Chapter 5
Face Alignment Models

Phil Tresadern, Tim Cootes, Chris Taylor, and Vladimir Petrović

5.1 Introduction

In building models of facial appearance, we adopt a statistical approach that learns the ways in which the shape and texture of the face vary across a range of images. We rely on obtaining a suitably large and representative training set of images of faces, each of which is annotated with a set of feature points that define correspondences across the set. The positions of the feature points also define the shape of the face, and are analysed to learn the ways in which the shape can vary. The patterns of intensities are analysed in a similar way to learn how the texture can vary. The result is a model which is capable of synthesising any of the training images and generalising from them, but is specific enough that only face-like images are generated.

To build a statistical appearance model, we require a set of training images that covers the types of variation we want the model to represent. For instance, if we are only interested in faces with neutral expressions, we need only include neutral expressions in the model. If, however, we want to synthesise and recognise a range of expressions, the training set should include images of people smiling, frowning, winking and so on. Ideally, the faces in the training set should be of at least as high a resolution as those in the images we wish to synthesise or interpret.

5.1.1 Statistical Models of Shape

To define a shape model, we first annotate each face with a fixed number of points that define the key facial features (and their correspondences across the training set) and represent the shape of the face in the image. Typically, we place points around

P. Tresadern · T. Cootes (✉) · C. Taylor · V. Petrović
Imaging Science and Biomedical Engineering, University of Manchester, Manchester, UK
e-mail: t.cootes@man.ac.uk

S.Z. Li, A.K. Jain (eds.), *Handbook of Face Recognition*,
DOI 10.1007/978-0-85729-932-1_5,

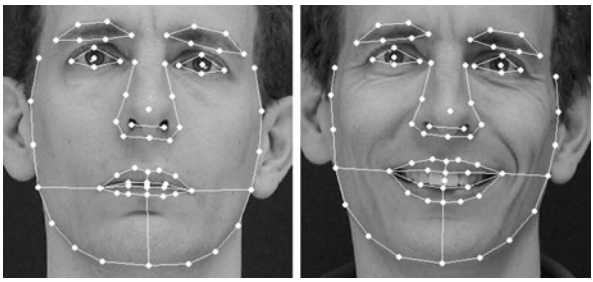


Fig. 5.1 Examples of 68 points defining facial features on two frontal images

the main facial features (eyes, nose, mouth and eye-brows) together with points that define the boundary of the face (Fig. 5.1). The more points we use, the more subtle the variations in shape that we can represent.

If we annotate the face with n feature points, $\{(x_i, y_i)\}$, then we can represent the geometry of the face with a $2n$ element vector,

$$\mathbf{x} = (x_1, \ldots, x_n, y_1, \ldots, y_n)^{\mathrm{T}}, \tag{5.1}$$

such that N training images provide N training vectors $\mathbf{x}_i$. The *shape* of the face can then be defined as that property of the configuration of points which is invariant under (that is, not explained by) some global transformation. In other words, if $S_{\mathbf{t}}(\mathbf{x})$ applies a transformation defined by parameters $\mathbf{t}$ to the points $\mathbf{x}$, the configurations of points defined by $\mathbf{x}$ and $S_{\mathbf{t}}(\mathbf{x})$ are considered to have the same shape. Typically, we use either the similarity transformation or the affine transformation where a 2D similarity has four parameters (x- and y-translation, rotation and scaling) and a 2D affine transformations has six (x- and y-translation, rotation, scaling, aspect ratio and skew).

Given a set of shapes as training data, we can then apply formal statistical techniques [23] to analyse their variation and synthesise new shapes that are similar. Before we perform statistical analysis on these training vectors, however, it is important that we first remove any differences that are attributable to the global transformation, $S_{\mathbf{t}}(\mathbf{x})$, leaving only genuine differences in shape (that is, we must align the shapes into a common coordinate frame).

5.1.1.1 Aligning Sets of Shapes

Of the various methods of aligning shapes into a common coordinate frame, the most popular is *Procrustes Analysis* [27] that finds the parameters, $\mathbf{t}$, that transform each shape in the set, $\mathbf{x}_i$, so that it is aligned with a mean shape, $\bar{\mathbf{x}}$, in the sense that minimises their sum of squared distances, $D = \sum_i |S(\mathbf{x}_i) - \bar{\mathbf{x}}|^2$. Though we can solve this analytically for a *set* of shapes, in practice we use a simple and effective iterative approach (Algorithm 5.1) to converge on a solution. At each iteration, we ensure that this measure is well defined by aligning the mean shape to the coordinate frame such that it is centred at the origin, has unit scale and some fixed but arbitrary orientation.

Algorithm 5.1: Aligning a set of shapes

1: Translate each example so that its centre of gravity is at the origin.
2: Choose one example as an initial estimate of the mean shape, $\bar{\mathbf{x}}$, and scale so that $|\bar{\mathbf{x}}| = 1$.
3: Record this first estimate as $\bar{\mathbf{x}}_0$ to define the default reference frame.
4: **repeat**
5: Align each shape with the current estimate of the mean shape.
6: Re-estimate mean from aligned shapes.
7: Apply constraints on the current estimate of the mean by aligning it with $\bar{\mathbf{x}}_0$ and scaling so that $|\bar{\mathbf{x}}| = 1$.
8: **until** converged (that is, the change in the mean since the previous iteration is sufficiently small).

5.1.1.2 Linear Models of Shape Variation

We now have N training vectors, $\mathbf{x}_i$, each with n points in d dimensions (usually $d = 2$ or $d = 3$) that are aligned to a common coordinate frame. By treating these vectors as points in nd-dimensional space and modelling their distribution, we can generate new examples that are similar to those in the original training set (that is, sample from the distribution) and examine new shapes to decide whether they are plausible examples (that is, evaluate a sample's probability).

Not every nd-dimensional vector forms a face, however, and the set of valid face shapes typically lies on a manifold that can be described by $k < nd$ underlying model parameters, $\mathbf{b}$. By computing a parameterised model of the form $\mathbf{x} = M(\mathbf{b})$, we can approximate the distribution over shape, $p(\mathbf{x})$, with the distribution over model parameters, $p(\mathbf{b})$, and therefore generate new shapes by sampling from $p(\mathbf{b})$ and applying the model, or evaluate a shape's probability by evaluating the probability of its corresponding parameters, $p(M^{-1}(\mathbf{x}))$. The simplest approximation we can make to the manifold is a linear subspace that passes through the mean shape (equivalent to assuming a Gaussian distribution over both $\mathbf{x}$ and $\mathbf{b}$). An effective approach to estimating the subspace parameters is to apply Principal Component Analysis (PCA) to the training vectors, $\mathbf{x}_i$ (Algorithm 5.2). This computes the directions of greatest variance in nd-dimensional space, of which we keep only the k most 'significant'. Each training example can then be described by its $k < nd$ projections onto these directions, reducing the effective dimensionality of the data.

We can then approximate any example shape from the training set, $\mathbf{x}$ using

$$\mathbf{x} \approx \bar{\mathbf{x}} + \mathbf{P}_s \mathbf{b}_s \tag{5.4}$$

where $\mathbf{P}_s = (\phi_1 | \phi_2 | \ldots | \phi_k)$ contains the eigenvectors corresponding to the k largest eigenvalues and $\mathbf{b}_s$ is a k-dimensional vector given by

$$\mathbf{b}_s = \left(\mathbf{P}_s^{\mathrm{T}} \mathbf{P}_s\right)^{-1} \mathbf{P}_s^{\mathrm{T}} (\mathbf{x} - \bar{\mathbf{x}}) = \mathbf{P}_s^{\mathrm{T}} (\mathbf{x} - \bar{\mathbf{x}}) \tag{5.5}$$

Algorithm 5.2: Principal component analysis

1: Compute the mean of the data,

$$\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i. \tag{5.2}$$

2: Compute the covariance of the data,

$$\mathbf{S} = \frac{1}{N-1}\sum_{i=1}^{N}(\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^{\mathrm{T}}. \tag{5.3}$$

3: Compute the orthonormal eigenvectors, ϕ_j, and their corresponding eigenvalues, λ_j, of $\mathbf{S}$ (sorted such that $\lambda_j \geq \lambda_{j+1}$). Efficient methods of computing the eigenvectors and eigenvalues exist for the case in which there are fewer samples than dimensions in the vectors [11].

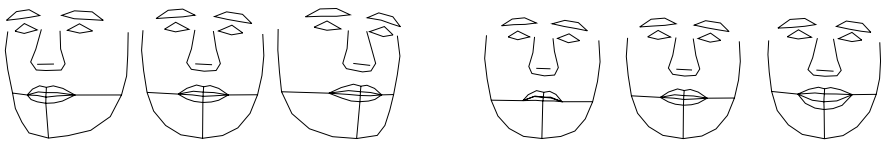


Fig. 5.2 Two modes of a face shape model (varied by ±2 s.d. from the mean). The first shape mode corresponds mostly to 3D head rotation whereas the second captures facial expression

where $\mathbf{P}_s^{\mathrm{T}}\mathbf{P}_s = \mathbf{I}$ because the eigenvectors are orthonormal. The vector $\mathbf{b}_s$ therefore defines a set of parameters of a deformable shape model and by varying the elements of $\mathbf{b}_s$ we can vary the generated shape, $\mathbf{x}$, via (5.4).

