Generalization to Novel Views from a Single Face Image

Thomas Vetter and Volker Blanz

Max-Planck-Institut für biologische Kybernetik
Spemannstr. 38
72076 Tübingen – Germany
thomas.vetter@tuebingen.mpg.de
volker.blanz@tuebingen.mpg.de

Abstract. When only a single image of a face is available, can we generate new images of the face across changes in viewpoint or illumination? The approach presented in this paper acquires its knowledge about possible image changes from other faces and transfers this prior knowledge to a novel face image. In previous work we introduced the concept of linear object classes (Vetter and Poggio, 1997; Vetter, 1997): In an image based approach, a flexible image model of faces was used to synthesize new images of a face when only a single 2D image of that face is available.

In this paper we describe a new general flexible face model which is now "learned" from examples of individual 3D-face data (Cyberware-scans). In an analysis-by-synthesis loop the flexible 3D model is matched to the novel face image. Variation of the model parameters, similar to multidimensional morphing, allows for generating new images of the face where viewpoint, illumination or even the expression is changed.

The key problem for generating a flexible face model is the computation of dense correspondence between all given example faces. A new correspondence algorithm is described, which is a generalization of existing algorithms for optic flow computation to 3D-face data.

1 Introduction

"Can you imagine?"
"Yes, I see ......"
In human language mental imagery seems to be a natural ability. Imagery is often discussed as one of the basic forms of human cognition for the analysis of situations or scenes (Kosslyn, 1994).

A similar concept is developed in machine intelligence for the problem of pattern analysis and image understanding. In order to separate or compensate for the many factors that can change the image of an object, image models are built that allow the influence of each of these imaging factors to be simulated. So in computer vision imagery is directly translated into image synthesis and an image analysis is performed by matching an image model to a novel image, thereby parameterizing the novel image in terms of a known model.

H. Wechsler et al. (eds.), *Face Recognition*
© Springer-Verlag Berlin Heidelberg 1998

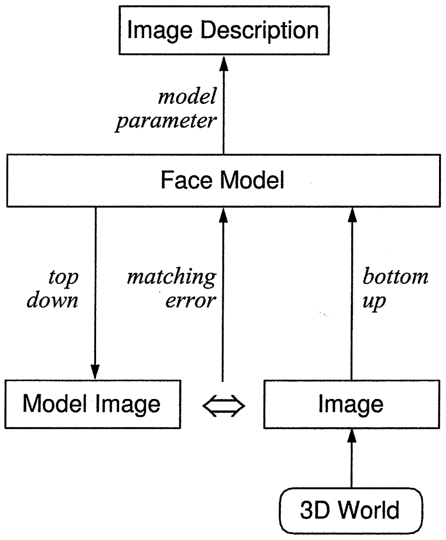

Figure 1: *A simple analysis by synthesis schema. To analyze an image if it represents a face, the face model generates a model image which is compared with the input image. After minimizing the matching error, the internal model parameters lead to the description of the input image.*

Analysis by synthesis

The requirement of pattern synthesis for pattern analysis has often been proposed within a Bayesian framework (Grenander, 1978; Mumford, 1996) or has been formulated as an alignment technique (Ullman, 1989). This is in contrast to pure of *bottom-up* techniques which have been advocated especially for the early stages of visual signal processing (Marr, 1982). Here a standard strategy is to reduce a signal to a feature vector and to compare this vector with those expected for signals in various categories. A crucial problem of these algorithms is that they cannot explicitly describe variations between or within the categories and therefore have difficulty separating unexpected noise from the variations within a particular category.

In contrast, the algorithm described in this paper works by actively reconstructing the signal to be analyzed. In an additional *top-down* path an estimated signal is generated and compared to the present input signal. Then by comparing the real signal with its reconstruction it is decided if the analysis is sufficient to explain the signal or not. Clearly, such an approach has the additional problem of defining a model function to reconstruct the input signal.

This paper focuses on the analysis and synthesis of images of a specific object class – that is on images of human faces. For object classes, such as faces or cars, where all objects share a common similarity, such a model function could be learned from examples. That is, the image model for the

whole object class is derived by exploiting some prototypical example images.
Models developed for the analysis and synthesis of images of a specific class of objects must solve two problems simultaneously:

- The model must be able to synthesize images that cover the whole range of possible images of the class.
- It must be possible to match the model to a novel image. Formally, this leads to an optimization problem with all of the associated requirements that a global minimum can be found.

In recent years, two-dimensional image-based face models have been constructed and applied for the synthesis of rigid and nonrigid face transformations (Choi et al., 1991; Beymer et al., 1993; Lanitis et al., 1995). These models exploit prior knowledge from example images of prototypical faces and work by building flexible image-based representations (*active shape models*) of known objects by a linear combination of labeled examples. These representations are applied for the task of image search and recognition or synthesis (Lanitis et al., 1995). The underlying coding of an image of a new object or face is based on linear combinations of the two-dimensional shape (warping fields) of examples of prototypical images as well as the linear combinations of their color values at corresponding locations (texture).

