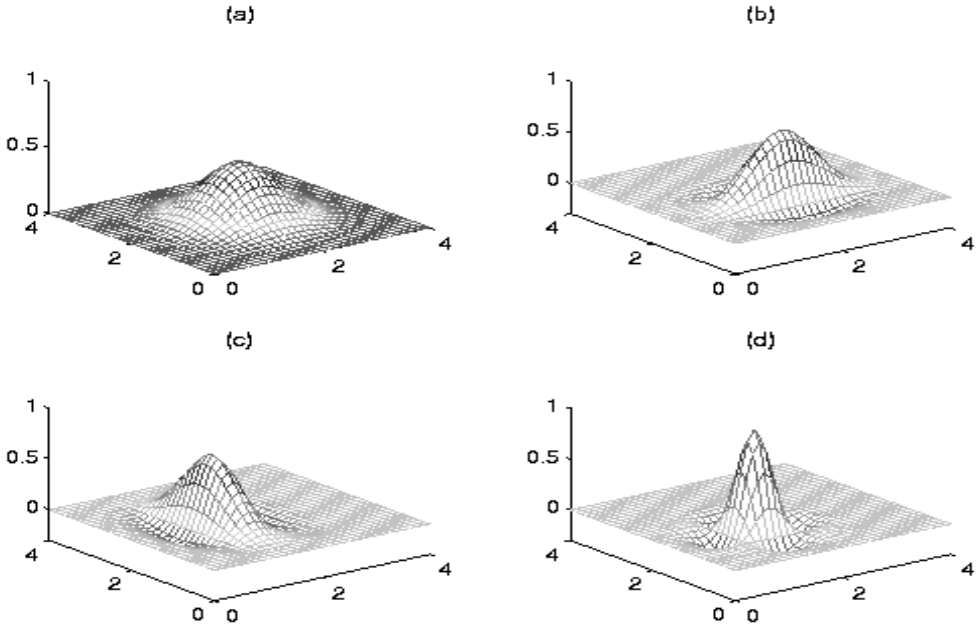
x

Wavelet $\psi(x)$



x

1-dimensional scaling function and wavelet.



2-dimensional scaling function and wavelets: (a) $\phi(x,y) = \phi(x)\ \phi(y)$,
(b) $\psi^H(x,y) = \phi(x)\ \psi(y)$, (c) $\psi^V(x,y) = \psi(x)\ \phi(y)$, and (d) $\psi^D(x,y) = \psi(x)\ \psi(y)$.

Figure 22.a 1- and 2-dimensional scaling functions and wavelets [101].

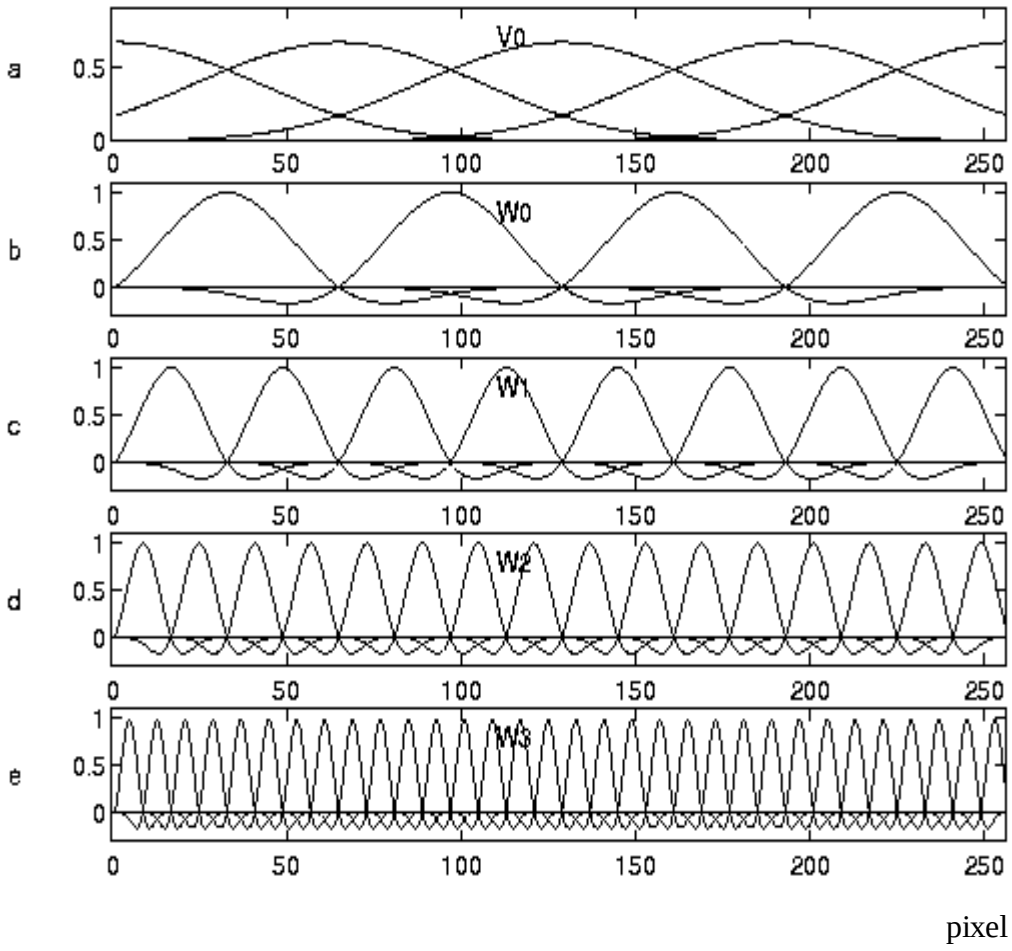


Figure 22.b Dilation and translation of 1-dimensional basis (scaling and wavelet) functions [101].

order (L = 4) B-spline function acting as a low-pass filter for smoothing the signals, this is represented by

$$\phi(x) = \frac{1}{6}\sum_{j=0}^{L}\binom{4}{j}(-1)^j (x-j)^3 \qquad \in H^2(I) \qquad \text{where } I = [0,L=4] \qquad (4\text{-}1)$$

where I denotes a finite interval and $H^2(I)$ represents the Sobolev space containing all continuous functions with the finite energy norm up to the second derivative. The wavelet $\psi(x)$ acting as a high-pass filter is derived from the scaling function through a 2-scale equation.

$$\psi(x) = \frac{-3}{7}\phi(2x) + \frac{12}{7}\phi(2x-1) - \frac{-3}{7}\phi(2x-2) \qquad (4\text{-}2)$$

The interval for $\psi(x)$ is [0,3]. The dilation 2^j and translation k of $\phi(x)$ and $\psi(x)$ are given by (Figure 22.a for j=0 and k=0)

$$\phi_{j,k}(x) = \phi(2^j x - k) \qquad j \geq 0, \quad k = \text{-2,..., } 2^j L\text{-2} \qquad (4\text{-}3)$$

$$\psi_{j,k}(x) = \psi(2^j x - k) \qquad j \geq 0, \quad k = \text{-1,..., } 2^j L\text{-2} \qquad (4\text{-}4)$$

A finite energy continuous function $d(x)$, which represents the motion at position x, can be decomposed into a scaling component $d_{-1}(x)$ at the coarsest resolution level -1 and many wavelet components $d_j(x)$'s at resolution levels $j \geq 0$,

$$d(x) \approx d_{-1}(x) + d_0(x) + d_1(x) + \ldots + d_j(x) \qquad (4\text{-}5)$$

where

$$d_{-1}(x) = \sum_{k=-2}^{L-2} c_{-1,k}\phi_{0,k}(x) \qquad (4\text{-}6)$$

$$d_j(x) = \sum_{k=-1}^{2^j L-2} c_{j,k}\psi_{j,k}(x) \qquad j \geq 0 \qquad (4\text{-}7)$$

In the motion estimation problem, $d(x)$ is unknown but satisfies the optical flow equation. Its solution is to be computed by the above multi-resolution approximation (Figure 22.b). By repeatedly estimating [101] the coefficients $c_{j,k}$, each component $d_j(x)$ is constructed from the linear combination of translated basis functions at level j.

For the case of a two-dimensional images, the two-dimensional scaling functions and wavelets may be constructed from the tensor products of two scaling function and wavelet (Figure 22.a for j=0, k_1=0 and k_2=0). Hence,

$$\phi_{j,k_1,k_2}(x,y) = \phi(2^j x - k_1)\phi(2^j y - k_2) \tag{4-8}$$

$$\psi^H_{j,k_1,k_2}(x,y) = \phi(2^j x - k_1)\psi(2^j y - k_2) \tag{4-9}$$

$$\psi^V_{j,k_1,k_2}(x,y) = \psi(2^j x - k_1)\phi(2^j y - k_2) \tag{4-10}$$

$$\psi^D_{j,k_1,k_2}(x,y) = \psi(2^j x - k_1)\psi(2^j y - k_2) \tag{4-11}$$

where j, k_1, and k_2 denote the resolution level, horizontal translation, and vertical translation, respectively.