Given a shape in the model frame, $\mathbf{x} = M(\mathbf{b})$, we can then generate the corresponding shape in the image frame, $\mathbf{X}$, by applying a suitable transformation such that $\mathbf{X} = S_{\mathbf{t}}(\mathbf{x})$. Typically $S_{\mathbf{t}}$ will be a similarity transformation described by a scaling, s, an in-plane rotation, θ, and a translation, (t_x, t_y). By representing the scaling and rotation jointly as (s_x, s_y), where $s_x = (s\cos\theta - 1)$ and $s_y = s\sin\theta$, we ensure that the pose parameter vector, $\mathbf{t} = (s_x, s_y, t_x, t_y)^{\mathrm{T}}$, is zero for the identity transformation and that $S_{\mathbf{t}}(\mathbf{x})$ is linear in the pose parameters.

Due to the ordering of the eigenvalues, λ_j, the corresponding modes of shape variation are sorted in descending order of 'importance'. For example, a model built from examples of a single individual with different viewpoints and expressions will often select 3D rotation of the head (that causes a large change in the projected shape) as the most significant mode, followed by expression (Fig. 5.2).

5.1.1.3 Choosing the Number of Shape Modes

The number of modes, k, that we keep determines how much meaningful shape variation (and how much meaningless noise) is represented by the model, and should therefore be chosen with care. A simple approach is to choose the smallest k that explains a given proportion (e.g., 98%) of the total variance exhibited in the training set, $V_T = \sum \lambda_j$, where each eigenvalue λ_j gives the variance of the data about the mean in the direction of the jth eigenvector. More specifically, it is normal to choose the smallest k that satisfies

$$\sum_{j=1}^{k} \lambda_j \geq f_v \cdot V_T \tag{5.6}$$

where f_v defines the proportion of V_T that we want to explain (e.g., $f_v = 0.98$). Alternatively, we could build a sequence of models with increasing k and choose the smallest model that approximates all training examples to within a given accuracy (e.g., at most one pixel of error for any feature point). Performing this test in a miss-one-out manner—testing each example against models built from all other examples—gives us further confidence in the chosen k.

5.1.1.4 Fitting the Model to New Points

Suppose now we wish to find the best pose and shape parameters to align a model instance $\mathbf{x}$ to a new set of image points, $\mathbf{X}'$. Minimising the sum of square distances between corresponding model and image points is equivalent to minimising the expression

$$E_{\text{pts}} = \left|\mathbf{X}' - S_{\mathbf{t}}(\bar{\mathbf{x}} + \mathbf{P}_s \mathbf{b}_s)\right|^2 \tag{5.7}$$

or, more generally,

$$E_{\text{pts}} = \left(\mathbf{X}' - S_{\mathbf{t}}(\bar{\mathbf{x}} + \mathbf{P}_s \mathbf{b}_s)\right)^{\mathrm{T}} \cdot \mathbf{W}_{\text{pts}} \cdot \left(\mathbf{X}' - S_{\mathbf{t}}(\bar{\mathbf{x}} + \mathbf{P}_s \mathbf{b}_s)\right) \tag{5.8}$$

where $\mathbf{W}_{\text{pts}}$ is a diagonal matrix that applies a different weight for each point. If the allowed global transformation $S_{\mathbf{t}}(\cdot)$ is more complex than a simple translation then this is a non-linear equation with no analytic solution. A good approximation can be found rapidly, however, by using a two-stage iterative approach (Algorithm 5.3) where each step solves a linear equation for common choices of transformation (e.g., similarity or affine) [30].

If the weights in $\mathbf{W}_{\text{pts}}$ relate to the uncertainty in the estimates of positions of target points, $\mathbf{X}'$, then E_{pts} can be related to the log likelihood. Adding a term representing the prior distribution on the shape parameters, $p(\mathbf{b}_s)$, then gives the log posterior,

$$E'_{\text{pts}} = E_{\text{pts}} - 2 \log p(\mathbf{b}_s). \tag{5.9}$$

Algorithm 5.3: Shape model fitting

1: **repeat**
2: Solve for the pose parameters, $\mathbf{t}$, assuming a fixed shape, $\mathbf{b}_s$.
3: Solve for the shape parameters, $\mathbf{b}_s$, assuming a fixed pose, $\mathbf{t}$.
4: **until** convergence.

If the training shapes, $\{\mathbf{x}_i\}$, have a multivariate Gaussian distributed then the parameters, $\mathbf{b}$, will have an axis-aligned Gaussian distribution, $p(\mathbf{b}) = N(\mathbf{0}, \Lambda)$, where $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_k)$ such that

$$\log p(\mathbf{b}_s) \propto \sum_{j=1}^{k} b_j^2/\lambda_j + \text{const.} \tag{5.10}$$

This prior, however, biases shape parameters toward zero which may be unjustified. Therefore, a common approach is to apply a uniform elliptical prior over shape parameters such that

$$\log p(\mathbf{b}_s) = \begin{cases} \text{const} & \text{if } \mathbf{b}^{\mathrm{T}} \Lambda^{-1} \mathbf{b} < \tau, \\ \infty & \text{otherwise} \end{cases} \tag{5.11}$$

where a suitable limit can be estimated from the data. In this case, maximising the posterior amounts to a linear projection (giving the maximum likelihood solution), followed by truncation of the corresponding shape parameters to lie within the elliptical bounds.

5.1.1.5 Further Reading

Our experiments on 2D images suggest that a Gaussian distribution over shape parameters (implying a linear subspace of face shapes) is a good approximation as long as the training set contains only modest viewpoint variation. Nonlinear changes in shape such as introduced by large viewpoint variation [15], result in a linear subspace model that captures all of the required variation but in doing so also permits invalid shapes that lie off the nonlinear manifold [43].

To account explicitly for 3D head rotation (or viewpoint change), several studies have computed a 3D linear shape model and fitted it to new image data under perspective [5, 61] or orthogonal [43] projection. These approaches separate all rigid movement of the head from nonrigid shape deformation while maintaining some linearity for efficient computation (see Chap. 6 for more details on explicit 3D models).

Other studies avoid modelling nonlinearities explicitly, instead opting to use methods that estimate the parameters of the nonlinear manifold directly. Early examples outside of the face domain include polynomial regression [54] and mixture

models [8] whereas later studies applied nonlinear PCA [24], Kernel PCA [46], the Gaussian Process Latent Variable Model [32] and tensor-based models [36] to the face alignment problem.

5.1.2 Statistical Models of Texture

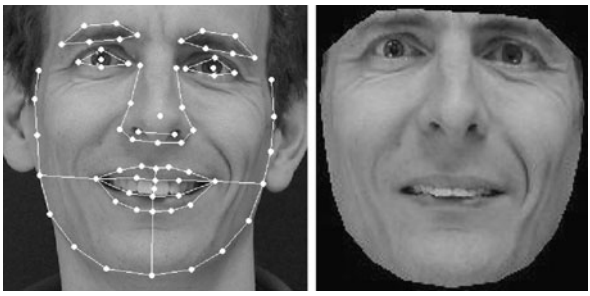
Though the *shape* of a face may give a weak indication of identity, the *texture* of the face provides a far stronger cue for recognition. We therefore apply similar techniques to those used to build a shape model in order to build a model of texture, given a set of training images. Fitting the texture model to new image data then summarising the properties of the underlying face (including its identity) through the texture model parameters.

5.1.2.1 Aligning Sets of Textures

Given a set of training images of faces that are coarsely-aligned (e.g., with respect to similarity transformations only), it has been shown that a linear subspace-based face model [34] provides a useful representation for recognition [58]. However, this coarse alignment does not compensate for nonrigid variation in shape due to identity, pose or expression. As a result, corresponding pixels over the set of training images actually originate from different points on the face (or possibly even the background) and spurious texture variation creeps into the model, reducing recognition performance [19].

To address this problem, we use the correspondences between facial *features* (e.g., eyes, nose and mouth) over the set of labelled training images to define an approximate correspondence between the *pixels* in the underlying image [4, 18]. In particular, we apply a continuous deformation—such as an interpolating spline or a piece-wise affine warp using a triangulation of the region—to warp each training image so that its feature points match a reference shape (typically the mean shape). The intensity information is then sampled from the *shape-normalised* image over the region covered by the mean shape (Fig. 5.3) to form a texture vector, $\mathbf{g}_{im}$. Since $\mathbf{g}_{im}$ is defined in the normalised shape frame, it has a fixed number of pixels, n_{pixels}, that is independent of the size of the object in the target image.

This nonlinear sampling (Algorithm 5.4) applies a *geometric* alignment of the textures, ensuring that corresponding elements over the set of texture vectors represent corresponding points on the face so that computed image statistics are meaningful. As in the case of the shape model, however, we want our texture model to represent only those changes that cannot be explained by a global transformation (e.g., due to changes in brightness and contrast). We therefore apply a *photometric* alignment of the texture samples before computing the image statistics that will define our texture model.



Original Image Warped to mean shape

Fig. 5.3 Example of face warped to the mean shape. Although the main shape variations due to smiling have been removed, there is considerable texture difference from a purely neutral face

Algorithm 5.4: Texture sampling

1: Precompute the positions of the sample pixels (typically all pixels in the region of interest) in the model reference frame, $(x_{s,i}, y_{s,i})$.
2: Construct a warping function, $W_{\mathbf{X}}(x, y)$ which maps the points of the mean shape onto the target points, $\mathbf{X}$.
3: For each element i in $\mathbf{g}_{im}$ sample, the target image at $W_{\mathbf{X}}(x_{s,i}, y_{s,i})$ using interpolation if appropriate.