For the problem of synthesizing novel views to a single example image of a face we developed over the last years the concept of *linear object classes* (Vetter and Poggio, 1997; Vetter, 1997). This image-based method allows us to compute novel views of a face from a single image. On the one hand, the method draws on a general flexible image model which can be learned automatically from examples images, and on the other hand, on an algorithm that allows for matching this flexible model to a novel face image. The novel image now can be described or coded through the internal model parameters which are necessary to reconstruct the image. The design of the model allows also for synthesizing new views of the face.

In this paper we replace the two-dimensional image model by a three-dimensional flexible face model. A flexible three-dimensional face model will lead on the one hand to a more efficient data representation and on the other hand to a better generalization to new illumination conditions.

In all these techniques, it is crucial to establish the correspondence between each example face and single reference face, matching image points in the two-dimensional approach, and surface points in the three-dimensional case. Correspondence is a key step posing a difficult problem. However, for images of objects which share many common features, such as faces all seen from a single specific viewpoint, automated techniques seem feasible. The techniques applied in the past can be separated in two groups, one which establishes the correspondence for a small number of feature points only and into the techniques computing the correspondence for every pixel in an image. For the first approach usually models of particular features like the eye corners or the whole chin line are developed off line and then matched to a new image (Lanitis et al., 1995; Herpers et al., 1996). The second technique computes the correspondence for each pixel in an image by comparing this image to a reference image using methods derived from optical flow computation(Beymer et al., 1993; Beymer and Poggio, 1996). In this paper we will extend this method of dense correspondence which we already applied successfully to face images (Vetter and Poggio, 1997; Vetter et al., 1997), to the three-dimensional face data.

The paper is organized as follows. First, we describe the flexible three-dimensional face model and compare it to the two-dimensional image models we used earlier. Second we describe an algorithm to compute dense correspondence between individual 3D models of human faces. Third we describe an algorithm that allows to match the flexible face model to a novel image. Finally we show examples for synthesizing new images from a single image of a face and describe future improvements.

2 Flexible 3D face models

In this section we will give the formulation of a flexible three-dimensional face model which captures prior knowledge about faces exploiting the general similarity among faces. The model is a straight forward extension of the linear object class approach as described earlier(Vetter and Poggio, 1997; Vetter, 1997). Prototypical examples of an object class like faces are linked to a general class model that captures the similarity and regularities specific for this object class.

Three-dimensional models

In computer graphics, presently the most realistic three-dimensional face representations consist of a 3D mesh describing the geometry and a texture map capturing the color data of a face. These representations of individual faces are obtained either by three-dimensional scanning devices or through photogrammetric techniques from several two-dimensional images of a specific face (Parke, 1974; Akimoto et al., 1993). The synthesis of new faces by interpolation between such face representation was already demonstrated in the pioneering work of Parke (1974). Recently this idea of forming linear combinations of faces was used and extended to a general three-dimensional flexible face model for the analysis and synthesis of two-dimensional facial images (Choi et al., 1991; Poggio and Vetter, 1992; Vetter and Poggio, 1997).

Shape model: The three-dimensional geometry of a face is represented by a shape-vector $\mathbf{S} = (X_1, Y_1, Z_1, X_2, ..., Y_n, Z_n)^T \in \Re^{3n}$, that contains the X, Y, Z-coordinates of its n vertices. The central assumption for the formation of a flexible face model is that a set of M example faces $\mathbf{S}_i$ is available. Additionally, it is assumed that all these example faces $\mathbf{S}_i$ consist of the same number of n consistently labeled vertices, in other words all example faces are in full correspondence (see next section on correspondence). Usually this labeling is defined on an average face shape, which is obtained iteratively and which is often denoted as reference face $\mathbf{S}_{ref}$. Additionally, all faces are assumed to be aligned in an optimal way by rotating and translating them in three-dimensional space. Under this assumptions a new face geometry $\mathbf{S}_{model}$ can be generated as a linear combination of M example shape-vectors $\mathbf{S}_i$ each weighted by c_i

$$\mathbf{S}_{model} = \sum_{i=1}^{M} c_i \; \mathbf{S}_i \; . \tag{1}$$

The linear shape models derived in equation (1) allows for representing a new shape $\mathbf{S}$ as its approximation through the M example shapes $\mathbf{S}_i$: $\mathbf{S} \approx \sum_{i=1}^{M} c_i\ \mathbf{S}_i$. In other words, the example shapes represent a shape basis onto which a new shape $\mathbf{S}$ is projected. The coefficients c_i of the projection then define a coding of the original shape vector in this vector space which is spanned by all examples. A common strategy for increasing the efficiency of such a coding schema is to reduce the statistical redundancy within the example space. Principal component analysis (PCA) also known as Karhunen-Loeve expansion, is the standard procedure to obtain an optimal reduction of the dimensionality in data (for more details see (Duda and Hart, 1973)). While an optimization of the data representation is essential for any application, here, however, it will not be further discussed since it does not affect the basic idea of this paragraph which is the modeling of faces in a linear vector space.
Texture model: The second component of a flexible three-dimensional face or head model is texture information, which is usually stored in a *texture map.* A texture map is simply a two-dimensional color pattern storing the brightness or color values, ideally only the *albedo* of the surface is stored. This pattern can be synthetically generated or can be a scanned image.