Let $u(x,y)$ and $v(x,y)$ be the displacement of flow functions at the horizontal and vertical directions, respectively. They can be closely approximated by linear combinations of the scaling functions and wavelets (window functions) given in equations (4-8) through (4-11) from the coarsest motion-resolution level -1 to the fine motion-resolution level J.

$$u(x,y) = u_{-1}(x,y) + \sum_{j=0}^{J}[u_j^H(x,y) + u_j^V(x,y) + u_j^D(x,y)] \tag{4-12}$$

$$v(x,y) = v_{-1}(x,y) + \sum_{j=0}^{J}[v_j^H(x,y) + v_j^V(x,y) + v_j^D(x,y)] \tag{4-13}$$

where

$$u_{-1}(x,y) = \sum_{k_1=-2}^{L_1-2}\sum_{K_2=-2}^{L_2-2} c_{-1,k_1,k_2}\phi_{0,k_1,k_2}(x,y) \tag{4-14}$$

$$u_j^H(x,y) = \sum_{k_1=-2}^{2^j L_1-2}\sum_{K_2=-1}^{2^j L_2-2} c^H_{j,k_1,k_2}\psi^H_{j,k_1,k_2}(x,y) \tag{4-15}$$

$$u_j^V(x,y) = \sum_{k_1=-1}^{2^j L_1-2}\sum_{K_2=-2}^{2^j L_2-2} c^V_{j,k_1,k_2}\psi^V_{j,k_1,k_2}(x,y) \tag{4-16}$$

$$u_j^D(x,y) = \sum_{k_1=-1}^{2^j L_1 - 2} \sum_{K_2=-1}^{2^j L_2 - 2} c^D_{j,k_1,k_2} \psi^D_{j,k_1,k_2}(x,y) \tag{4-17}$$

and

$$v_{-1}(x,y) = \sum_{k_1=-2}^{L_1-2} \sum_{K_2=-2}^{L_2-2} d_{-1,k_1,k_2} \phi_{0,k_1,k_2}(x,y) \tag{4-18}$$

$$v_j^H(x,y) = \sum_{k_1=-2}^{2^j L_1 - 2} \sum_{K_2=-1}^{2^j L_2 - 2} d^H_{j,k_1,k_2} \psi^H_{j,k_1,k_2}(x,y) \tag{4-19}$$

$$v_j^V(x,y) = \sum_{k_1=-1}^{2^j L_1 - 2} \sum_{K_2=-2}^{2^j L_2 - 2} d^V_{j,k_1,k_2} \psi^V_{j,k_1,k_2}(x,y) \tag{4-20}$$

$$v_j^D(x,y) = \sum_{k_1=-1}^{2^j L_1 - 2} \sum_{K_2=-1}^{2^j L_2 - 2} d^D_{j,k_1,k_2} \psi^D_{j,k_1,k_2}(x,y) \tag{4-21}$$

The four window functions are used for the following representations. $\phi_{0,k_1,k_2}(x,y)$ is used initially to represent the flow at the coarsest level so as to achieve a fast convergence. Wavelets $\psi^H_{j,k_1,k_2}(x,y)$, $\psi^V_{j,k_1,k_2}(x,y)$ and $\psi^D_{j,k_1,k_2}(x,y)$ are used to represent the high gradient components of the optical flow at the *j*th resolution level in the vertical, horizontal and diagonal directions, respectively. Based on these four basis window functions, we can estimate the Hessian matrix *G* and difference-gradient vector *e* by [101]

$$G = \sum_{x,y} g(x,y) g(x,y)^T w(x,y) \tag{4-22}$$

$$e = \sum_{x,y} [I_t(x - u(x,y), y - v(x,y)) - I_{t+1}(x,y)]\, g(x,y) w(x,y) \tag{4-23}$$

where

window (weighted) function $w(x,y)$ is unity, (4-24)

$$g(x,y)=\begin{bmatrix}\phi_{0,-2,-2}I_x^{'}\cdots\phi_{0,L_1-2,L_2-2}I_x^{'} & \psi^H_{0,-2,-1}I_x^{'}\cdots\psi^H_{j,2^jL_1-2,2^jL_2-2}I_x^{'}\\ \psi^V_{0,-1,-2}I_x^{'}\cdots\psi^V_{j,2^jL_1-2,2^jL_2-2}I_x^{'} & \psi^D_{0,-1,-1}I_x^{'}\cdots\psi^D_{j,2^jL_1-2,2^jL_2-2}I_x^{'}\\ \phi_{0,-2,-2}I_y^{'}\cdots\phi_{0,L_1-2,L_2-2}I_y^{'} & \psi^H_{0,-2,-1}I_y^{'}\cdots\psi^H_{j,2^jL_1-2,2^jL_2-2}I_y^{'}\\ \psi^V_{0,-1,-2}I_y^{'}\cdots\psi^V_{j,2^jL_1-2,2^jL_2-2}I_y^{'} & \psi^D_{0,-1,-1}I_y^{'}\cdots\psi^D_{j,2^jL_1-2,2^jL_2-2}I_y^{'}\end{bmatrix}^T \tag{4-25}$$

$$I_t^{'}(x,y)=(I_x^{'},I_y^{'})$$

$$=\left(\frac{\partial I_t(x-u(x,y),y-v(x,y))}{\partial x},\frac{\partial I_t(x-u(x,y),y-v(x,y))}{\partial y}\right) \tag{4-26}$$

Since the feature of Cai and Wang's basis functions is to use wavelet coefficients from coarse-to-fine levels to represent any given function, the flow functions $u(x,y)$ and $v(x,y)$ can be determined by estimating the wavelet coefficient vectors $c^T=(\ldots,c_{j,k_1,k_2},\ldots)$ and $d^T=(\ldots,d_{j,k_1,k_2},\ldots)$ from the coarsest level –1 to current level j using iterations of both coefficient vectors,

$$(c^T,d^T)^T=([\ldots c_{-1,k_1,k_2}\ldots c^H_{0,k_1,k_2}\ldots c^V_{0,k_1,k_2}\ldots c^D_{0,k_1,k_2}\ldots c^H_{j,k_1,k_2}\ldots c^V_{j,k_1,k_2}\ldots c^D_{j,k_1,k_2}\ldots],$$
$$[\ldots d_{-1,k_1,k_2}\ldots d^H_{0,k_1,k_2}\ldots d^V_{0,k_1,k_2}\ldots d^D_{0,k_1,k_2}\ldots c^H_{j,k_1,k_2}\ldots c^V_{j,k_1,k_2}\ldots c^D_{j,k_1,k_2}\ldots])^T$$
$$=G^{-1}e \tag{4-27}$$

where

$-2\le k_1\le L_1-2$ and $-2\le k_2\le L_2-2$ for c_{-1,k_1,k_2} and d_{-1,k_1,k_2}

$-2\le k_1\le 2^jL_1-2$ and $-1\le k_2\le 2^jL_2-2$ for c^H_{j,k_1,k_2} and d^H_{j,k_1,k_2}

$-1\le k_1\le 2^jL_1-2$ and $-2\le k_2\le 2^jL_2-2$ for c^V_{j,k_1,k_2} and d^V_{j,k_1,k_2}

$-1\le k_1\le 2^jL_1-2$ and $-1\le k_2\le 2^jL_2-2$ for c^D_{j,k_1,k_2} and d^D_{j,k_1,k_2}

and

$L_1=L_2=4$, and $j=0,1,\ldots,J$

4.2 Dense Flow Tracking

In this method, the wavelet-based dense flow is used to automatically track a large region, for example, the entire 417 x 385-pixel face image (cropped from the original 490 x 640-pixel image) for each image sequence of a certain length (from the neutral to the peak expression) so as to include the whole motion information of a facial expression (Figure 23). Since the movement of each pixel is estimated between two consecutive frames, the ending position of each tracked pixel at the previous motion estimation (between images I_{t-1} and I_t) is the beginning position at the current motion estimation (between images I_t and I_{t+1}). The motion of subpixel accuracy is estimated, and the gray value at the non-integer-valued ending position for each tracked pixel is bilinearly interpolated for further processing.

Because using the wavelet-based dense flow method is very time consuming at the present time, taking more than 2 hours for three-level (-1, 0 and 1) or 20 minutes for two-level (-1 and 0) computation between two 417 x 385-pixel frames using a SGI-Irix workstation, we use the two-level wavelet-based dense flow computation to save time. When less levels are used, it will restrict how small the window size can be at the finest level and, hence, may be less sensitive to subtle motions (less than 2 pixels). It will also miss tracking large displacements (more than 15 pixels). In spite of these, our experimental work has shown better overall recognition rate in comparison to the feature point tracking in 5 levels. Since facial expressions are produced by movements of interdependent muscles over a region, the dense flow tracking has the advantage that it can include the entire motion information and so may be more effective to capture the facial motion.