More specifically, we express a texture in the image frame as a 1D affine transformation, $T_{\mathbf{u}}(\cdot)$, of the corresponding model texture, $\mathbf{g}$, such that

$$\mathbf{g}_{im} = T_{\mathbf{u}}(\mathbf{g}) = (1 + u_1) \cdot \mathbf{g} + u_2 \cdot \mathbf{1} \tag{5.12}$$

where $\mathbf{u} = (u_1, u_2)^{\mathrm{T}}$ is a vector of parameters corresponding to contrast and brightness, and $\mathbf{u} = \mathbf{0}$ gives the identity transformation. Unlike shape normalisation, this transformation is linear and we can find a closed-form solution for parameters to give the sampled vector, $\mathbf{g}_s$, zero sum and unit variance is the number of elements in the vectors. The normalised texture vector in the model frame is then given by the inverse transformation,:

$$u_2 = (\mathbf{g}_{im} \cdot \mathbf{1})/n_{\text{pixels}}, \tag{5.13}$$

$$1 + u_1 = |\mathbf{g}_{im}|^2/n_{\text{pixels}} - u_2^2 \tag{5.14}$$

where n_{pixels} is the number of elements in the vectors. The normalised texture vector in the model frame is then given by the inverse transformation, $\mathbf{g}_s = T_{\mathbf{u}}^{-1}(\mathbf{g}_{im})$:

$$\mathbf{g}_s = (\mathbf{g}_{im} - u_2 \cdot \mathbf{1})/(1 + u_1). \tag{5.15}$$

For colour images, each plane can be normalised separately though we have found that grey-scale models are able to generalise to unseen images more effectively than colour models.

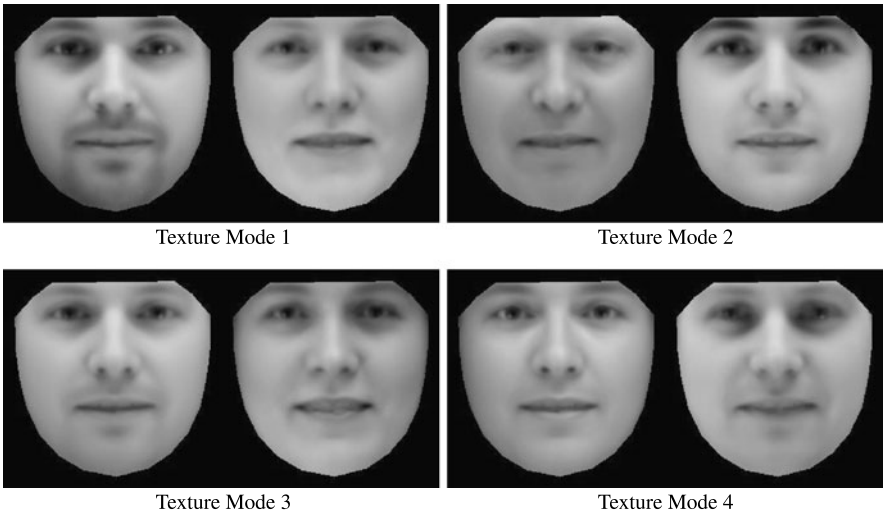


Fig. 5.4 Four modes of a face texture model built from 400 images (including neutral, smiling, frowning and surprised expressions) of 100 different individuals with around 20 000 pixels per example. Texture parameters have been varied by ±2 standard deviations from the mean

5.1.2.2 Linear Models of Texture Variation

Once we have compensated for the effects of brightness and contrast, we apply PCA to the set of normalised texture vectors to obtain a linear subspace model of texture,

$$\mathbf{g} \approx \bar{\mathbf{g}} + \mathbf{P}_g \mathbf{b}_g, \tag{5.16}$$

where $\bar{\mathbf{g}}$ is the mean texture over the training set, $\mathbf{P}_g$ is a set of orthogonal *modes of texture variation* and $\mathbf{b}_g$ is a vector of texture parameters. We can then generate a variety of plausible, shape-normalised face textures (Fig. 5.4) by varying $\mathbf{b}_g$ within limits learnt from the training set. Brightness and contrast variation can then be added by varying $\mathbf{u}$ and applying (5.12):

$$\mathbf{g}_{im} = (1 + u_1) \cdot (\bar{\mathbf{g}} + \mathbf{P}_g \mathbf{b}_g) + u_2 \cdot \mathbf{1}. \tag{5.17}$$

5.1.2.3 Choosing the Number of Texture Modes

As with the shape model, the simplest means of choosing the number of texture modes is to keep the smallest number of modes needed to capture a fixed proportion (e.g., 98%) of the total texture variation in the training set. Since the number of elements in the texture vector is typically much higher than in a shape vector, the texture model usually needs many more modes than the shape model to capture the same proportion of variance—278 modes were needed to capture 98% of the variance in our example (Fig. 5.4).

Algorithm 5.5: Fitting a texture model to new data

1: Compute the global transformation parameters, $\mathbf{u}$, using (5.13) and (5.14).
2: Compute the texture model parameters, $\mathbf{b}_g = \mathbf{P}_g^{\mathrm{T}}(T_{\mathbf{u}}^{-1}(\mathbf{g}_{im}) - \bar{\mathbf{g}})$.

5.1.2.4 Fitting the Model to New Textures

Like the shape model, fitting the texture model to new data proceeds in a two-step algorithm (Algorithm 5.5) and the model fitting to $\mathbf{g}_{im}$ is then given by (5.17). Unlike when fitting a shape model, however, no iteration is required for the texture model.

5.1.2.5 Further Reading

Though raw image intensities (or colour values) are adequate for most applications, modelling local image gradients may offer improved performance since gradients yield more information, are less sensitive to lighting and seem to favour edges over flat regions. If we compute local image gradients (g_x, g_y) via a straightforward linear transformation of the intensities, however, the subsequent Principal Component Analysis effectively reverses this transformation such that the basis images are almost identical to those obtained from raw intensities (apart from some boundary effects).

Instead, robust matching was demonstrated using a *non-linearly* normalised gradient at each pixel [10]: $(g'_x, g'_y) = (g_x, g_y)/(g + g_0)$ where g is the magnitude of the gradient, and g_0 is the mean gradient magnitude over a region. Other approaches have demonstrated improved performance by combining multiple feature bands such as intensity, hue and edge information [56], by including features derived from measures of 'cornerness' [53] and by learning filters that give smooth error surfaces [35].

Like shapes, textures also lie on a low-dimensional, nonlinear manifold embedded in the high-dimensional texture space [40]. As a result, linear methods such as PCA often cannot capture sufficient variance in the training set without also permitting invalid textures. If the training set is such that this becomes a problem (e.g., when significant viewpoint variation is present), multilinear [60] and nonlinear methods such as Locally Linear Embedding [47], IsoMap [57] and Laplacian Eigenmaps [3] may be useful (though probabilistic interpretation of such methods is nontrivial).

5.1.3 Combined Models of Appearance

The shape and texture of any example in a normalised frame can thus be summarised by the parameter vectors, $\mathbf{b}_s$ and $\mathbf{b}_g$, and though shape and texture may be considered independently [33], this can miss informative correlations between shape and

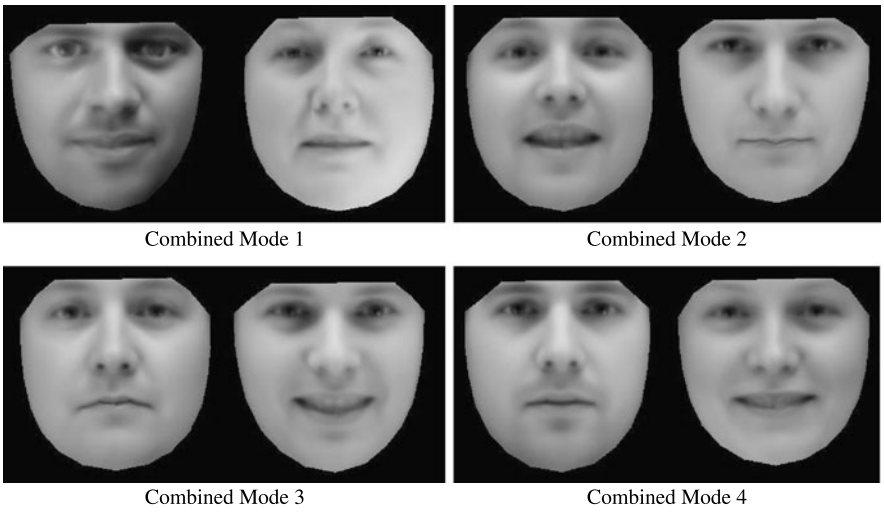


Fig. 5.5 Four modes of combined shape and texture model built from the same 400 face images as the texture-only model (Fig. 5.4). Combined parameters were varied by ±2 standard deviations from the mean

texture variations (e.g., square jaws correlating with facial hair). We therefore model these correlations by concatenating shape and texture parameter vectors into a single vector,

$$\mathbf{b} = \begin{pmatrix} \mathbf{W}_s \mathbf{b}_s \\ \mathbf{b}_g \end{pmatrix} = \begin{pmatrix} \mathbf{W}_s \mathbf{P}_s^{\mathrm{T}} (\mathbf{x} - \bar{\mathbf{x}}) \\ \mathbf{P}_g^{\mathrm{T}} (\mathbf{g} - \bar{\mathbf{g}}) \end{pmatrix}, \tag{5.18}$$

where $\mathbf{W}_s$ is a diagonal matrix of weights for each shape parameter, that accounts for the difference in units between the shape and texture models (see Sect. 5.1.3.1). We then apply a PCA on these combined vectors to give a model

$$\mathbf{b} \approx \mathbf{P}_c \mathbf{c} = \begin{pmatrix} \mathbf{P}_{cs} \\ \mathbf{P}_{cg} \end{pmatrix} \mathbf{c} \tag{5.19}$$

where $\mathbf{P}_c$ are the eigenvectors and $\mathbf{c}$ is a vector of *appearance* parameters (with zero-mean by construction) that jointly controls both the shape and texture of the model. Note that the linear nature of the model allows us to express the shape and grey-levels directly as functions of $\mathbf{c}$

$$\mathbf{x} = \bar{\mathbf{x}} + \mathbf{Q}_s \mathbf{c} \quad \text{where } \mathbf{Q}_s = \mathbf{P}_s \mathbf{W}_s^{-1} \mathbf{P}_{cs}, \tag{5.20}$$

$$\mathbf{g} = \bar{\mathbf{g}} + \mathbf{Q}_g \mathbf{c} \quad \text{where } \mathbf{Q}_g = \mathbf{P}_g \mathbf{P}_{cg}. \tag{5.21}$$

An example image can then be synthesised for a given $\mathbf{c}$ by generating the shape-free, grey-level image from the vector $\mathbf{g}$ and warping it using the control points described by $\mathbf{x}$ to give images that combine variations due to identity, lighting, viewpoint and expression (Fig. 5.5).