A u, v coordinate system is introduced to associate the texture map with the modeled surface. The texture map is defined in the two-dimensional u, v coordinate system. For polygonal surfaces as defined through a shape vector $\mathbf{S}$, each vertex has an assigned u, v texture coordinate. For points on the surface between vertices, the u, v coordinates are interpolated. For convenience we assume that the total number n of stored values in such a texture map is equal to the total number of vertices in a shape vector $\mathbf{S}$.

The linear texture model, described in (Choi et al., 1991), starts from a set of M example face textures $\mathbf{T}_i$. Equivalent to the shape model described earlier it is assumed that all M textures $\mathbf{T}_i$ consist of the same number of n consistently labeled texture values that is all textures are in full correspondence. For texture synthesis linear models are used again. A new texture $\mathbf{T}_{model}$ is generated as the weighted sum of M given example textures $\mathbf{T}_i$ as follows

$$\mathbf{T}_{model} = \sum_{i=1}^{M} b_i \mathbf{T}_i, \tag{2}$$

Equivalent to the linear expansion of shape vectors (equation 1) the linear expansion of textures can be understood and used as an efficient coding schema for textures. A new texture can be coded by its M projection coefficients b_i in the 'texture vector space" spanned by M basis textures. Again, these basis textures can be just some prototypical example textures or can be derived from a large example set through an data optimization technique like PCA.

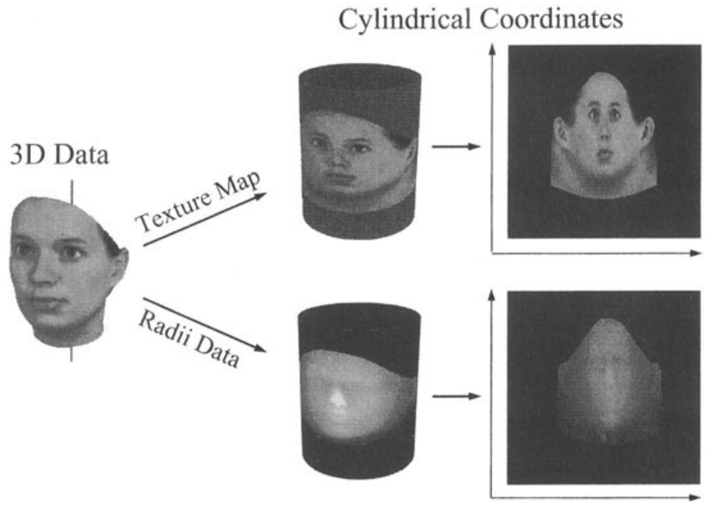

Figure 2: *Three-dimensional head data represented in cylindrical coordinates result in a data format which consists of two 2D-images. One image is the texture map (top right), and in the other image the geometry is coded (bottom right).*

3 3D Correspondence with Optical Flow

The key assumption of the flexible face model is the correspondence between all three-dimensional example data sets. That is, we have to find for every vertex location in one face data set, e.g. a vertex located on the nose, the corresponding vertex location on the nose in a reference face. This is in general a hard problem. However, assuming that all face data sets are roughly aligned and they are not categorical different like having a beard or not, an automatic technique is feasible for computing the correspondence. The key idea of the work described in this paragraph is to modify an existing optical flow algorithm to match points on the surfaces of three-dimensional objects instead of points in 2D-images. While establishing correspondence between three-dimensional objects has only recently become an issue, matching correspondent structures in two-dimensional images has been studied for many years.

Optical Flow Algorithm

In video sequences, in order to estimate the velocities of scene elements with respect to the camera it is necessary to compute the vector-field of optical flow, which expresses at each point $p_1 = (x_1, y_1)$ in the first image the displacement $(\delta x, \delta y) = (x_2 - x_1, y_2 - y_1)$ to the corresponding point $p_2 = (x_2, y_2)$ in the following image. A variety of different optical flow algorithms have been designed to solve this problem (for a review see (Barron et al., 1994)). Unlike temporal sequences taken from one scene, a comparison of images of completely different scenes or faces may violate a number of important assumptions made in optical flow estimation. However, it was shown that some optical flow algorithms can still cope with this more difficult matching problem, opening up a wide range of applications in image analysis and synthesis (Beymer et al., 1993).

For building flexible image models of faces (Vetter and Poggio, 1997) we used a coarse-to-fine gradient-based method (Bergen et al., 1992) applied to the Laplacians of the images and followed an implementation described in (Bergen and Hingorani, 1990). The Laplacian of the images were computed from the Gaussian pyramid adopting the algorithm proposed by (Burt and Adelson, 1983). For every point x, y in an image $I(x, y)$, the algorithm attempts to minimize the error term $E = \sum (I_x \delta x + I_y \delta y - \delta I)^2$ for $\delta x, \delta y$, with I_x, I_y being the spatial image derivatives of the Laplacians and δI the difference of the Laplacians of the two compared images. The coarse-to-fine strategy starts with low resolution images and refines the computed displacements when finer levels are processed. The final result of this computation $(\delta x, \delta y)$ is used as an approximation of the spatial displacement of each pixel between two images.

