Estimating Coloured 3D Face Models from Single Images: An Example Based Approach

Thomas Vetter and Volker Blanz

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

Abstract. In this paper we present a method to derive 3D shape and surface texture of a human face from a single image. The method draws on a general flexible 3D face model which is "learned" from examples of individual 3D-face data (Cyberware-scans). In an analysis-by-synthesis loop, the flexible model is matched to the novel face image.
From the coloured 3D model obtained by this procedure, we can generate new images of the face across changes in viewpoint and illumination. Moreover, nonrigid transformations which are represented within the flexible model can be applied, for example changes in facial expression.
The key problem for generating a flexible face model is the computation of dense correspondence between all given 3D example faces. A new correspondence algorithm is described which is a generalization of common algorithms for optic flow computation to 3D-face data.

1 Introduction

Almost an infinite number of different images can be generated from the human face. Any system that tries to analyse images of faces is confronted with the problem of separating different sources of image variation. For example, in order to identify a face of a person, the system must be able to ignore changes in illumination, orientation and facial expression, but it must be highly sensitive to attributes that are characteristic of identity. One approach to solving this problem is to transform the input image to a feature vector which is then compared to stored representations. In these 'bottom-up' strategies, the crucial problem is to choose appropriate features and criteria for comparison. An alternative approach is to build an explicit model of the variations of images. An image analysis is performed by matching an image model to a novel image, thereby coding the novel image in terms of a known model.

This paper focuses on the analysis and synthesis of images of a specific object class – that is on images of human faces. A model of an entire object class, such as faces or cars, with all objects sharing a common similarity, can be learned from a set of prototypical examples. Developed for the analysis and synthesis of images of a specific class of objects, it must solve two problems simultaneously:

- The model must be able to synthesize images that cover the whole range of possible images of the class.

– Matching the model to a novel image must be possible, avoiding local minima.

Recently, two-dimensional image-based face models have been constructed and applied for the synthesis of rigid and nonrigid face transformations [1, 2, 3]. 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 used for image search and recognition or synthesis [3]. The underlying coding of an image of a new object or face is based on linear combinations of prototypical images in terms of both two-dimensional shape (warping fields), and color values at corresponding locations (texture).

For the problem of synthesizing novel views to a single example image of a face, we have developed over the last years the concept of *linear object classes* [4]. 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 this flexible model to be matched to a novel face image. The novel image can now be described or coded by means of the internal model parameters which are necessary to reconstruct the image. The design of the model also allows new views of the face to be synthesized.

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 a single reference face, either by matching image points in the two-dimensional approach, or 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. 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 techniques computing the correspondence for every pixel in an image. For the first approach models of particular features such as the eye corners or the whole chin line are developed off line and then matched to a new image [3]. 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[2, 5].

In this paper, the method of dense correspondence, which we have already applied successfully to face images [4], will be extended to the three-dimensional face data. Firstly, we describe the flexible three-dimensional face model and compare it to the two-dimensional image models we used earlier. Secondly we describe an algorithm to compute dense correspondence between individual 3D models of human faces. Thirdly we describe an algorithm that allows us 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 a formulation of a flexible three-dimensional face model which captures prior knowledge about faces exploiting the general similarity among faces. The model is a straightforward extension of the linear object class approach as described earlier[4]. Exploiting the general similarity among faces, prototypical examples are linked to a general class model that captures regularities specific for this object class.

Three-dimensional models

In computer graphics, at present 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 by means of photogrammetric techniques from several two-dimensional images of a specific face [6, 7]. Synthesis of new faces by interpolation between such face representation was already demonstrated in the pioneering work of Parke (1974). Recently the idea of forming linear combinations of faces has been used and extended to a general three-dimensional flexible face model for the analysis and synthesis of two-dimensional facial images [1, 4].

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 model allows for approximating any given shape $\mathbf{S}$ as a linear combination with coefficients that are computed by projecting $\mathbf{S}$ onto the example shapes $\mathbf{S}_i$. 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.

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). This pattern can be recorded in a scanning process or generated synthetically.

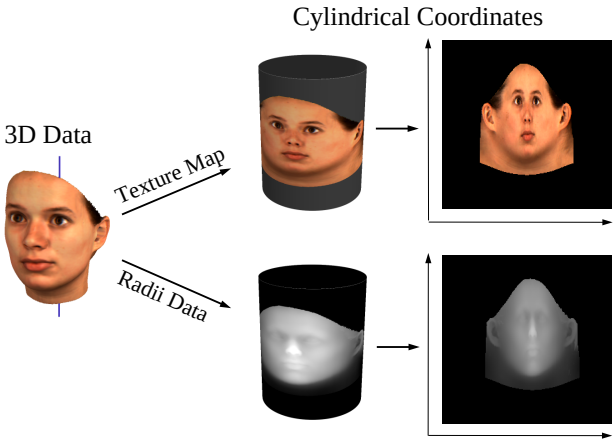

Fig. 1. *Three-dimensional head data represented in cylindrical coordinates result in a data format which consists of two 2D-images. One image is the texture map (top right), and in the other image the geometry is coded (bottom right).*

A u, v coordinate system associates 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 by a shape vector $\mathbf{S}$, each vertex has an assigned u, v texture coordinate. Between vertices, u, v coordinates are interpolated.