5.1.3.1 Choosing Shape Parameter Weights

In the combined model, the elements of $\mathbf{b}_s$ have units of distance whereas those of $\mathbf{b}_g$ have units of intensity. As a result, applying unweighted PCA to the concatenated parameter vectors may incorrectly place greater emphasis on capturing variation in one more than the other. To address this problem, we first scale the shape parameters via the weighting matrix, $\mathbf{W}_s$, so that the units of $\mathbf{b}_s$ and $\mathbf{b}_g$ are comparable.

A simple approach to choosing $\mathbf{W}_s$ is to set $\mathbf{W}_s = r\mathbf{I}$ where r^2 is the ratio of the total intensity variation to the total shape variation in the normalised frames. A more systematic approach is to measure the effect of varying $\mathbf{b}_s$ on the sample $\mathbf{g}$ by displacing each element of $\mathbf{b}_s$ from its optimum value for each training example and sampling the image given the displaced shape; the RMS change in $\mathbf{g}$ per unit change in shape parameter b_s gives the weight w_s to be applied to that parameter in (5.18). In practice, however, we have found that synthesis and search algorithms are relatively insensitive to the choice of $\mathbf{W}_s$.

5.1.3.2 Separating Sources of Variability

In many applications, some sources of appearance variation are more useful than others. In face recognition, for example, variations due to identity are essential whereas variations due to other sources (e.g., expression) are a nuisance whose effects we want to minimise. Since the combined appearance model mixes these two sources, each element of the parameter vector encodes both between- and within-identity variation. The sources can, however, be separated by splitting the subspace defined by $\mathbf{P}_c$ into two orthogonal subspaces,

$$\mathbf{c} = \mathbf{P}_b\mathbf{c}_b + \mathbf{P}_w\mathbf{c}_w, \tag{5.22}$$

where $\mathbf{P}_b$ and $\mathbf{c}_b$ encode between-identity variation, and $\mathbf{P}_w$ and $\mathbf{c}_w$ encode within-identity variation [24].

Computing the within-identity subspace is straightforward if we know the identity of the person in every training image—the columns of $\mathbf{P}_w$ are the eigenvectors of the covariance matrix computed using the deviation of each $\mathbf{c}$ from the mean appearance vector *for the same identity*. Varying $\mathbf{c}_w$ then indirectly changes $\mathbf{c}$ and thus the appearance of the face but only in ways that a specific individual's face can change, such as expression (Fig. 5.6, top).

The orthogonal subspace, $\mathbf{P}_b$, that represents between-identity variation can then be computed by subtracting the within-identity variation, $\mathbf{P}_w\mathbf{P}_w^{\mathrm{T}}\mathbf{c}$, and doing PCA over the resulting appearance parameter vectors. Alternatively, the between-class covariance matrix can be computed using the set of identity-specific means, though the mean is not guaranteed to be free of corruption by some non-neutral expression or head pose (Fig. 5.6, bottom). Iterative methods have also shown success in separating different sources of variability [16].

In contrast, tensor-based methods keep sources of variability separate at all times by building a multilinear model of texture variation [36, 60]. These methods, however, have strict requirements in terms of training data—namely, every combination

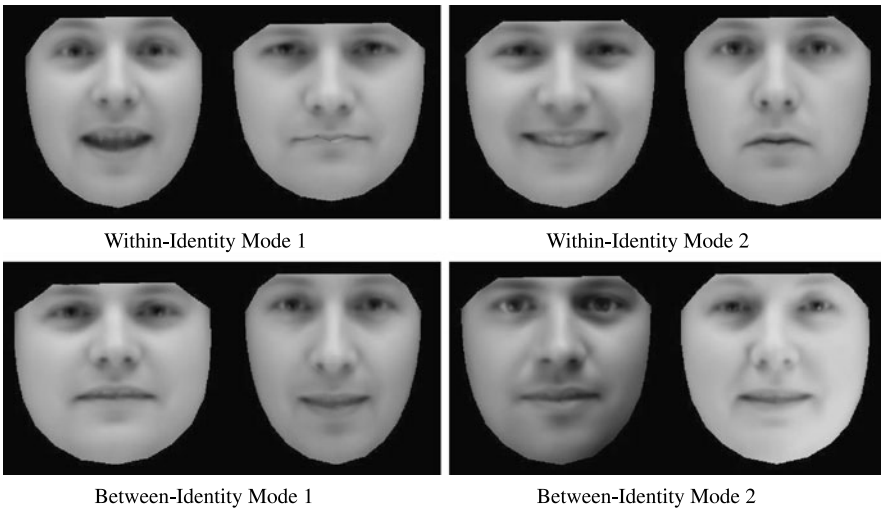


Fig. 5.6 (*Top*) Two within-identity modes of individual face variation; (*bottom*) two between-identity modes of variation between individuals. Some residual variation in expression is present due to not every mean face being completely neutral

of variation must be present in the training set (that is, every expression at every pose for every identity).

5.2 Active Shape Models (ASMs)

Once we have built a statistical shape model from labelled training images, we need a method of matching the model to an unseen image of the face so that we can interpret the underlying properties of the image. One method, known as the *Active Shape Model* (ASM) [11], does this by alternating between locally searching for features to maximise a 'goodness of fit' measure and regularising the located shape to filter out spurious local matches caused by noisy data.

5.2.1 Goodness of Fit

Given a set of shape parameter values, $\mathbf{b}_s$, and pose parameters, $\mathbf{t}$, we can define the shape of the object in the image frame. If we also define a measure of how well given parameters explain the observed image data, we can find 'better' parameter values by searching in a local region around each feature point to find alternative feature locations that match the model more closely. In general, we can model appearance with a 2D patch centred at the feature location and search a 2D region of interest around the current estimate for better matches (see Sect. 5.2.5).

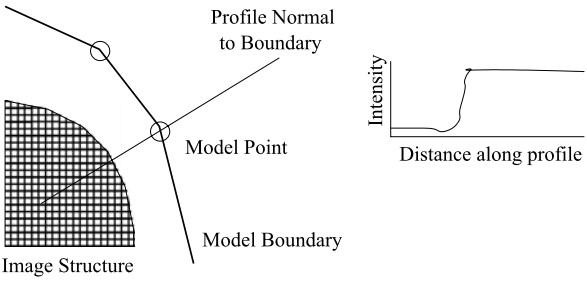


Fig. 5.7 At each model point, we sample along a profile normal to the boundary

In the specific case of the Active Shape Model [11], however, we reduce computational demands by looking along 1D linear profiles that pass through each model point and are normal to the model boundary (Fig. 5.7). If we assume that the model boundary corresponds to an edge, the strongest edge along the profile suggests a new location for the model point. Model points, however, are not always found on the strongest edge in the locality—they may instead be associated with a weaker secondary edge or some other image structure—and so instead we learn from the training set what to look for in the target image.

One popular method is to build a statistical model of the grey-level structure along the profile, normal to the boundary in the training set. Suppose for a given point we sample along a profile k pixels either side of the model point in the ith training image. We then have $2k+1$ samples which can be put in a vector $\mathbf{g}_i$. To avoid the effects of a constant offset in the intensities (that is, differences in brightness), we sample the derivative along the profile rather than the absolute grey-level values. We similarly compensate for changes in contrast by dividing through by the sum of absolute element values such that

$$\mathbf{g}_i \rightarrow \frac{1}{\sum_j |g_{ij}|}\mathbf{g}_i. \tag{5.23}$$

We repeat this for every training image to get a set of normalised samples, $\{\mathbf{g}_i\}$, whose distribution we can then model. If we assume that these profile samples have a multivariate Gaussian distribution, for example, we can build a statistical model of the grey-level profiles by computing their mean, $\bar{\mathbf{g}}$, and covariance, $\mathbf{S}_g$. The quality of fit of a new sample, $\mathbf{g}_s$, to the model is then given by the Mahalanobis distance of the sample from the model mean,

$$f(\mathbf{g}_s) = (\mathbf{g}_s - \bar{\mathbf{g}})^{\mathrm{T}}\mathbf{S}_g^{-1}(\mathbf{g}_s - \bar{\mathbf{g}}), \tag{5.24}$$

and is related to the negative log of the probability that $\mathbf{g}_s$ is drawn from the learned distribution such that minimising $f(\mathbf{g}_s)$ is equivalent to finding the maximum likelihood solution.