Three-dimensional face representations.
The adaptation and extension of this optical flow algorithm to the three-dimensional head data is straight forward due to the fact that the cylindrical representation of a head model is analogous to images (see figure 2). Instead of grey-level values in image coordinates x,y here we store the radius values and the color values for each angle ϕ and height h. A parameterization of a three-dimensional head in cylindrical coordinates results therefore in two 'images', one representing the geometry of the head and the other containing the texture information. In order to compute the correspondence between different heads, both texture and geometry were considered simultaneously. The optical flow algorithm as described earlier had to be modified in the following way. Instead of comparing a scalar grey-level function $I(x, y)$, our modification of the algorithm attempts to find the best fit for the vector function

$$\vec{F}(h, \phi) = \begin{pmatrix} radius(h, \phi) \\ red(h, \phi) \\ green(h, \phi) \\ blue(h, \phi) \end{pmatrix}$$

$$\text{in a norm } \left\| \begin{pmatrix} radius \\ red \\ green \\ blue \end{pmatrix} \right\|^2$$

$$= w_1 \cdot radius^2 + w_2 \cdot red^2 + w_3 \cdot green^2 + w_4 \cdot blue^2.$$

The coefficients $w_1 ... w_4$ correct for the different energies [contrasts] in range and color values, and they assign roughly the same weight to shape as to all color channels taken together.

For representing the geometry, radius values can be replaced by other surface properties such as Gaussian curvature or surface normals.

The system's output, at this stage, is a surface flow-field or correspondence function

$$C(h,\phi) = \begin{pmatrix} dh(h,\phi) \\ d\phi(h,\phi) \end{pmatrix}. \tag{3}$$

After computing the correspondence between all individual faces of the training set to the reference face, the T_i and S_i of the flexible face model can be computed.

4 Matching the flexible 3D model to a 2D image

Based on an example set of faces which are already in correspondence, new 3D shape vectors S^{model} and texture maps T^{model} can be generated by varying the coefficients c_i and b_i in equations (1) and (2). Combining model shape and model texture results in a complete 3D face representation which now can be rendered to a new model image I^{model}. This model image is not only a function of the model parameters c_i and b_i, it also depends on some projection parameters p_j and on the surface reflectance properties and illumination parameters r_j used for rendering. For the general matching problem of the model to a novel image I^{novel} we define the following error function

$$E(\vec{c},\vec{b},\vec{p},\vec{r}) = \frac{1}{2}\sum_{x,y}\left[I^{novel}(x,y) - I^{model}(x,y)\right]^2 \tag{4}$$

where the sum is over all pixels (x,y) in the images, I^{novel} is the novel image being matched and I^{model} is the current guess for the model image for a specific parameter setting $(\vec{c},\vec{b},\vec{p},\vec{r})$. Minimizing the error yields the model image which best fits the novel image with respect to the L_2 norm.

However, the optimization of the error function in equation (4) is extremely difficult for several reasons. First, the function is not linear in most of the parameters, second, the number of parameters is large (> 100) and additionally, the whole computation is extremely expensive since it requires the rendering of the three-dimensional face model to an image for each evaluation of the error function.

In this paper we will simplify the problem by assuming the illumination parameters $\vec{r}$ and also the projection parameters $\vec{p}$, such as view point, are known. This assumptions allows us to reduce the amount of rendering and also to use image modeling techniques developed earlier (Jones and Poggio, 1996; Vetter et al., 1997). By rendering images from all example faces under fixed illumination and projection parameters, the flexible 3D model is transformed into a flexible 2D face model. This allows us to generate new model images all depicting faces in the requested spatial orientation and under the known illumination. After matching this flexible 2D model (see below) to the novel image the optimal model parameters are used within the flexible 3D model to generate a three-dimensional face representation which best matches the novel target image.

4.1 Linear image model

To built the flexible 2D model we first render all 3D example faces under the given projection and illumination parameters to images $I_0, I_1, \ldots, I_M$. Let I_0 be the reference image which defines the topology for the whole model, and positions within I_0 are parameterized in (u, v). The 2D correspondence s_j between each pixel in the rendered reference image I_0 and its corresponding location in each rendered example image I_i, can be directly computed as the projection P of the 3D shapes between all 3D faces and the reference face, as $s_j = PS_j - PS_0$ with $s_o \equiv 0$.

These pixelwise correspondences between I_0 and each example image are mappings $\mathbf{s}_j : \mathcal{R}^2 \to \mathcal{R}^2$ which map the points of I_0 onto I_j, i.e. $\mathbf{s}_j(u,v) = (x,y)$ where (x,y) is the point in I_j which corresponds to (u,v) in I_0. We refer to $\mathbf{s}_j$ as a *correspondence field* and interchangeably as the *2D shape vector* for the vectorized I_j. Warping image I_j onto the reference image I_0 we obtain $\mathbf{t}_j$ as:

$$\mathbf{t}_j(u,v) = I_j \circ \mathbf{s}_j(u,v) \Leftrightarrow I_j(x,y) = \mathbf{t}_j \circ \mathbf{s}_j^{-1}(x,y).$$

So, $\{\mathbf{t}_j\}$ is the set of shape normalized prototype images, referred to as *texture vectors*. They are normalized in the sense that their shape is the same as the shape of the chosen reference image.
The flexible image model is the set of images I^{model}, parameterized by $\vec{c} = [c_0, c_1, \ldots, c_M], \vec{b} = [b_0, b_1, \ldots, b_M]$ such that

$$I^{model} \circ (\sum_{i=0}^{M} c_i s_i) = \sum_{j=0}^{M} b_j t_j. \tag{5}$$

The summation $\sum_{i=0}^{M} c_i s_i$ constrains the 2D shape of every model image to be a linear combination of the example 2D shapes. Similarly, the summation $\sum_{j=0}^{M} b_j t_j$ constrains the texture of every model image to be a linear combination of the example textures.