The scaling function and wavelets at multi-resolution levels provide window functions of multiple support sizes to capture both local and global characteristics in the optimization process. This makes the wavelet-based motion estimation stable and accurate, especially for the low texture regions where the large window size at the coarse level may include sufficient higher gradient information in the neighboring regions to

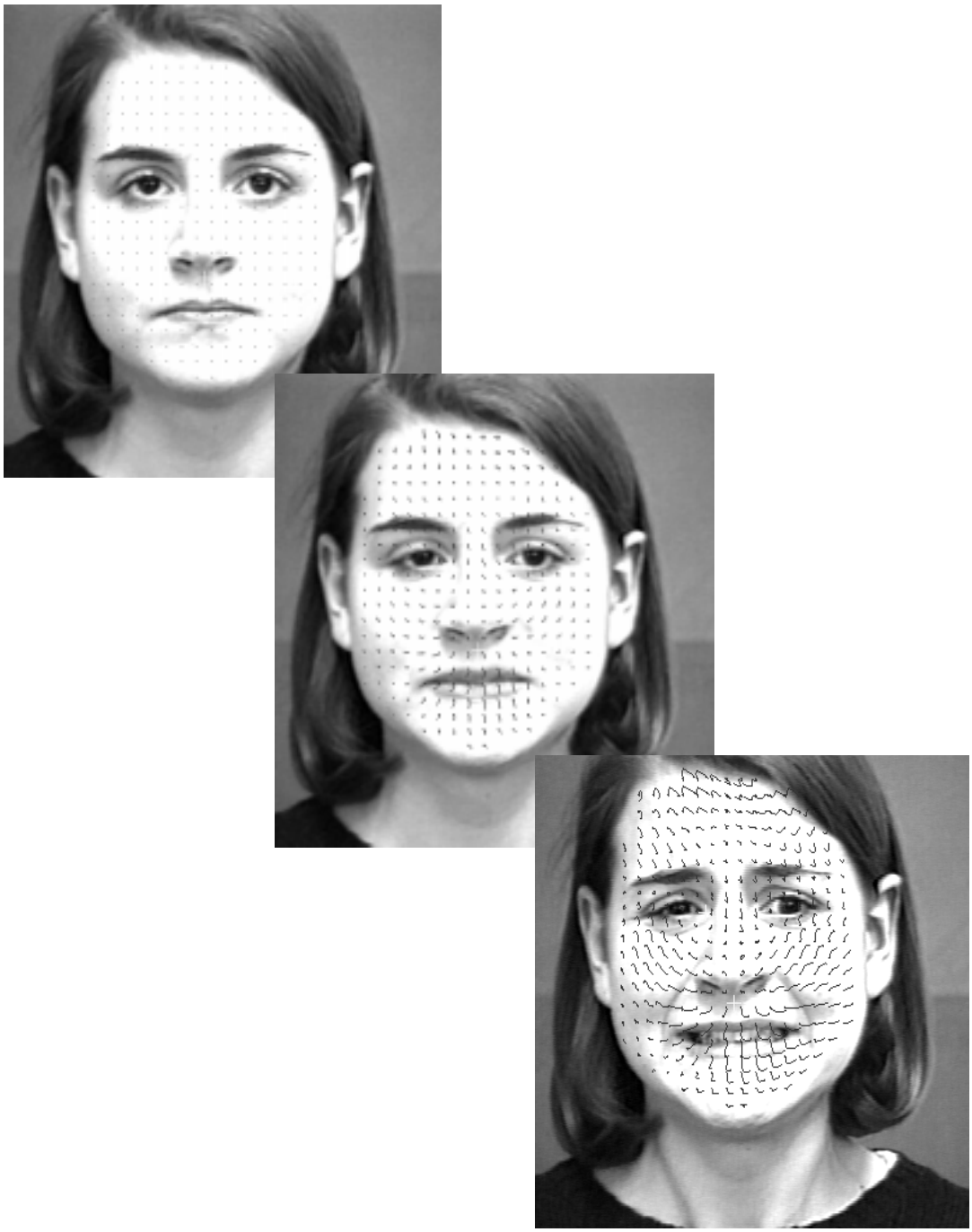


Figure 23 Automatic dense flow tracking for an image sequence. Out-of-plane motion (pitch) occurs at the bottom image. Dense flows are shown once for every 13 pixels.

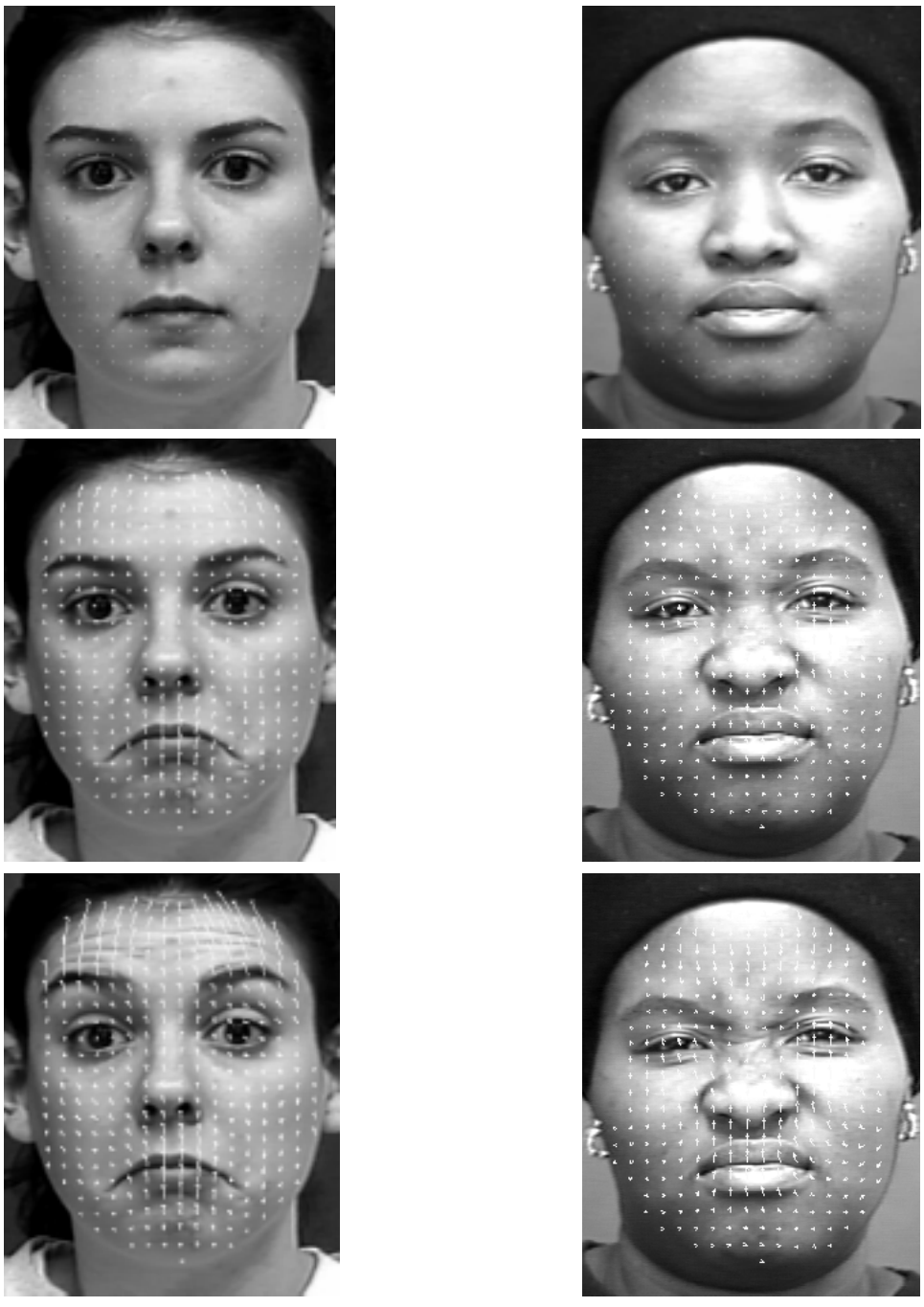
give a consistent motion information (Figure 24). It is also capable to track motions of certain discontinuities such as furrows with only two coarse-to-fine levels (-1 and 0) chosen in the current scheme, however, the resulting window functions are not localized enough to accurately capture sharp changes in the motion field (large motion more than 15 pixels) in a highly textured region such as the large movement of brows raised or of mouth opening together with the appearance of high gradient components, for example, teeth and tongue (Figure 25), while the multi-level feature point tracking can track 100- pixel movement with fast computation. Nevertheless, the overall performance of the wavelet-based dense flow is very good. The main issue is how to significantly improve the computation speed to enable more than 2-level estimation.

4.3 Eigenflow Computation

The motion captured in consecutive frames of an image sequence is strongly correlated. The information gathered by 417 x 385-pixel dense flows of many frames each sequence need to be compressed to retain significant characteristics and inter-frame correlations for yielding an efficient representation of facial expressions. The principal component analysis (PCA) has excellent properties and can be used to achieve this purpose. Although PCA has been widely applied to image gray values. This is one of the pioneering researches that it is being applied to motion fields.