In practice, when performing a local search for a given feature point we first sample a profile of $m > k$ pixels either side of the current estimate. We then test the quality of fit of the corresponding grey-level model to each of the $2(m-k)+1$ possible positions along the sample and choose the one which gives the best match

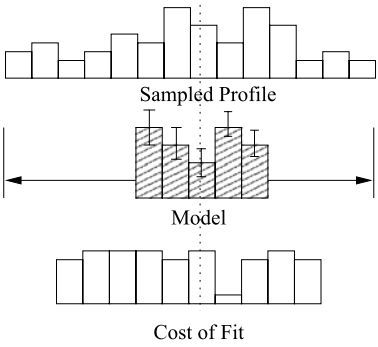


Fig. 5.8 Search along sampled profile to find best fit of grey-level model

Algorithm 5.6: Active Shape Model (ASM) fitting

1: **repeat**
2: Examine a region of the image around each point, $\mathbf{X}_i$, to find the best nearby match, $\mathbf{X}'_i$ (see Sect. 5.2.1).
3: Update the parameters $(\mathbf{t}, \mathbf{b}_s)$ to fit the shape model to the newly found local matches $\mathbf{X}'$ (see Sect. 5.1.1.4).
4: **until** converged (that is, change in regularised estimate is sufficiently small).

(as shown in Fig. 5.8) that is, the lowest value of $f(\mathbf{g}_s)$. Repeating this for each feature point gives a new estimate for the shape of the face.

5.2.2 *Iterative Model Refinement*

Local searches on their own, however, are prone to spurious matches due to noisy data and unmodelled image properties. To ensure that the estimated shape agrees with the statistical model learned from training data (see Sect. 5.1.1), we regularise our solution by fitting the shape model to the local matches. Hopefully, this regularised estimate is closer to the true solution such that repeating the search-regularise cycle gives progressively better estimates (Algorithm 5.6). Since each point also has a quality of match score, given by (5.24), these scores can be used to weight points differently during model fitting, as in (5.8), according to our belief in their reliability [30].

5.2.3 *Multi-Resolution Active Shape Models*

To avoid local minima when searching the image, it is useful to smooth the error function in early stages and reduce the level of smoothing gradually with each

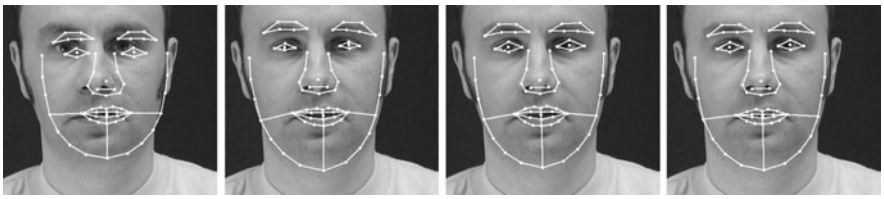


Fig. 5.9 Successful search for a face using the Active Shape Model

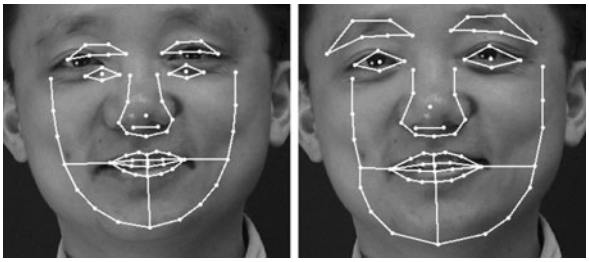


Fig. 5.10 Failure of the Active Shape Model to localise a face where the search profiles are not long enough to locate the edges of face

iteration. In practice, we apply this smoothing by implementing the ASM in a multi-resolution framework using a Gaussian image pyramid. This involves first searching for the object in a coarse image, then refining the shape in a series of progressively finer resolution images. Not only is this more robust to local minima but also more efficient, since less complex models can be used at the coarse levels of the pyramid.

5.2.4 Examples of ASM Search

In one example of an ASM search to locate the features of a face (Fig. 5.9), we place the model instance near the centre of the image and perform a coarse to fine search, starting on the 3rd level of a Gaussian pyramid ($1/8$ the resolution in x and y compared to the original image). In the first few iterations, large improvements are made that get the position and scale roughly correct. As the search progresses, however, more subtle adjustments to the shape are made using the finer resolution images. After 18 iterations (with at most 10 iterations per pyramid level), the process has converged and gives a good match to the target image.

In another example, the ASM fails to localise the face (Fig. 5.10). This is most likely due to the initialisation being too far from the true solution such that the correct feature positions are beyond the scope of the local search and the process falls into a local minimum.

5.2.5 *Further Reading*

The Active Shape Model can be viewed as a specific example of a 'constrained local model' (CLM)—a class of algorithms that perform a local search for each feature (based on an independent set of learned texture models) then fit a learned shape model to the set of local matches. Addressing susceptibility to local minima, however, has been a driving force for various modifications to the match metric and search algorithm.

Although profile gradients have proven to be effective for local search, discriminative models of profile intensity can distinguish between correct and incorrect matches and improve performance further [59]. Better still, using 2D patches instead of 1D profiles makes the model even more discriminative [44], based on measures such as normalised correlation [21], boosted classification [20] or mixtures of linear experts [50] to define a match score. Where to look for potential matches is usually defined by hand (e.g., a rectangular or elliptical grid) but may also be learned from training data [38].

Once a response surface (that is, the set of match scores for all candidate locations) has been computed for each point, the ASM naïvely picks the best match for each point before projecting the set of matches back onto the subspace of permitted shapes. Effectively, this approximates each response surface by a Gaussian likelihood function with diagonal covariance; by including off-diagonal terms, we can model directional uncertainty and can further improve performance [41, 45]. If the response surface is not approximated by a parametric function, or is approximated by a complex function such as a mixture of Gaussians [28] or nonparametric kernel density estimate [51], the match function may be optimised using iterative methods such as gradient-free optimisers such as the Nelder–Mead Simplex method [21] or mean-shift [51].

When using a PCA model of shape, each feature imposes constraints on every other feature such that computational limitations force us to select the match for each point independently of all other points. By assuming conditional independence between features, however, we can reduce the complexity of the graph and use Markov Random Field methods at little or no cost in efficiency [37]. Simplifying the graph in this way allows us to consider multiple candidates for each feature point and therefore increase robustness by avoiding local minima due to spurious matches that do not agree with the possible matches for other feature points. When choosing which dependencies to eliminate, trees [25] and k-fans [17] are popular due to their simplicity though more effective graph structures may be learned from training data [29].

5.3 Active Appearance Models (AAMs)

One criticism of the approaches related to the ASM is that they use only sparse local information around the points of interest. In addition, they often treat the information at each point as independent which is rarely the case. These criticisms

Algorithm 5.7: Image residual computation

1: Use (5.21) to compute the shape, $\mathbf{x}$, and texture, $\mathbf{g}_m$, in the normalised reference frame.
2: Compute the shape in the image frame by applying $\mathbf{X} = S_{\mathbf{t}}(\mathbf{x})$.
3: Sample the target image in the region defined by $\mathbf{X}$ to get $\mathbf{g}_{im}$ (see Sect. 5.1.2.1).
4: Normalise the sampled texture with respect to brightness and contrast using $\mathbf{g}_s = T_{\mathbf{u}}^{-1}(\mathbf{g}_{im})$.
5: Compute the residual, $\mathbf{r}(\mathbf{p}) = \mathbf{g}_s - \mathbf{g}_m$.

are largely addressed by the following approach—dubbed the *Active Appearance Model* (AAM) [14]—that uses a combined model of appearance (Sect. 5.1.3) for image interpretation via the *interpretation through synthesis* paradigm: if we can find appearance model parameters which synthesise a face very similar to that in the target image, those parameters summarise the shape and texture of the face and can therefore be used directly for interpretation. In contrast to the Active Shape Model, the Active Appearance Model directly *predicts* incremental updates to appearance parameters from image residuals rather than performing a local search, making the method very efficient.

5.3.1 Goodness of Fit

Given combined appearance model parameters, $\mathbf{c}$, a set of pose parameters, $\mathbf{t}$, and a set of texture normalisation parameters, $\mathbf{u}$, we can concatenate the parameters into a single vector, $\mathbf{p} = (\mathbf{c}^{\mathrm{T}}|\mathbf{t}^{\mathrm{T}}|\mathbf{u}^{\mathrm{T}})^{\mathrm{T}}$, synthesise a new face image and compute the residual, $\mathbf{r}(\mathbf{p}) = \mathbf{g}_s - \mathbf{g}_m$, with respect to the observed data. We then assess the quality of the synthesis (Algorithm 5.7) by some function of $\mathbf{r}(\mathbf{p})$, such as the sum of squared error,

$$E_{\mathrm{sse}}(\mathbf{p}) = \left|\mathbf{r}(\mathbf{p})\right|^2 = \mathbf{r}(\mathbf{p})^{\mathrm{T}}\mathbf{r}(\mathbf{p}), \tag{5.25}$$

as used in our examples. Like the ASM, we also can make assumptions about the distributions of residuals to estimate $p(\mathbf{r} \mid \mathbf{p})$ and place the matching in a Bayesian framework [9].

5.3.2 Updating Model Parameters

Given one estimate of the parameters, $\mathbf{p} = \mathbf{p}^* + \delta\mathbf{p}$ (where $\delta\mathbf{p}$ is our displacement from the true solution, $\mathbf{p}^*$), and the corresponding residual, $\mathbf{r}(\mathbf{p})$, we then want to modify the parameters by $\delta\mathbf{p}$ to minimise $|\mathbf{r}(\mathbf{p} - \delta\mathbf{p})|^2$. Though we could do this

via gradient descent [33], the AAM instead assumes that $\delta\mathbf{p}$ can be predicted linearly from the residual vector such that $\delta\mathbf{p} = \mathbf{R}\mathbf{r}(\mathbf{p})$. In this section, we present two approaches to learning the matrix $\mathbf{R}$ from training data that consists of random parameter displacements, $\{\delta\mathbf{p}\}$ (stored in the columns of a matrix, $\mathbf{C}$), and the corresponding residuals, $\{\mathbf{r}(\mathbf{p}^*+\delta\mathbf{p})\}$ (stored in the columns of a matrix, $\mathbf{V}$).