For any values for c_i and b_i, a model image can be rendered by computing $(x,y) = \sum_{i=0}^{M} c_i s_i(u,v)$ and $g = \sum_{j=0}^{M} b_j t_j(u,v)$ for each (u,v) in the reference image. Then the (x,y) pixel is rendered by assigning $I^{model}(x,y) = g$, that is by warping the texture into the model shape.

4.2 Matching a 2D face model to an image

For matching the flexible image model to a novel image we used the method described in (Jones and Poggio, 1996; Vetter et al., 1997). In 2D the error function as defined earlier in equation (4) is reduced to a function of the model parameters $\vec{c}$ and $\vec{b}$.

$$E(\vec{c}, \vec{b}) = \frac{1}{2} \sum_{x,y} \left[I^{novel}(x,y) - I^{model}(x,y)\right]^2$$

In order to compute I^{model} (see equation (5)) the shape transformation

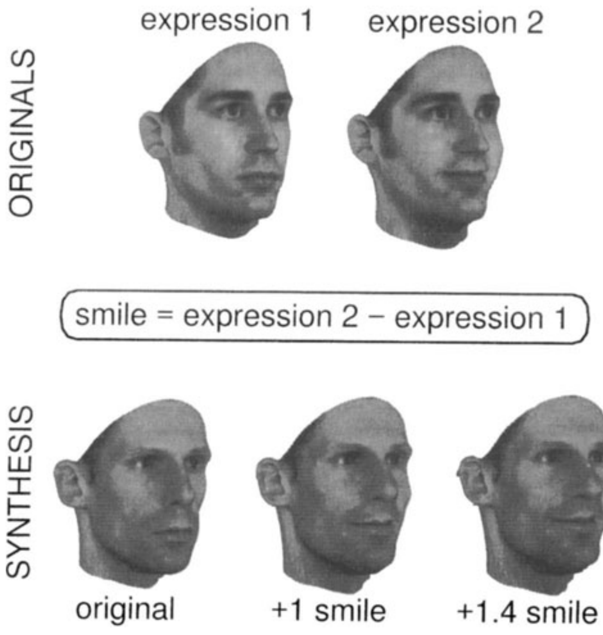

Figure 3: *Correspondence between faces allows to map expression changes from one face to the other. The difference between the two expressions in the top row is mapped on the left face in the lower row multiplied by a factor of 1 (center) or by 1.4 (lower right)*

$(\sum c_i s_i)$ has to be inverted or one has to work in the coordinate system (u, v) of the reference image, which is computationally more efficient. Therefore, the shape transformation (given some estimated values for $\vec{c}$ and $\vec{b}$) is applied to both I^{novel} and I^{model}. From equation (5) we obtain

$$E = \frac{1}{2} \sum_{u,v} [I^{novel} \circ (\sum_{i=0}^{M} c_i s_i(u, v)) - \sum_{j=0}^{M} b_j \mathbf{t}_j(u, v)]^2.$$

Minimizing the error yields the model image which best fits the novel image with respect to the L_2 norm. The optimal model parameters $\vec{c}$ and $\vec{b}$ are found by a stochastic gradient descent algorithm (Viola, 1995), a method that is fast and has a low tendency to be caught in local minima.

The robustness of the algorithm is further improved using a coarse-to-fine approach (Burt and Adelson, 1983). In addition to the textural pyramids, separate resolution pyramids are computed for displacement fields s in x and y.

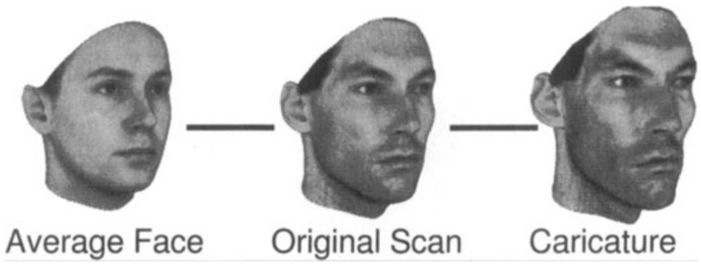

Figure 4: *The comparison the 3D model of an individual face to the average face allows for exaggerating the characteristics of that face. After establishing correspondence between all example faces* (200)*, the average face can be computed. Here a caricature of the face was obtained by increasing the difference of its shape and of texture vectors to the average face by a factor of two.*

5 Novel view synthesis

After matching the 2D image model to the novel image, the 2D model parameters $\vec{c}$ and $\vec{b}$ can be used in the three-dimensional flexible face model as defined in equations (1) and (2). The justification of this parameter transfer is discussed in detail under the aspect of *linear object classes* in (Vetter and Poggio, 1997). The output of the 3D flexible face model should be seen as an estimate of the three-dimensional shape from the two-dimensional image. Since this result is a complete 3D face model new images can be rendered from any viewpoint or under any illumination condition.