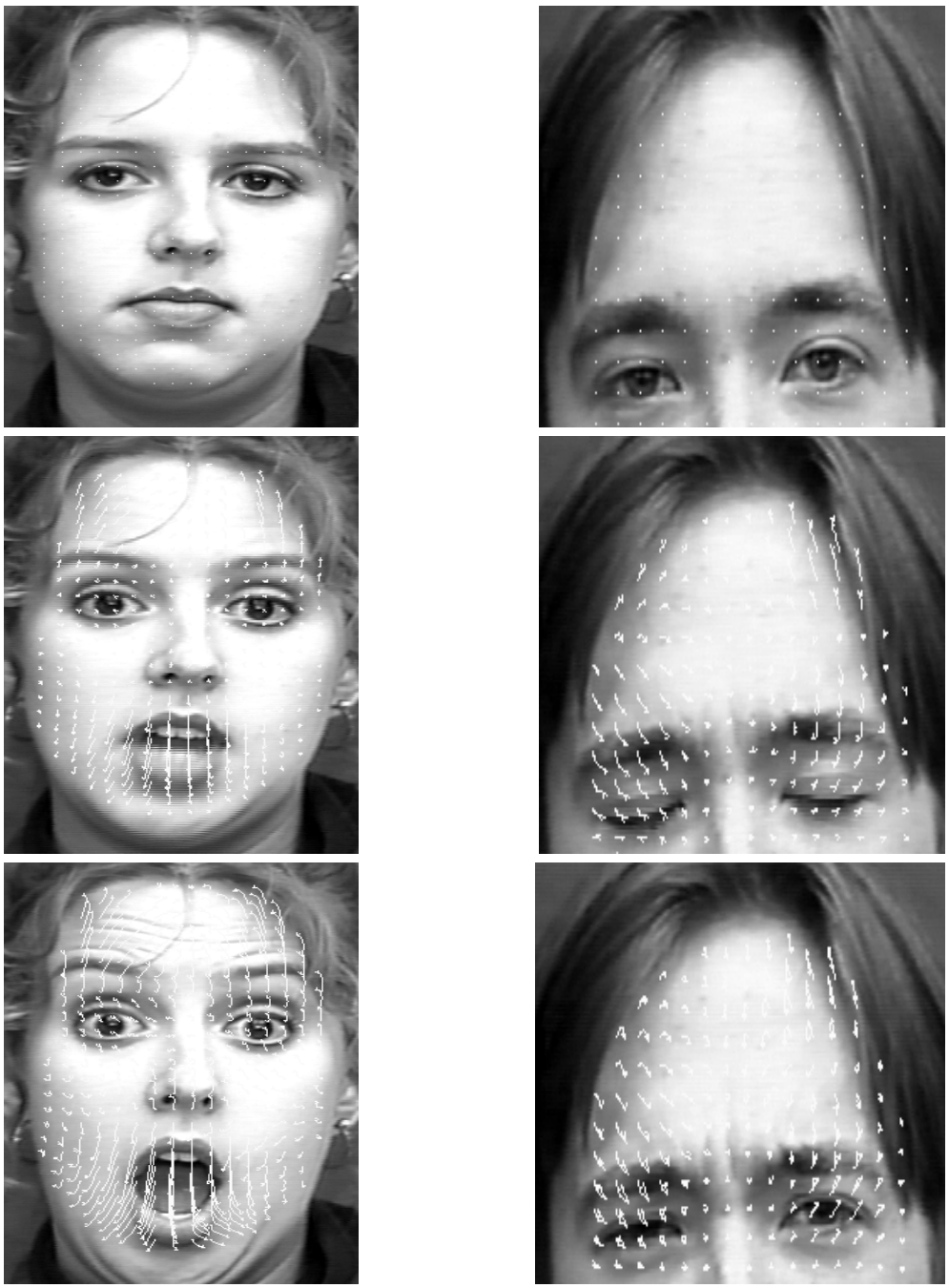
Before applying the PCA, it is necessary to ensure that the dense flows of individual frames have relative geometric correspondence. An affine transformation described before is used to automatically warp the dense flow of each frame to the two-dimensional face model based on three points: the medial canthus of both eyes and the uppermost point on the philtrum (Figure 26).

Based on FACS criteria, we can separate facial expressions into upper face motion (forehead, brows and eyes) and lower face motion (eyes, cheek, nose, mouth and chin) for facial expression analysis. It is assumed that AU expressions at upper and lower facial



(a) (b)

Figure 24 Good tracking performance of using the wavelet-based dense flow for (a) furrow discontinuities at the forehead and chin regions, and (b) textureless regions with reflections at the forehead and cheek.



(a) (b)

Figure 25 Tracking errors of the 2-level wavelet-based dense flow because of (a) large movements of brows or mouth, and (b) eye blinking also introduces motion error at brow regions.

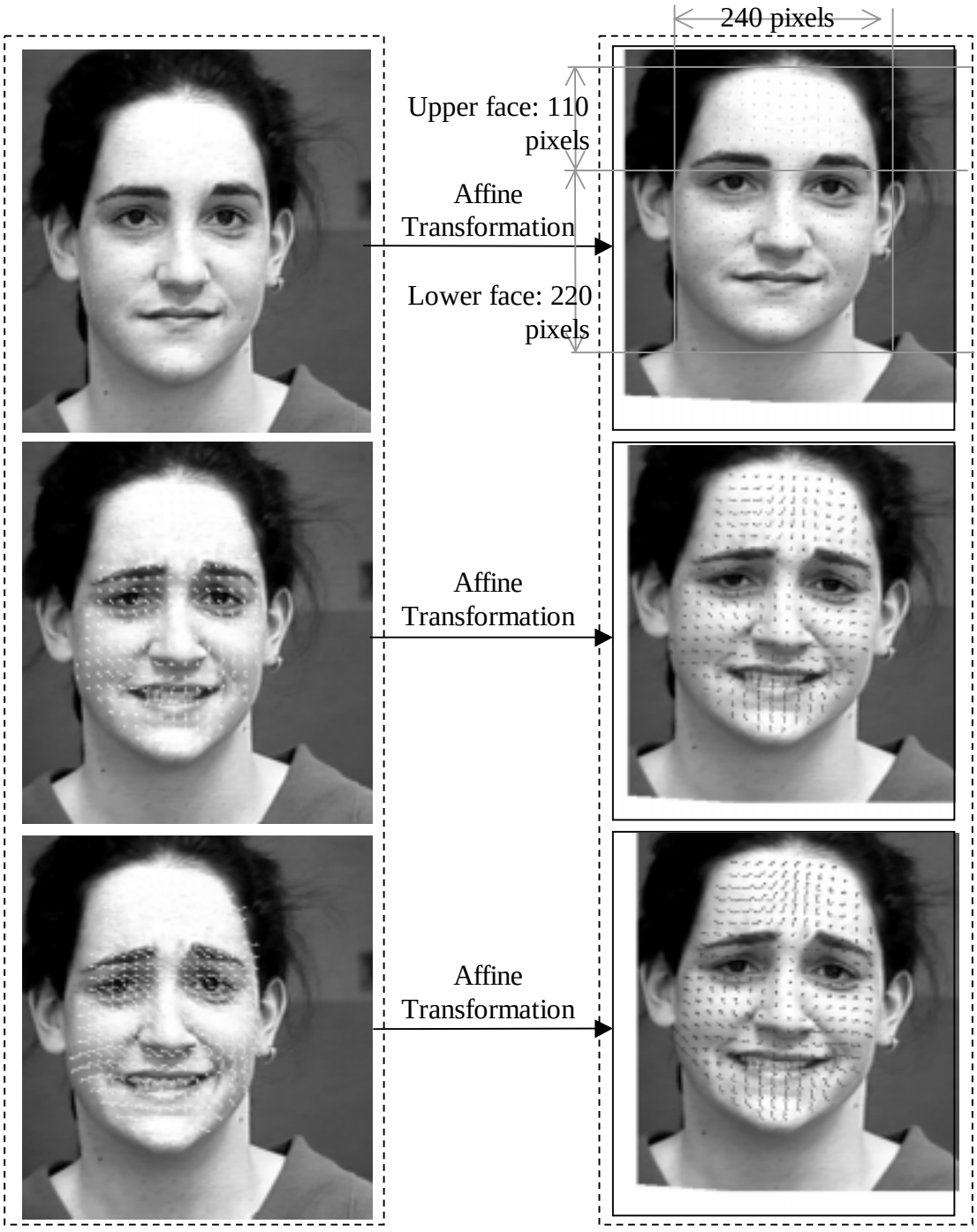


Figure 26 Dense flow normalization using affine transformation: (a) includes both the rigid head motion in upward and leftward direction and non-rigid facial expression, and (b) eliminates the rigid head motion by using the affine transformation.

regions are independent. The size of the face model is 417 x 385 (row x column) pixels. The upper face region is 110 x 240 pixels and the lower face region is 220 x 240 pixels (Figure 26). It should be noted that each flow unit contains both horizontal flow and vertical flow components. The PCA is applied to each dense flow region, the horizontal dense-flow region and vertical dense-flow region, separately.