Since we want the model to be independent of the background in the training images, perturbed texture samples that include pixels from the background must be accounted for when building the model. One approach is to remove background pixels from the update model though we use the simpler alternative of setting background pixels to some random value.

5.3.2.1 Estimating R via Linear Regression

Given parameter displacements, $\mathbf{C}$, and the corresponding image residuals, $\mathbf{V}$, a linear update relationship gives

$$\mathbf{C} = \mathbf{R}\mathbf{V} \quad \Rightarrow \quad \mathbf{R} = \mathbf{C}\mathbf{V}^{\dagger} \tag{5.26}$$

where $\mathbf{V}^{\dagger}$ is the pseudo-inverse of $\mathbf{V}$ [12].

Unless, however, there are more displacements than pixels modelled (a rare occurrence) the model will overfit to the training data. To address this problem, applying PCA to reduce the dimensionality of the residuals (and effectively increase sampling density) before performing the regression has been shown to reduce overfitting and improve performance [31]. Alternatively, rather than projecting onto a lower dimensional subspace that maximises the variance of the projected inputs (that is, image residuals), Canonical Component Analysis (CCA) improves performance further [22] by computing subspaces for both inputs and outputs (that is, parameter displacements) that maximises the correlation between their respective projections.

5.3.2.2 Estimating R via Gauss–Newton Approximation

An alternative way to avoid overfitting is suggested by the first order Taylor expansion,

$$\mathbf{r}(\mathbf{p} - \delta\mathbf{p}) = \mathbf{r}(\mathbf{p}) - \frac{\partial \mathbf{r}}{\partial \mathbf{p}}\delta\mathbf{p}, \tag{5.27}$$

where the ijth element of the matrix $\frac{\partial \mathbf{r}}{\partial \mathbf{p}}$ is $\frac{dr_i}{dp_j}$ such that $|\mathbf{r}(\mathbf{p} - \delta\mathbf{p})|^2$ is minimised with respect to $\delta\mathbf{p}$ by the RMS solution,

$$\delta\mathbf{p} = \mathbf{R}\mathbf{r}(\mathbf{p}) \quad \text{where } \mathbf{R} = \left(\frac{\partial \mathbf{r}}{\partial \mathbf{p}}^{\mathrm{T}} \frac{\partial \mathbf{r}}{\partial \mathbf{p}}\right)^{-1} \frac{\partial \mathbf{r}}{\partial \mathbf{p}}^{\mathrm{T}}. \tag{5.28}$$

In a standard optimisation scheme, we would recalculate $\frac{\partial \mathbf{r}}{\partial \mathbf{p}}$ at every step—a computationally expensive operation. Since it is being computed in a normalised reference frame, however, we assume that it is approximately fixed and can be precomputed from the training set [14]. In practice, we express (5.27) in terms of the training data, $\mathbf{C}$ and $\mathbf{V}$, to give

$$\frac{\partial \mathbf{r}}{\partial \mathbf{p}} = \arg\min_{\mathbf{J}} \|\mathbf{V} - \mathbf{J}\mathbf{C}\|_{\mathrm{F}}^2 \quad \Rightarrow \quad \frac{\partial \mathbf{r}}{\partial \mathbf{p}} = \mathbf{V}\mathbf{C}^{\dagger} \tag{5.29}$$

where $\|\cdot\|_{\mathrm{F}}$ denotes the Frobenius norm. This Gauss–Newton approximation is popular because computing the pseudoinverse $\mathbf{C}^{\dagger}$ is usually quicker and more robust than computing $\mathbf{V}^{\dagger}$ due to their relative sizes. We then precompute $\mathbf{R}$ via (5.28) and use it in all subsequent image searches. To ensure a reliable estimate, we measure residuals at displacements of differing magnitudes (typically up to 0.5 standard deviations of each parameter) and combine them by smoothing with a Gaussian kernel. Qualitatively, computing the update via a Gauss-Newton approximation should be more stable, has a clearer mathematical interpretation and allows extra constraints to be incorporated easily [9]. Quantitatively, however, tests comparing the different approaches [7] have shown that using linear regression gives better localisation performance.

5.3.3 Iterative Model Refinement

Given an initial estimation of the model parameters, $\mathbf{c}$, the pose, $\mathbf{t}$, and the texture transformation, $\mathbf{u}$, we repeatedly apply (5.28) to update model parameters based on the measured residual, $\mathbf{r}$, giving estimates that get progressively closer to the true solution (Algorithm 5.8).

When we update the parameter vector $\mathbf{p} = (\mathbf{c}^{\mathrm{T}}|\mathbf{t}^{\mathrm{T}}|\mathbf{u}^{\mathrm{T}})^{\mathrm{T}}$, the simplest approach is to subtract a the predicted displacement $\delta\mathbf{p} = (\delta\mathbf{c}^{\mathrm{T}}|\delta\mathbf{t}^{\mathrm{T}}|\delta\mathbf{u}^{\mathrm{T}})^{\mathrm{T}}$ such that $\mathbf{p} \rightarrow \mathbf{p} - \delta\mathbf{p}$. The update step, however, estimates corrections in the *model* frame which must then be projected into the image frame using the current pose and texture transformations. Strictly speaking, therefore, we should update the parameters controlling the pose, $\mathbf{t}$, and texture transformation, $\mathbf{u}$, by *composing* the resulting transformations (during both training and image search). In other words, we should compute pose parameters $\mathbf{t}'$ such that $S_{\mathbf{t}'}(\mathbf{x}) = S_{\mathbf{t}}(S_{\delta\mathbf{t}}(\mathbf{x}))$ and new texture transformation parameters $\mathbf{u}'$ such that $T_{\mathbf{u}'}(\mathbf{g}) = T_{\mathbf{u}}(T_{\delta\mathbf{u}}(\mathbf{g}))$ where updates are applied in the model frame before transforming to the image frame.

5.3.4 Multi-Resolution Active Appearance Models

As in the Active Shape Model, we estimate the appearance models and update matrices at a range of image resolutions using a Gaussian image pyramid. We can then

Algorithm 5.8: Active Appearance Model (AAM) fitting

Calculate the image points, $\mathbf{X}$, and model frame texture, $\mathbf{g}_m$.
Sample the image to get $\mathbf{g}_{im}$
Normalise with respect to brightness and contrast using $\mathbf{g}_s = T_{\mathbf{u}}^{-1}(\mathbf{g}_{im})$.
Compute the residual, $\mathbf{r} = \mathbf{g}_s - \mathbf{g}_m$, and corresponding error, $E_{\text{sse}} = |\mathbf{r}|^2$.
repeat
  Predict the displacement from the true model parameters, $\delta\mathbf{p} = \mathbf{R}\mathbf{r}(\mathbf{p})$.
  Set $k = 1$.
  repeat
    Compute the updated model parameters, $\mathbf{p}' = \mathbf{p} - k \cdot \delta\mathbf{p}$. STATE Calculate the new points, $\mathbf{X}'$, and model frame texture, $\mathbf{g}'_m$.
    Sample the image at the new points to get $\mathbf{g}'_{im}$
    Normalise with respect to brightness and contrast using $\mathbf{g}'_s = T_{\mathbf{u}'}^{-1}(\mathbf{g}'_{im})$.
    Calculate a new residual vector, $\mathbf{r}' = \mathbf{g}'_s - \mathbf{g}'_m$, and corresponding error, $E'_{\text{sse}} = |\mathbf{r}'|^2$.
    Set $k = k/2$
  until $E'_{\text{sse}} < E_{\text{sse}}$
  Set $\mathbf{p} = \mathbf{p}'$, $\mathbf{r} = \mathbf{r}'$ and $E_{\text{sse}} = E'_{\text{sse}}$.
until converged ($E_{\text{sse}} <$ threshold) or maximum number of iterations exceeded

use a multi-resolution search algorithm in which we start at a coarse resolution and iterate to convergence at each level before projecting the current solution to the next level of the model [33]. This is more efficient and can converge to the correct solution from further away than search at a single resolution. Computationally, the complexity of the AAM at a given level is $O(n_{\text{modes}} \cdot n_{\text{pixels}})$ since each iteration samples n_{pixels} points from the image then multiplies by a $n_{\text{modes}} \times n_{\text{pixel}}$ matrix.

5.3.5 *Examples of AAM Search*

When using an AAM to localise a face in a previously unseen image, the algorithm typically requires fewer than 20 iterations to converge to a faithful reproduction of the face (Fig. 5.11). Like the ASM, however, the AAM is prone to local minima if started too far from the true solution (Fig. 5.12).