5.1 Non rigid face transformations

The correspondence between all faces within this flexible model allows for mapping non rigid face transitions 'learned' from one face onto all other faces in the model.
Facial expressions: In figure 3 the transformation for a smile is extracted from one person and then mapped onto the face of a different person. By computing the correspondence between two examples of one persons face, one example showing the face smiling and the other showing the face in a neutral expression, this results in a correspondence field or deformation field which captures the spatial displacement for each vertex in the model according to the smile. This expression specific correspondence field is formal identical to the correspondence fields between different persons described earlier. Such a 'smile-vector' now can be added or subtracted from each face which is in correspondence to one of the originals, making a neutral looking face more smily or giving a smiling face a more emotionless expression.

In the following, we will describe two procedures that evaluate the characteristics of an individual face in respect to the other faces in the data base.

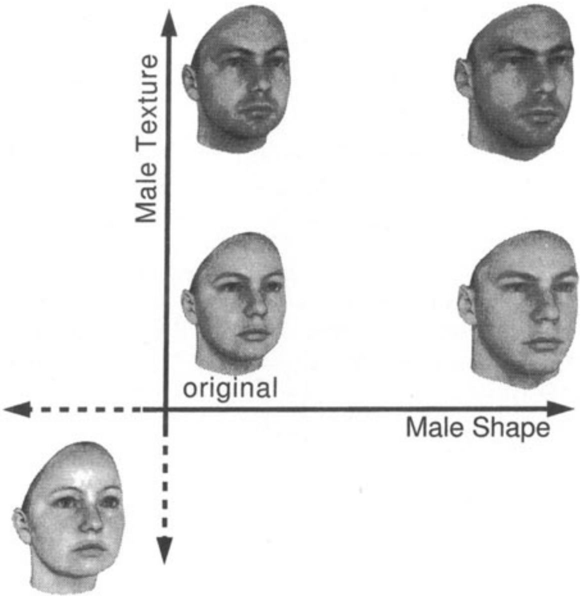

Figure 5: *Varying the perceived sex of a face. An original face is modified with respect to its shape (horizontal) and its texture (vertical). A sex specific vector is added or subtracted to the 3D-model of that face. The vector is defined by the difference between the male and the female average face where each average is computed over 100 faces.*

Caricaturing: Automated algorithms to enhance the characteristics of an individual face in images are well known (Brennan, 1985; Benson and Perrett, 1993). Typically, such algorithms operate as follows: First, a measure of the average value of a set of "features" across a large number of faces is computed. These features are defined, usually, as a set of facial landmark locations (e.g., corners of the eye and other points that are reasonably easy to localize/match on all faces) in the two-dimensional image. Next, to create a caricature of an individual face, a measure of the deviation of the face from the average two-dimensional configuration is computed. Finally, "distinctive" or unusual features of the face are exaggerated to produce the caricature. This generic algorithm applied to images, easily transfers to three-dimensional head representations (see also O'Toole et al., 1997) . Instead of using localized features we computed the average head shape by averaging over all shape vectors $\mathbf{S}_i$ and computed the average texture over all texture vectors $\mathbf{T}_i$. Now individual 3D-models of faces can be directly compared with this average at each vertex position. By increasing the difference (distance) to the average the characteristics of a face can be exaggerated, for example a relatively big nose, compared to the average, becomes even bigger and a small nose will shrink even more. In figure 4 a caricature of a face is shown, the caricature is generated by doubling the distance of the original face to the average.

A different question to ask is, how are male and female faces distributed in our flexible face model, are there simple parameters in the model that correlate with the sex of a face. Such a question can not be answered in advance, since the sex of faces did not influence our design of the flexible model. To answer this question we performed a simple classification experiment on our faces, which were represented through their shape and texture vectors. Surprisingly, already a very simple linear classifier separated female and male faces to more than 90% correct. The classifier we used was a hyperplane in our model space, which was between the female and the male average and additionally perpendicular to the vector that is defined as the direction between the female and the male average. In other words, the projection of an individual face onto the vector, defined through the direction between the female and the male average, is already a good indicator for the sex of a face. *Manipulating the sex of a face:* The capability of our face model to synthesize new faces, can be used to modify the sex specific appearance of a face. The vector, defined by the direction between the male and female average, added or subtracted to an individual face, generates a new face which is more female or male. In figure 5 such a manipulation is shown. The original face of a female person is modified separately in its texture and shape components by adding or subtracting this sex specific vector.

6 Data set

We used a 3D data set obtained from 200 laser scanned (*Cyberware*^{TM}) heads of young adults (100 male and 100 female).