For PCA computation, initially let the normalized estimated dense flow, either horizontal or vertical component, in a region (either upper or lower face) of frame *i* be represented lexicographically by a normalized dense-flow vector $f_{i,}$

$$f_i = [p_{i,1}, p_{i,2}, ..., p_{i,n}, ..., p_{i,N}]^T, \qquad p_{i.n} = u_{i,n} \ \ or \ \ v_{i,n} \tag{4-28}$$

where

$$1 \le i \le M \qquad and \qquad 1 \le n \le N$$

Here, $p_{i,n}$ is the normalized optical flow in either horizontal or vertical direction ($u_{i,n}$ or $v_{i,n}$) at pixel *n* of frame *i,* and *N* is the total number of pixels of frame *i* (or in upper or lower face region). There are *X* different facial expressions and a total of *M* training frames. The number of frames for each facial expression sequence varies from 9 to 47 frames. The variance matrix *F* of all normalized dense-flow training frames is given by

$$\begin{aligned} F &= [F_1, F_2, ..., F_i, ..., F_M] \\ &= [f_1 - c, f_2 - c, ..., f_i - c, ..., f_M - c] \qquad 1 \le i \le M \end{aligned} \tag{4-29}$$

where

$$c = \frac{\sum_{i=1}^{M} f_i}{M} \tag{4-30}$$

c is the mean flow (Figure 27) and the size of *F* is *N* x *M*. The *N* x *N* covariance matrix *C* of all normalized dense flows is given by

$$C = F\,F^T \tag{4-31}$$

which will have *N* different *N*-dimensional eigenvectors E_i (called eigenflows because of the flow-based eigenspace) corresponding to *N* eigenvalues λ_i of *C* ranked in the descending order, $\lambda_1 \ge \lambda_2 \ge ... \ge \lambda_N$,

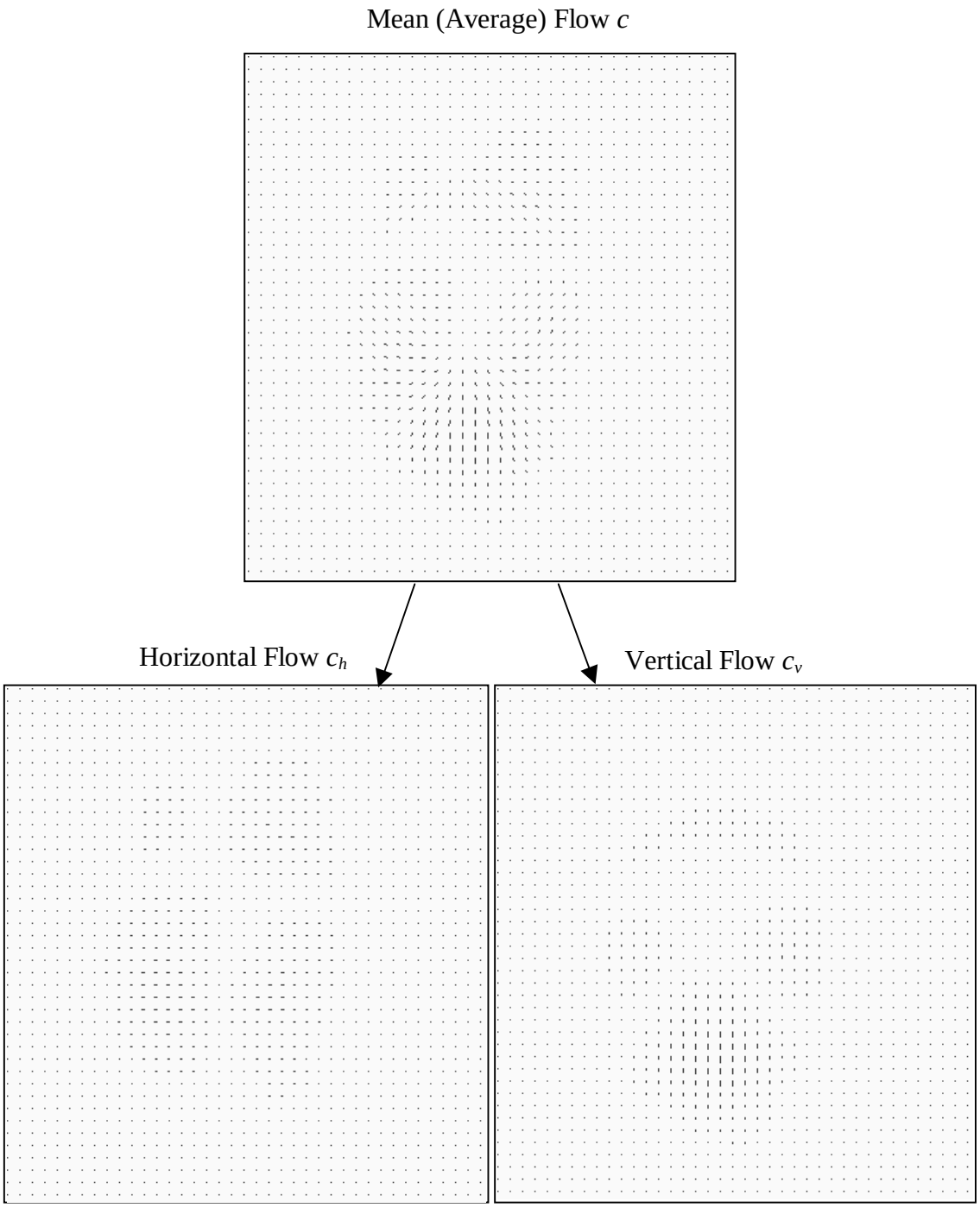


Figure 27 The mean (average) flow c is divided into horizontal flow c_h and vertical flow c_v for further processing by the principal component analysis (PCA).

$$C E_i = \lambda_i E_i, \qquad 1 \le i \le N \tag{4-32}$$

Generally, N is much larger than the total of training frames M, *i.e.* $N >> M$, where N is 110 x 240 pixels for the upper face region and 220 x 240 pixels for the lower face region, and M is 932 and 1212 training frames in the respective region. It will be impractical to compute eigenvalue λ_i and eigenvectors E_i directly from the N x N covariance matrix C. We will flow a more feasible computation approach described below.

When total number of dense-flow training frames M is far less than the number of dense flows N in the measured region, there are only M meaningful eigenvectors (eigenflows) associating with the non-zero eigenvalues of C. The remaining N - M eigenvectors correspond to zero eigenvalues. We can use a much more efficient method to obtain the M meaningful eigenvectors E_i by solving for the M-dimensional eigenvectors e_i and their corresponding eigenvalues $\tilde{\lambda}_i$ of an M x M covariance matrix $\tilde{C}$, where

$$\tilde{C} = F^T F \tag{4-33}$$

$$\tilde{C} e_i = \tilde{\lambda}_i e_i \qquad 1 \le i \le M \tag{4-34}$$

Multiplying both sides by F, gives

$$F F^T F e_i = \tilde{\lambda}_i F e_i \qquad 1 \le i \le M \tag{4-35}$$

then

$$C F e_i = \tilde{\lambda}_i F e_i \qquad 1 \le i \le M \tag{4-36}$$

F x e_i are the first M eigenvectors of the covariance matrix C, leading to

$$\begin{aligned} E_i &= F e_i \\ &= \sum_{j=1}^{M} e_{i,j} F_j \\ &= \sum_{j=1}^{M} e_{i,j} (f_j - c) \quad 1 \le i \le M \end{aligned} \tag{4-37}$$

This computation greatly reduces the order of time complexity from *N*-dimensional dense flows of each measured region to *M* dense-flow training frames where $M << N$.