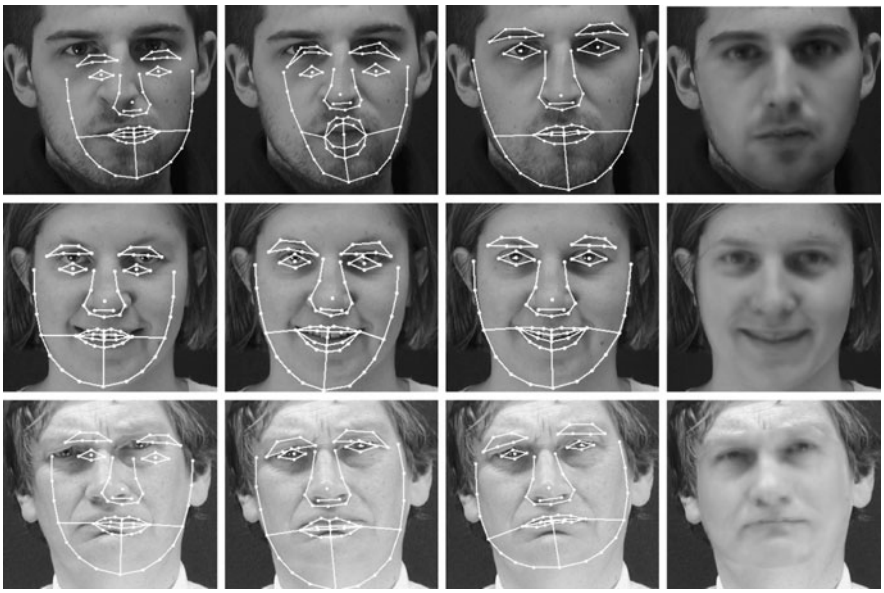


Fig. 5.11 Search using the Active Appearance Model on faces not in the training set, showing evolution of the shape and the final image reconstruction. Initial iterations are performed using a low resolution model and resolution increases as the search progresses

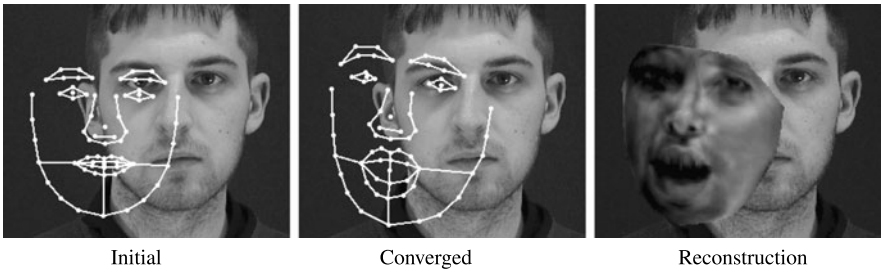


Fig. 5.12 Example of AAM search failure where the initialisation was too far from true position. The model has matched the eye and eyebrow to the wrong side of the face, and attempted to explain the dark background by shading one side of the reconstructed face

5.3.6 Alternative Strategies

Following the Active Appearance Model, a variety of related approaches to matching models of shape and texture have been suggested. Here, we summarise some of the key contributions.

5.3.6.1 Shape AAM

Though combining shape and appearance parameters has its uses in capturing correlations, treating the parameters separately can have computational benefits. Consider the case where we use the residuals to update only the pose, $\mathbf{t}$, and shape model parameters, $\mathbf{b}_s$, such that

$$\delta\mathbf{t} = \mathbf{R}_t\mathbf{r} \quad \text{and} \quad \delta\mathbf{b}_s = \mathbf{R}_s\mathbf{r}, \tag{5.30}$$

where the model texture, $\mathbf{g}_m$, is now simply the projection of the normalised sample, $\mathbf{g}_s$, onto the texture subspace (since shape and texture are treated independently). In this case,

$$\begin{aligned}\delta\mathbf{b}_s &= \mathbf{R}_s\big(\mathbf{g}_s - \big(\bar{\mathbf{g}} + \mathbf{P}_g\mathbf{P}_g^{\mathrm{T}}(\mathbf{g}_s - \bar{\mathbf{g}})\big)\big) &(5.31)\\ &= \mathbf{R}_s\big(\big(\mathbf{I} - \mathbf{P}_g\mathbf{P}_g^{\mathrm{T}}\big)(\mathbf{g}_s - \bar{\mathbf{g}})\big) &(5.32)\\ &= \mathbf{R}'_s\mathbf{g}_s - \mathbf{g}_0 &(5.33)\end{aligned}$$

where $\mathbf{R}'_s = \mathbf{R}_s(\mathbf{I} - \mathbf{P}_g\mathbf{P}_g^{\mathrm{T}})$ and $\mathbf{g}_0 = \mathbf{R}_s(\mathbf{I} - \mathbf{P}_g\mathbf{P}_g^{\mathrm{T}})\bar{\mathbf{g}}$ can be precomputed such that the texture model is required only to compute the texture error for the purposes of detecting convergence. Using a fixed number of iterations or changes in the shape parameters, however, dispenses with the texture model altogether and results in a much faster (though less accurate) algorithm. If required, a combined model of shape and texture can be used to apply *post hoc* constraints to the relative shape and texture parameter vectors by projecting them into the combined appearance space. This approach, known as the 'Shape AAM' [13], is closely related to the 'Active Blob' method [52] that uses an elastic deformation model rather than a statistical model of shape.

5.3.6.2 Compositional Approach

As noted earlier (Sect. 5.3.3), pose and texture transformation parameters should be updated via composition (rather than addition) and it can be shown that there are benefits from updating shape parameters in the same way [42]. If we consider (5.4) as a parameterised transformation of the mean shape, $\mathbf{x} = U_{\mathbf{b}}(\bar{\mathbf{x}})$, then we need to find parameters, $\mathbf{b}'$, such that $U_{\mathbf{b}'}(\bar{\mathbf{x}}) = U_{\mathbf{b}}(U_{\delta\mathbf{b}}(\bar{\mathbf{x}}))$, for example by approximating the transformation with a thin-plate spline (Algorithm 5.9). Using the *inverse compositional* image alignment algorithm [1] improves efficiency further by specifying Jacobians and Hessians as functions of template images (rather than sampled images) such that they can be precomputed, thus saving computation at run-time. Also decoupling shape from texture for efficiency, the resulting inverse compositional AAM [42] has demonstrated model fitting at speeds of up to 200 frames per second.

Algorithm 5.9: Compositional AAM fitting with a Thin Plate Spline [6]

1: Compute the thin plate spline, $T_{\text{tps}}(.)$, that maps the points $\bar{\mathbf{x}}$ to $\mathbf{x}$
2: Compute the modified mean points, $\mathbf{x}_\delta = \bar{\mathbf{x}} + \mathbf{P}_s\delta$
3: Apply the transformation, $\mathbf{x}' = T_{\text{tps}}(\mathbf{x}_\delta)$
4: Find the shape parameters which best match, $\mathbf{b}'_s = \mathbf{P}_s^{\mathrm{T}}(\mathbf{x}' - \bar{\mathbf{x}})$

5.3.7 Further Reading

Since their introduction, Active Appearance Models have spawned many variants [26] and also demonstrated considerable success in medical image analysis (for which, software is publicly available [55]). In addition to the two variants already described (Sect. 5.3.6), other modifications include methods for expressing the update matrix, $\mathbf{R}$, as a function of the current residual for improved convergence [2] and sequential implementations that tune the training data to match the expected error distribution [49].

Predicting parameter updates via nonlinear regression has also been proposed, where boosting a number of weak regressors is currently popular [48, 63]. Using gradient descent-based algorithms to minimise an error metric learned from training data has also shown promise [39], as has selecting updates via a pairwise comparison of two potential candidates [62].

5.4 Conclusions

In this chapter, we have described powerful statistical models of the shape and texture of faces that are capable of synthesising a wide range of convincing face images. Algorithms such as the Active Shape Model (ASM) and Active Appearance Model (AAM) rapidly fit these appearance models to unseen image data such that the parameters capture the underlying properties of the face, isolating those sources of variation that are essential to face recognition (that is, identity) from those that are not (e.g., expression).

One weakness of both the ASM and AAM (and their variations) is that they are local optimisation techniques and tend to fall into local minima if initialisation is poor. Where independent estimates of feature point positions are available (e.g., from an eye tracker) these can be incorporated into the matching schemes and lead to more reliable matching [9].

These approaches also rely on an annotated corpus of training data and therefore can only deal effectively with certain types of variation in appearance. For example, person-specific variation that cannot be corresponded (e.g., wrinkles on the forehead or the appearance of moles) tends to get blurred out by the averaging process inherent in the modelling. This suggests that these methods may be improved by adding further layers of information to the model in order to represent individual differences which are poorly represented as a result of pooling in the current models.

Open questions (some of which are currently under investigation) include:

- How do we obtain accurate correspondences across the training set?
- What is the optimal choice of model size and number of model modes?
- How should image structure be represented?
- What is the best method of matching the model to the image?
- How do we avoid local minima in the error surface?

Acknowledgements The authors would like to thank their numerous colleagues who have contributed to the research summarised in this chapter, including C. Beeston, F. Bettinger, D. Cooper, D. Cristinacce, G. Edwards, A. Hill, J. Graham, H. Kang, P. Kittipanya-ngam and M. Roberts.

References

1. Baker, S., Matthews, I.: Lucas–Kanade 20 years on: A unifying framework. Part I: The quantity approximated, the warp update rule and the gradient descent approximation. Int. J. Comput. Vis. (2004)
2. Batur, A.U., Hayes, M.H.: Adaptive active appearance models. IEEE Trans. Med. Imaging **14**(11), 1707–1721 (2005)
3. Belkin, M., Nigoyi, P.: Laplacian eigenmaps for dimensionality reduction and data representation. Neural Comput. **15**, 1373–1396 (2003)
4. Benson, P.J., Perrett, D.I.: Synthesizing continuous-tone caricatures. Image Vis. Comput. **9**, 123–129 (1991)
5. Blanz, V., Vetter, T.: Face recognition based on fitting a 3D morphable model. IEEE Trans. Pattern Anal. Mach. Intell. (2003)
6. Bookstein, F.L.: Principal warps: Thin-plate splines and the decomposition of deformations. IEEE Trans. Pattern Anal. Mach. Intell. **11**(6), 567–585 (1989)
7. Cootes, T.F., Kittipanya-ngam, P.: Comparing variations on the active appearance model algorithm. In: 13th British Machine Vision Conf., vol. 2, pp. 837–846, September 2002
8. Cootes, T., Taylor, C.J.: A mixture model for representing shape variation. Image Vis. Comput. **17**(8), 567–574 (1999)
9. Cootes, T.F., Taylor, C.J.: Constrained active appearance models. In: 8th Int'l Conf. on Comp. Vis., vol. 1, pp. 748–754, July 2001. IEEE Computer Society Press, Los Alamitos (2001)
10. Cootes, T.F., Taylor, C.J.: On representing edge structure for model matching. Comput. Vis. Pattern Recognit. **1**, 1114–1119 (2001)
11. Cootes, T.F., Taylor, C.J., Cooper, D., Graham, J.: Active shape models—their training and application. Comput. Vis. Image Underst. **61**(1), 38–59 (1995)
12. Cootes, T.F., Edwards, G.J., Taylor, C.J.: Active appearance models. In: Burkhardt, H., Neumann, B. (eds.) 5th European Conf. on Comp. Vis., vol. 2, pp. 484–498. Springer, Berlin (1998)
13. Cootes, T.F., Edwards, G.J., Taylor, C.J.: A comparative evaluation of active appearance model algorithms. In: British Machine Vision Conf., vol. 2, pp. 680–689, September 1998
14. Cootes, T., Edwards, G., Taylor, C.: Active appearance models. IEEE Trans. Pattern Anal. Mach. Intell. **23**(6), 681–685 (2001)
15. Cootes, T.F., Wheeler, G.V., Walker, K.N., Taylor, C.J.: View-based active appearance models. Image Vis. Comput. **20**, 657–664 (2002)
16. Costen, N., Cootes, T.F., Taylor, C.J.: Compensating for ensemble-specificity effects when building facial models. Image Vis. Comput. **20**, 673–682 (2002)
17. Crandall, D., Felzenszwalb, P., Huttenlocher, D.: Spatial priors for part-based recognition using statistical models. In: Proc. IEEE Conf. on Comp. Vis. and Patt. Recog., vol. 1 (2005)