The laser scans provide head structure data in a cylindrical representation, with radii of surface points sampled at 512 equally-spaced angles, and at 512 equally spaced vertical distances. Additionally, the RGB-color values were recorded in the same spatial resolution and were stored in a texture map with 8 bit per channel. All faces were without makeup, accessories, and facial hair. After the head hair was removed digitally (but with manual editing), individual heads were represented by approximately 70000 vertices and the same number of color values.
The data set of 200 human faces was split randomly in a test and in a training set (100 faces each). The training set was used to built the flexible face model while the images to reconstruct were rendered from the test set .
Images were rendered showing the faces 30° from frontal using mainly ambient light. The image size used in the experiments was 256-by-256 pixel and 8 bit per color channel.

7 Results

The correspondence between all 3D example face models and a reference face modell was computed automatically. The results were correct (visual inspection) for almost all 200 faces, only in 7 cases obvious correspondence errors occurred.

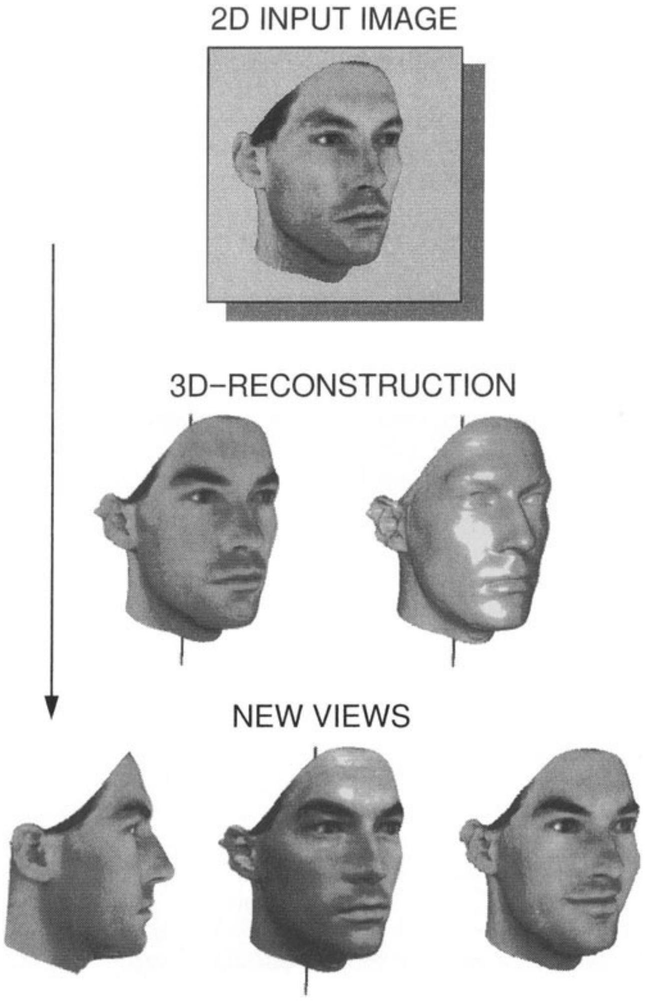

Figure 6: *Three-dimensional reconstruction of a face (center row) from a single two-dimensional image of known orientation and illumination (top row). The prior knowledge about faces was given through a training set of three-dimensional data of 100 faces different from the input face. The lower row shows new views generated by the flexible face model. From the left to the right, the face is shown from a new view point, under a changed illumination and smiling.*

Figure 6 shows an example of a three-dimensional face reconstruction from a single image. For the view synthesis we split our data set of 200 faces randomly into a training and a test set, each consisting of 100 faces. The training set was used to 'learn' a flexible model, where as test images of known orientation and illumination were rendered from the test set. After matching the model to the test image the model parameters were used to generate a complete three-dimensional face reconstuction. Presently, evaluation of the three-dimensional face reconstructions from single images is only based on visual inspection. Out of 100 reconstructions, 43 faces were highly similar and often hard to distinguish from the original. In 29 cases the shape reconstructions were good but texture showed obvious deficiencies in color. For the remaining 28 faces the reconstructions showed no similarity to the original. Still, face shape was natural and in only one case not human like, while texture problems were more severe. Figure 6 shows an example of a three-dimensional face reconstruction from a single image. We rated this example as highly similar but within this category it is lower average. The figure shows, beside the reconstruction in the center row, images where the view point, the illumination or the facial expression is changed.

8 Conclusions

We presented a method for approximating the three-dimensional shape of a face from just a single image. In an analysis-by-synthesis loop a flexible 3D-face model is matched to a novel image. The novel image now can be described or coded through the model parameters reconstructing the image. Prior knowledge on the three-dimensional appearance of faces derived from an example set of faces allows for predicting new images of a face. The results presented in this paper are preliminary. We hope the color problem can be resolved by appropriate constraints in color space. We also plan to apply a more sophisticated evaluation of reconstruction quality based on ratings by naive human subjects and automated similarity measures.

Clearly, the present implementation with its intermediate step of generating a complete 2D face model can not be the final solution. Next, we plan for each iteration step to form linear combinations in our 3D-representation first, render an image from this model and then perform the comparison to the target image. This requires several changes in our matching procedure to keep the computational costs tolerable.