We can reconstruct exactly the original dense flows of each measured region by a liner combination of all *M* eigenflows E_i. But, in general, using a small number *M'* (*M'* < *M*) of significant eigenflows that correspond to *M'* largest eigenvalues is adequate to reconstruct a good approximation of the original dense flow in each measured region without losing significant feature characteristics. The eigenflows (eigenvectors) with the largest eigenvalues represent the most significant characteristics in their corresponding dimensions of the eigenspace, where the variances of measured dense flows are bigger in terms of correlation. We rank the eigenflows according to the ranking of their corresponding eigenvalues, which contain the most useful information about the variational characteristics among the training dense flows. Furthermore, if the motion variation (deviation) among the training regions is large, we need a large number *M'* (*M'* < *M*) of eigenflows to accurately approximate the original dense flows. If the variation of motion among training regions is small, then a very small number *M'* (*M'* << *M*) of eigenflows is adequate to approximate the original dense flows. To determine the number *M'* of eigenflows needed to represent adequately the primary characteristics of the original dense flows in the region, we may use an information criterion such that

$$R = \frac{\sum_{i=1}^{M'} \tilde{\lambda}_i}{\sum_{i=1}^{M} \tilde{\lambda}_i} \geq T \quad \textit{where} \quad \tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \ldots \geq \tilde{\lambda}_{M'} \geq \ldots \geq \tilde{\lambda}_M \tag{4-38}$$

and

$$R \ln R \qquad \text{is the representative entropy} \tag{4-39}$$

where *T* is a threshold close to unity. The linear combination (the weighted sum) of the *M'* eigenflows is sufficient to reconstruct accurately the significant characteristics of each original dense flows in the region. The *M' N*-dimensional eigenflows E_i are computed by

$$E_i = \sum_{j=1}^{M} e_{i,j} F_j$$

$$= \sum_{j=1}^{M} e_{i,j} (f_j - c) \quad 1 \le i \le M' \tag{4-40}$$

We project each variational dense-flow F_i (N dimensions) of facial expression sequence into the eigenflow space by taking its inner product with each eigenflow E_j (N dimensions) of the set $E = [E_1, E_2, ..., E_{M'}]$ in the M'-dimensional subspace to produce the M'-dimensional weight vector W_i. Each element $w_{i,j}$ of the weight vector W_i is the projected component of F_i at the eigenflow dimension E_j in the eigenspace. Any N-dimensional normalized dense-flow f_i can be represented by its corresponding M'-dimensional weight vector W_i

$$W_i = [w_{i,1}, w_{i,2}, ..., w_{i,M'}]^T \qquad \textit{where } 1 \le i \le M \tag{4-41}$$

and

$$w_{i,j} = E_j F_i^T$$

$$= E_j (f_i - c)^T \qquad \textit{where } 1 \le i \le M \textit{ and } 1 \le j \le M' \tag{4-42}$$

by projecting the N-dimensional variational region F_i to the M'-dimensional eigenspace ($M' << N$).

4.4 Data Quantization and Conversion for the Recognition System

Figure 28 shows the flow image which will project to the flow-based eigenspace for PCA process. PCA enables us to convert dense flows of each image (or region) to a low-dimensional representative weight vector for input to the recognition system. For the 110 x 240-pixel upper face region, 10 most significant eigenflows are chosen and for the 220 x 240-pixel lower face region, 15 most significant eigenflows are chosen. The decision for the choice of M' is that R should be greater than or equal to T where T is set to be 0.9 (refer to equation (4-38)), which led us to select M'=10 for the upper facial expressions

and M'=15 for the lower facial expressions as shown in Figure 29. The same of eigenflows should be used for both horizontal flow and vertical flow. These eigenflows are the eigenvectors corresponding to the 10 (and 15) largest eigenvalues of the 932 x 932- (and 1212 x 1212-) covariance matrix constructed by 932 (and 1212) dense-flow training frames from 45 (and 60) training image sequences for the upper (and lower) face regions (Figure 29). The compression rate is 93:1 as shown in Figure 29.a (and 80:1 as shown in Figure 29.b). Because the variation (deviation) among the training data of the upper facial "expression units" is smaller than that of the lower facial "expression units," it is expected that the number of eigenflows used for representing the upper facial "expression units" (M' = 10) is fewer than that used for representing the lower facial "expression units" (M' = 15). As shown by the later experiments, this choice leads to good performance on overall recognition rate of 92% (Figure 29).

The dense flow at each frame region of an expression sequence is projected onto the flow-based eigenspace by taking its inner product with each element of the respective eigenflow set, producing a 10- (and 15-) dimensional weight vector for the upper (and lower) facial expressions as shown in Figure 30. The 10- (and 15-) dimensional horizontal-flow weight vector and 10- (and 15-) dimensional vertical-flow weigh vector are concatenated to form a 20- (and 30-) dimensional weight vector for each dense-flow region. After vector quantization, the concatenated weight-vector sequence is converted into a symbol sequence. Table 6 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.

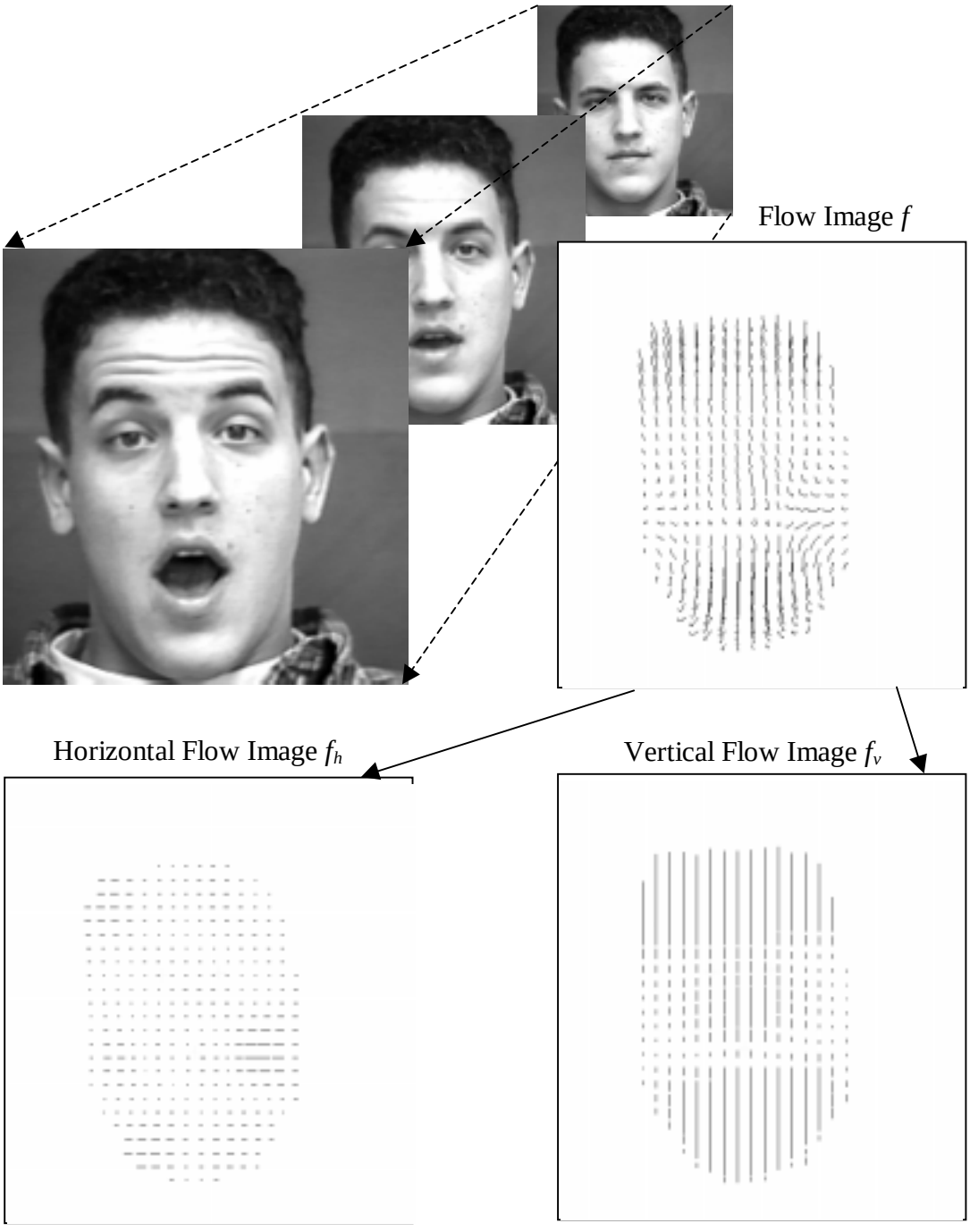
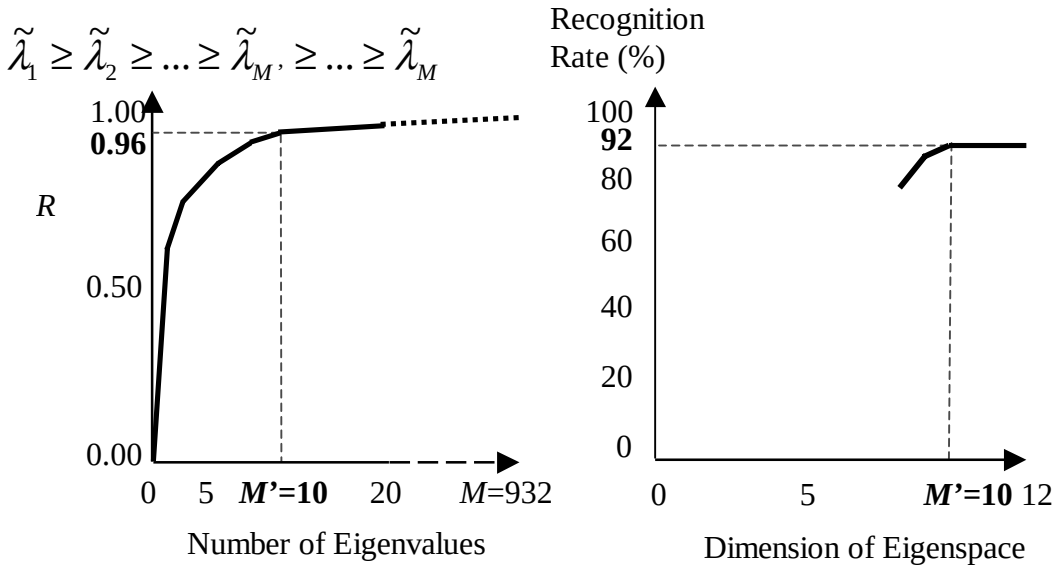
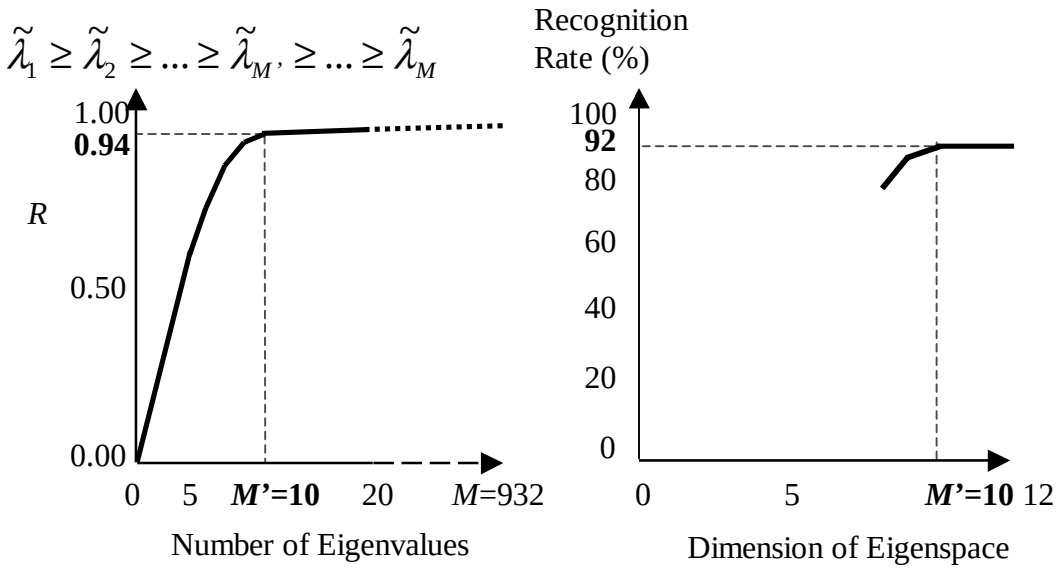