18. Craw, I., Cameron, P.: Parameterising images for recognition and reconstruction. In: 2nd British Machine Vision Conf., pp. 367–370. Springer, London (1991)
19. Craw, I., Cameron, P.: Face recognition by computer. In: Hogg, D., Boyle, R. (eds.) 3rd British Machine Vision Conf., pp. 489–507. Springer, London (1992)
20. Cristinacce, D., Cootes, T.: Facial feature detection using AdaBoost with shape constraints. In: Proc. British Machine Vision Conf. (2003)
21. Cristinacce, D., Cootes, T.F.: Automatic feature localisation with constrained local models. Pattern Recognit. **41**, 3054–3067 (2008)
22. Donner, R., Reitner, M., Langs, G., Peloschek, P., Bischof, H.: Fast active appearance model search using canonical correlation analysis. IEEE Trans. Pattern Anal. Mach. Intell. **28**(10), 1690–1694 (2006)
23. Dryden, I., Mardia, K.V.: The Statistical Analysis of Shape. Wiley, London (1998)
24. Edwards, G.J., Lanitis, A., Taylor, C.J., Cootes, T.F.: Statistical models of face images—improving specificity. Image Vis. Comput. **16**(3), 203–211 (1998)
25. Felzenszwalb, P., Huttenlocher, D.: Pictorial structures for object recognition. Int. J. Comput. Vis. **61**(1), 55–79 (2005)
26. Gao, X., Su, Y., Li, X., Tao, D.: A review of active appearance models. IEEE Trans. Syst. Man Cybern., Part C, Appl. Rev. **40**(2), 145–158 (2010)
27. Goodall, C.: Procrustes methods in the statistical analysis of shape. J. R. Stat. Soc. B **53**(2), 285–339 (1991)
28. Gu, L., Kanade, T.: A generative shape regularization model for robust face alignment. In: Proc. European Conf. on Computer Vision (2008)
29. Gu, L., Xing, E.P., Kanade, T.: Learning GMRF structures for spatial priors. In: Proc. IEEE Conf. on Comp. Vis. and Patt. Recog. (2007)
30. Hill, A., Cootes, T.F., Taylor, C.J.: Active shape models and the shape approximation problem. Image Vis. Comput. **14**, 601–607 (1996)
31. Hou, X., Li, S., Zhang, H., Cheng, Q.: Direct appearance models. In: Computer Vision and Pattern Recognition Conf. 2001, vol. 1, pp. 828–833 (2001)
32. Huang, Y., Liu, Q., Metaxas, D.N.: A component based deformable model for generalized face alignment. In: Proc. IEEE Int'l Conf. on Comp. Vis., pp. 1–8 (2007)
33. Jones, M.J., Poggio, T.: Multidimensional morphable models: A framework for representing and matching object classes. Int. J. Comput. Vis. **2**(29), 107–131 (1998)
34. Kirby, M., Sirovich, L.: Application of the Karhumen–Loeve procedure for the characterization of human faces. IEEE Trans. Pattern Anal. Mach. Intell. **12**(1), 103–108 (1990)
35. la Torre, F.D., Collet, A., Quero, M., Cohn, J.F., Kanade, T.: Filtered component analysis to increase robustness to local minima in appearance models. In: Proc. IEEE Conf. on Comp. Vis. and Patt. Recog. (2007)
36. Lee, H.-S., Kim, D.: Tensor-based AAM with continuous variation estimation: Application to variation-robust face recognition. IEEE Trans. Pattern Anal. Mach. Intell. **31**(6), 1102–1116 (2009)
37. Liang, L., Wen, F., Xu, Y.-Q., Tang, X., Shum, H.-Y.: Accurate face alignment using shape constrained Markov network. In: Proc. IEEE Conf. on Comp. Vis. and Patt. Recog. (2006)
38. Liang, L., Xiao, R., Wen, F., Sun, J.: Face alignment via component-based discriminative search. In: Proc. European Conf. on Computer Vision (2008)
39. Liu, X.: Discriminative face alignment. IEEE Trans. Pattern Anal. Mach. Intell. **31**(11), 1941–1954 (2009)
40. Lu, H.-M., Fainman, Y., Hecht-Nelson, R.: Image manifolds. In: Proc. SPIE Symposium on Electronic Imaging: Science and Technology (1998)
41. Lucey, S., Wang, Y., Saragih, J., Cohn, J.F.: Non-rigid face tracking with enforced convexity and local appearance consistency constraint. Image Vis. Comput. **28**(5), 781–789 (2010)
42. Matthews, I., Baker, S.: Active appearance models revisited. Int. J. Comput. Vis. **26**(10), 135–164 (2004)
43. Matthews, I., Xiao, J., Baker, S.: 2D vs. 3D deformable face models: Representational power, construction, and real-time fitting. Int. J. Comput. Vis. **75**(1), 93–113 (2007)

44. Milborrow, S., Nicolls, F.: Locating facial features with an extended active shape model. In: Proc. European Conf. on Computer Vision (2008)
45. Paquet, U.: Convexity and Bayesian constrained local models. In: Proc. IEEE Conf. on Comp. Vis. and Patt. Recog. (2009)
46. Romdhani, S., Gong, S., Psarrou, A.: A multi-view non-linear active shape model using kernel PCA. In: 10th British Machine Vision Conf., vol. 2, pp. 483–492, September 1999
47. Roweis, S., Saul, L.: Nonlinear dimensionality reduction by locally linear embedding. Science (2000)
48. Saragih, J., Goecke, R.: A nonlinear discriminative approach to AAM fitting. In: Proc. IEEE Int'l Conf. on Comp. Vis. (2007)
49. Saragih, J., Goecke, R.: Learning AAM fitting through simulation. Pattern Recognit. **42**(11), 2628–2636 (2009)
50. Saragih, J.M., Lucey, S., Cohn, J.F.: Deformable model fitting with a mixture of local experts. In: Proc. IEEE Int'l Conf. on Comp. Vis. (2009)
51. Saragih, J.M., Lucey, S., Cohn, J.F.: Face alignment through subspace constrained mean-shifts. In: Proc. IEEE Int'l Conf. on Comp. Vis. (2009)
52. Sclaroff, S., Isidoro, J.: Active blobs. In: 6th Int'l Conf. on Comp. Vis., pp. 1146–1153 (1998)
53. Scott, I.M., Cootes, T.F., Taylor, C.J.: Improving appearance model matching using local image structure. In: Information Processing in Medical Imaging, pp. 258–269. Springer, Berlin (2003)
54. Sozou, P.D., Cootes, T.F., Taylor, C.J., Mauro, E.C.D.: Non-linear generalization of point distribution models using polynomial regression. Image Vis. Comput. **13**(5), 451–457 (1995)
55. Stegmann, M.B., Ersbøll, B.K., Larsen, R.: FAME—a flexible appearance modelling environment. IEEE Trans. Med. Imaging **22**(10), 1319–1331 (2003)
56. Stegmann, M.B., Larsen, R.: Multi-band modelling of appearance. Image Vis. Comput. **21**(1), 66–67 (2003)
57. Tenenbaum, J.B., Silva, V.D., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Science **290**, 2319–2323 (2000)
58. Turk, M., Pentland, A.: Eigenfaces for recognition. J. Cogn. Neurosci. **3**(1), 71–86 (1991)
59. van Ginneken, B., Frangi, A.F., Stall, J.J., ter Haar Romeny, B.M.: Active shape model segmentation with optimal features. IEEE Trans. Med. Imaging **21**, 924–933 (2002)
60. Vasilescu, M.A.O., Terzopoulos, D.: Multilinear analysis of image ensembles: TensorFaces. In: Proc. European Conf. on Computer Vision (2002)
61. Vetter, T.: Learning novel views to a single face image. In: 2nd Int'l Conf. on Automatic Face and Gesture Recognition 1996, pp. 22–27, October 1996
62. Wu, H., Liu, X., Doretto, G.: Face alignment via boosted ranking model. In: Proc. IEEE Conf. on Comp. Vis. and Patt. Recog. (2008)
63. Zhou, S.K., Comaniciu, D.: Shape regression machine. In: Proc. Int'l Conf. on Information Processing in Medical Imaging (2007)