Also, the present approach is restricted to images of faces where projection parameters and illumination conditions are known. The extension of the method to face images taken under arbitrary conditions will need several improvements. One the one hand, adding more free parameters into the matching procedure will require more sophisticated model representations especially in terms of the statistical dependence of the parameters. On the other hand, the linear model depends on the given example set. In order to represent faces from a different race or a different age group, the model will need examples of these, an effect also well known in human perception (cf. e.g. (O'Toole et al., 1994)).

A final judgment and comparison of the presented 3D-model approach with our 2D-image models presented earlier (Vetter and Poggio, 1997; Vetter,

1997) is currently difficult. At present, the two-dimensional linear object class approach seems to be more reliable in terms of its ability of match novel images. The number of wrong reconstuctions is less, however, for faces where the input image could be reconstructed, the 3D model showed a much better ability to generalize to new images. The implicit estimate of the surface normals in the image allowed for modifying the illumination conditions in the same image.

References

Akimoto, T., Suenaga, Y., and Wallace, R. (1993). Automatic creation of 3D facial models. *IEEE Computer Graphics and Applications*, 13(3):16–22.

Barron, J., Fleet, D., and Beauchemin, S. (1994). Performance of optical flow techniques. *Int. Journal of Computer Vision*, pages 43–77.

Benson, P. J. and Perrett, D. (1993). Perception and recognition of photographic quality caricatures : implications for the recognition of natural images. *European Journal of Cognitive Psychology*, 3:105–135.

Bergen, J., Anandan, P., Hanna, K., and Hingorani, R. (1992). Hierarchical model-based motion estimation. In *Proceedings of the European Conference on Computer Vision*, pages 237–252, Santa Margherita Ligure, Italy.

Bergen, J. and Hingorani, R. (1990). Hierarchical motion-based frame rate conversion. Technical report, David Sarnoff Research Center Princeton NJ 08540.

Beymer, D. and Poggio, T. (1996). Image representation for visual learning. *Science*, 272:1905–1909.

Beymer, D., Shashua, A., and Poggio, T. (1993). Example-based image analysis and synthesis. A.I. Memo No. 1431, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Brennan, S. E. (1985). The caricature generator. *Leonardo*, 18:170–178.

Burt, P. and Adelson, E. (1983). The Laplacian pyramide as a compact image code. *IEEE Transactions on Communications*, (31):532–540.

Choi, C., Okazaki, T., Harashima, H., and Takebe, T. (1991). A system of analyzing and synthesizing facial images. In *Proc. IEEE Int. Symposium of Circuit and Syatems (ISCAS91)*, pages 2665–2668.

Duda, R. and Hart, P. (1973). *Pattern classification and scene analysis.* John Wiley & Sons, New York.

Grenander, U. (1978). *Pattern Analysis, Lectures in Pattern Theory.* Springer, New York, 1 edition.

Herpers, R., Michaelis, M., Lichtenauer, K. H., and Sommer, G. (1996). Edge and keypoint detection in facial regions. In *Proc. International Conference on Automatic Face and Gesture Recognition*, pages 22–27, Killington, VT.

Jones, M. and Poggio, T. (1996). Model-based matching by linear combination of prototypes. A.i. memo no., Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Kosslyn, S. M. (1994). *Image and Brain.* MIT Press, Cambridge, MA.

Lanitis, A., Taylor, C., Cootes, T., and Ahmad, T. (1995). Automatic interpretation of human faces and hand gestures using flexible models. In M.Bichsel, editor, *Proc. International Workshop on Face and Gesture Recognition*, pages 98–103, Zurich, Switzerland.

Marr, D. (1982). *Vision,*. W. H. Freeman, San Fancisco.

Mumford, D. (1996). Pattern theory: A unifying perspective. In Knill, D. and Richards, W., editors, *Perception as Bayesian Inference.* Cambridge University Press.

O'Toole, A., Deffenbacher, K., Valentin, D., and Abdi, H. (1994). Structural aspects of face recognition and the other-race effect. *Memory and Cognition*, 22:208–224.

O'Toole, A., Vetter, T., Volz, H., and Salter, E. (1997). Three-dimensional caricatures of human heads: distinctiveness and the perception of facial age. *Perception*, in press.

Parke, F. (1974). A parametric model of human faces. Doctoral thesis, University of Utah, Salt Lake City.

Poggio, T. and Vetter, T. (1992). Recognition and structure from one 2D model view: observations on prototypes, object classes, and symmetries. A.I. Memo No. 1347, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.

Ullman, S. (1989). Aligning pictorial descriptions: An approach for object recognition. *Cognition*, 32:193–254.

Vetter, T. (1997). Synthestis of novel views from a single face image. *International Journal of Computer Vision*, (in press).

Vetter, T., Jones, M. J., and Poggio, T. (1997). A bootstrapping algorithm for learning linear models of object classes. In *IEEE Conference on Computer Vision and Pattern Recognition – CVPR'97*, Puerto Rico, USA. IEEE Computer Society Press.

Vetter, T. and Poggio, T. (1997). Linear objectclasses and image synthesis from a single example image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):733–742.

Viola, P. (1995). Alignment by maximization of mutual information. A.I. Memo No. 1548, Artificial Intelligence Laboratory, Massachusetts Institute of Technology.