Figure 28 Each dense flow image is divided into horizontal and vertical flow images for the principal component analysis (PCA).

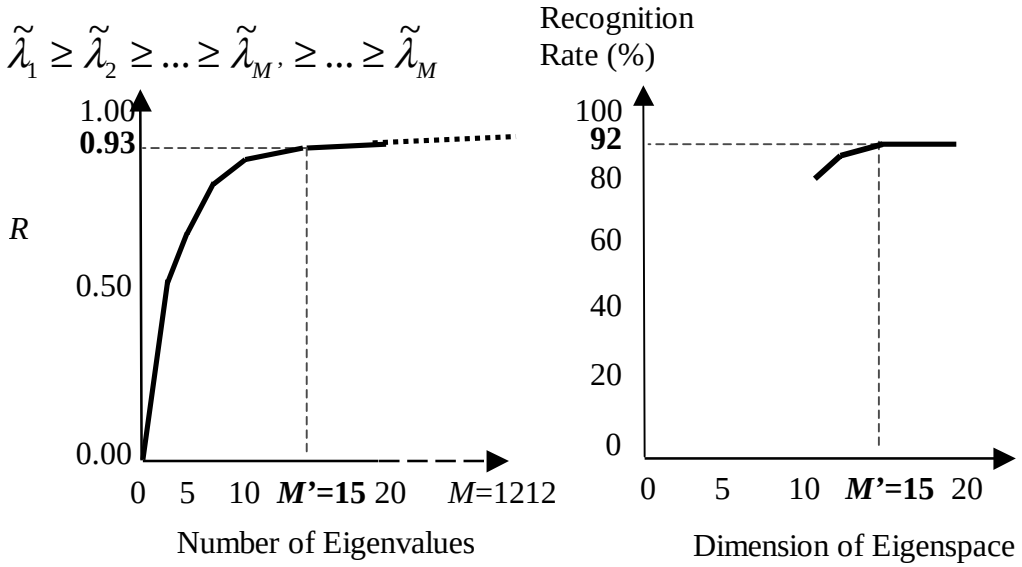


(a.1)

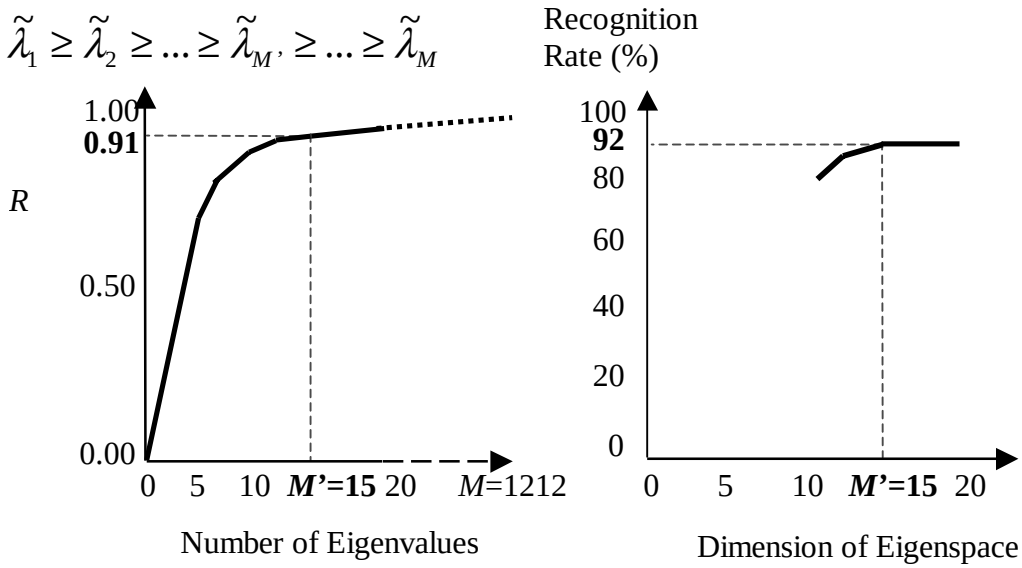


(a.2)

Figure 29.a Computation of eigenflow (eigenvector) number for the upper facial expressions: (a.1) is for the horizontal flow and (a.2) is for the vertical flow. The compression rate is 93:1 (932:10) and from which the recognition rate is 92% based on 45 training and 60 testing image sequences.



(b.1)



(b.2)

Figure 29.b Computation of eigenflow (eigenvector) number for the lower facial expressions: (b.1) is for the horizontal flow and (b.2) is for the vertical flow. The compression rate is 80:1 (1212:15) and from which the recognition rate is 92% based on 60 training and 90 testing image sequences.

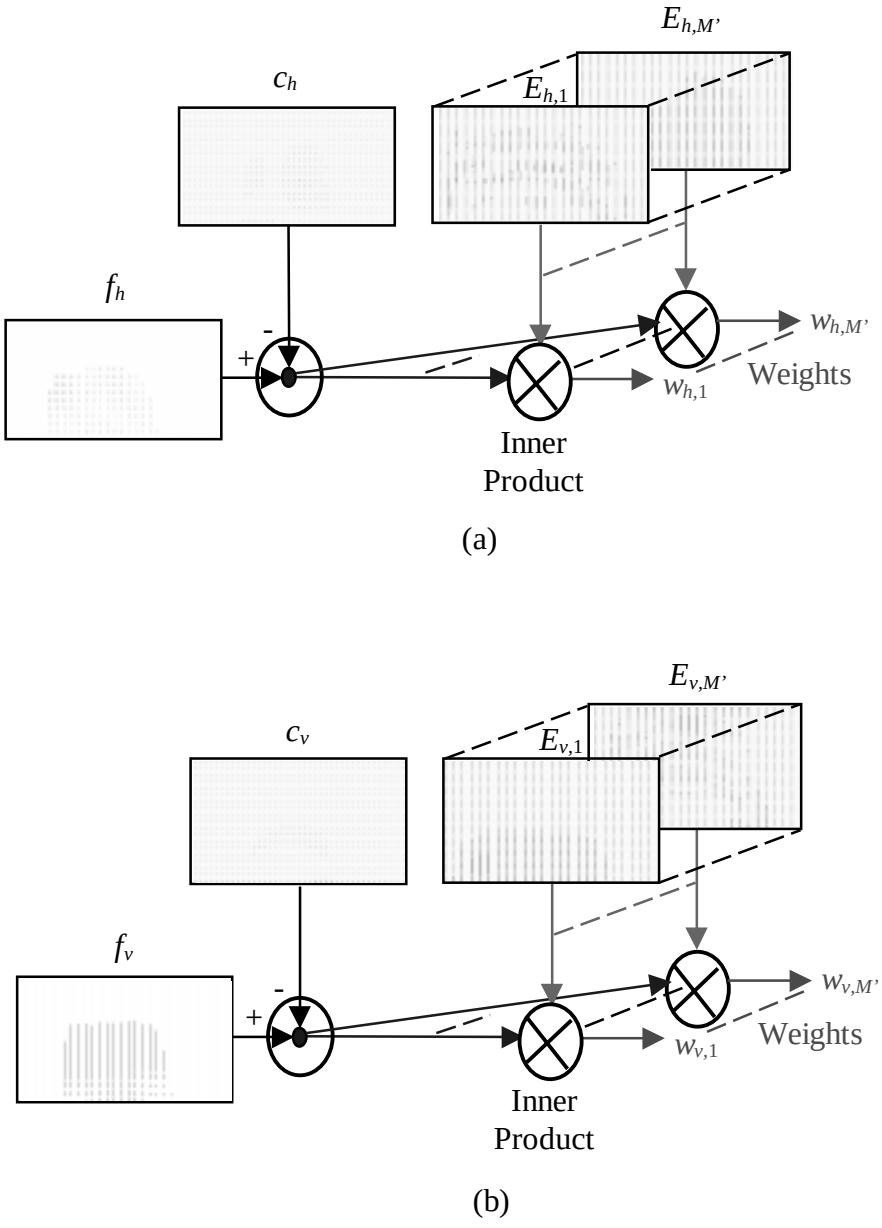


Figure 30 Principal component analysis (PCA) for (a) horizontal flow weight vector and (b) vertical flow weight vector for the upper facial expressions (M' = 10).

Table 6 Sample symbol sequences for three upper facial expressions and six lower facial expressions under consideration.

AUs	**Dense Flows with PCA: 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* 7 7 7 7 *7* 7
1+2	3 3 3 6 *6* 2 2 2 2 *2* 2 2 2
AUs	**Dense Flow with PCA: Lower Facial Expressions (Symbol Sequence)**
12	6 6 6 6 *6* 6 9 9 1 *1* 1
6+12+25	6 6 9 9 *3* 3 3 3 3 *3* 3
20+25	6 6 6 11 *13* 13 13 13 13 *13*
9+17	6 6 6 7 *7* 7 7 2 2 *2* 2
15+17	6 6 6 6 *6* 14 14 14 14 *14* 14 14
17+23+24	6 6 6 12 *12* 4 4 4 4

4.5 Correlation and Distance in Eigenspace

The sum of squared differences (SSD) between two dense-flow frame regions f_i and f_j is given by

$$\begin{aligned} \left\| f_i - f_j \right\|^2 &= (f_i - f_j)^T (f_i - f_j) \\ &= (f_i^T f_i + f_j^T f_j) - 2 f_i^T f_j \end{aligned} \tag{4-43}$$

where $f_i^T f_j$ represents the correlation between f_i and f_j. The distance SSD becomes smaller when the correlation between the two flows is stronger. Each dense flow f_i in frame i can be represented by its representative weight vector W_i in the M'-dimensional eigenflow space. It can be reconstructed by a linear combination of M' eigenflows E_j.

$$f_i \approx \sum_{j=1}^{M'} w_{i,j} E_j + c \tag{4-44}$$

where $w_{i,j}$ is the jth element of the weight vector W_i flow f_i. Since

$$\begin{aligned} \left\| f_i - f_j \right\|^2 &\approx \left\| \left(\sum_{k=1}^{M'} w_{i,k} E_k + c \right) - \left(\sum_{k=1}^{M'} w_{j,k} E_k + c \right) \right\|^2 \\ &= \left\| \sum_{k=1}^{M'} (w_{i,k} - w_{j,k}) E_k \right\|^2 \\ &= \sum_{k=1}^{M'} \sum_{l=1}^{M'} E_k^T E_l * (w_{i,k} - w_{j,l})^2 \\ &= \begin{cases} \sum_{k=1}^{M'} (w_{i,k} - w_{j,k})^2 & \text{if} \quad k = l \\ 0 & \text{otherwise} \ (\text{since } E_k \perp E_l) \end{cases} \end{aligned} \tag{4-45}$$

This can also be expressed as

$$\left\| f_i - f_j \right\|^2 \approx \left\| W_i - W_j \right\|^2 \tag{4-46}$$

Thus, we can estimate the expression similarity between two dense flows f_i and f_j by measuring the distance between their representative weight vectors W_i and W_j in the M'-

dimensional eigenspace. Smaller distance indicates greater correlation or similarity between two dense flows. This can be used in expression intensity estimation as described below.

4.6 Expression Intensity Estimation

The expression intensity of any frame in a sequence may be estimated by using the correlation property of the PCA. The expression intensity of individual frames in each training image sequence is determined a priori by experts, beginning from the neutral expression (expression intensity: 0.0) to the peak expression (expression intensity: 1.0). The length of each image sequence varies from 9 to 47 frames. Each frame in the training sequence has its representative weight vector. So, the relationship between weight vector and expression intensity can be established.

After a test facial expression sequence is recognized, the expression intensity of any frame f_i in the sequence can be estimated as follows. Consider the distances between its representative weight vector W_i and all weight vectors W_k^0 in a training sequence whose expression intensity values are known, the minimum distance between W_i and W_k^0 indicates the maximum correlation or expression similarity between the two. Then, the expression intensity of frame i in the testing sequence will be estimated to have the same value as that of W_k^0 training data (Figure 31).

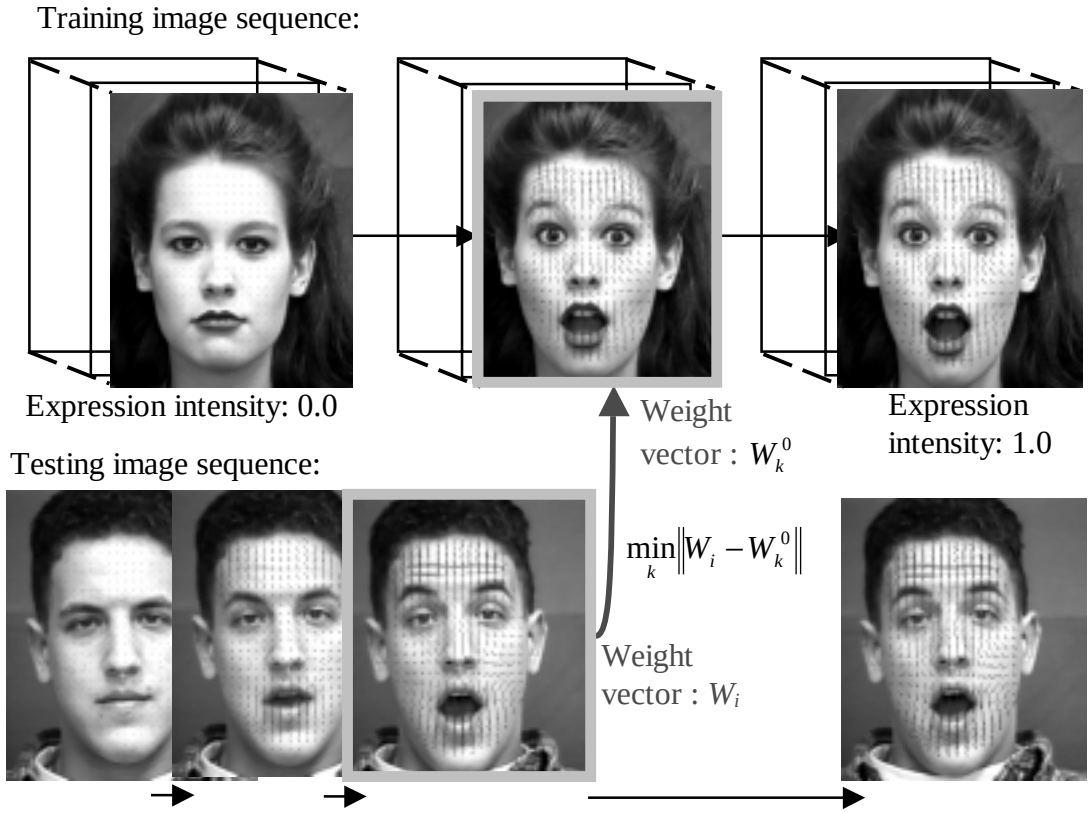


Figure 31 Expression intensity matching by seeking the minimum distance between the weight vector of the testing frame and the weight vectors of all frames in a training sequence, whose expression intensity values are known. Each weight vector of the training image corresponds to a given expression intensity value.