The linear texture model, described in [1] 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.

3 3D Correspondence with Optical Flow

In order to construct a flexible 3D face model, it is crucial to establish correspondence between a reference face and each individual face example. For all vertices of the reference face, we have to find the corresponding vertex location on each

face in the dataset. If, for example, vertex j in the reference face is located on the tip of the nose, with a 3D position described by the vector components X_j, Y_j, Z_j in $\mathbf{S}_{ref}$, then we have to store the position of the tip of the nose of face i in the vector components X_j, Y_j, Z_j of $\mathbf{S}_i$. In general, this is a difficult problem, and it is difficult to formally specify what correct correspondence is supposed to be. However, assuming that there are no categorical differences such as some having a beard and others not, an automatic method 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 on 2D images.

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 defines the displacements $(\delta x, \delta y) = (x_2 - x_1, y_2 - y_1)$ between points $p_1 = (x_1, y_1)$ in the first image and corresponding points $p_2 = (x_2, y_2)$ in the following image.

A variety of different optical flow algorithms have been designed to solve this problem (for a review see [8]). 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, some optical flow algorithms can still cope with this more difficult matching problem.

In previous studies [4], we built flexible image models of faces based on correspondence between images, using a coarse-to-fine gradient-based method [9] and following an implementation described in [10]:

For every point x, y in an image $I(x, y)$, the algorithm attempts to minimize the error term $E(x, y) = \sum_{R(x,y)}(I_x \delta x + I_y \delta y - \delta I)^2$ for $\delta x, \delta y$, with I_x, I_y being the spatial image derivatives and δI the difference of grey-levels of the two compared images. The region R is a 5x5 pixel neighbourhood of (x, y). Solving this optimization problem is achieved in a single iteration.

Since this is only a crude approximation to the overall matching problem, an iterative coarse-to-fine strategy is required. The algorithm starts with an estimation of correspondence on low-resolution versions of the two input images. The resulting flow field is used as an initial value to the computation on the next higher level of resolution. Iteratively, the algorithm proceeds to full resolution.

In our applications, results were dramatically improved if images on each level of resolution were computed not only by downsampling the original (Gaussian Pyramid), but also by band-pass filtering (Laplacian Pyramid). The Laplacian Pyramid was computed from the Gaussian pyramid adopting the algorithm proposed by [11].

Three-dimensional face representations.

The adaptation and extension of this optical flow algorithm to face surfaces in 3D is straightforward due to the fact that these two-dimensional manifolds can be parameterized in terms of two variables: In a cylindrical representation (see

figure 1), faces are described by radius and color values at each angle ϕ and height h. Images, on the other hand, consist of grey-level values in image coordinates x,y. Thus, in both cases correspondence can be expressed by a mapping $C : \Re^2 \to \Re^2$ in parameter space.

In order to compute correspondence between different heads, both texture and geometry were considered simultaneously. The optical flow algorithm described above had to be modified in the following way. Instead of comparing scalar grey-level functions $I(x, y)$, our modification of the algorithm attempts to find the best fit for the vector function

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

in a norm

$$\|(radius, red, green, blue)^T\|^2 = w_1 \cdot radius^2 + w_2 \cdot red^2 + w_3 \cdot green^2 + w_4 \cdot blue^2.$$

The coefficients $w_1 ... w_4$ correct for the different contrasts in range and color values, assigning approximately the same weights to variations in radius as to variations in all color channels taken together.

Radius values can be replaced by other surface properties such as Gaussian curvature or surface normals in order to represent the shapes of faces more appropriately.

The displacement between corresponding surface points is expressed by a correspondence function

$$C(h, \phi) = (\delta h(h, \phi), \delta \phi(h, \phi)). \tag{3}$$

Interpolation in low-contrast areas.
It is well known that in areas with no contrast or with strongly oriented intensity gradients, the problem of optical flow computation cannot be uniquely solved based on local image properties only (aperture problem). In our extension of the algorithm to surfaces of human faces, there is no structure to define correct correspondence on the cheeks, along the eyebrows and in many other areas, and indeed the method described so far yields spurious results here. While these problems might remain undetected in a simple morph between two faces, they still have significant impact on the quality of the flexible face model.

The ambiguities of correspondence caused by the aperture problem can be resolved if the flow field is required to be smooth. In a number of optical flow algorithms, smoothness constraints have been implemented as a part of an iterative optimization of correspondence [12, 13, 14].

In our algorithm, smoothing is performed as a separate process after the estimation of flow on each level of resolution. For the smoothed flow field $(\delta h'(h, \phi), \delta \phi'(h, \phi))$, an energy function is minimized using conjugate gradient descent such that on the one hand, flow vectors are kept as close to constant as possible over the whole domain, and on the other hand as close as possible to

the flow field $(\delta h(h,\phi), \delta\phi(h,\phi))$ obtained in the computation described above. The first condition is enforced by quadratic potentials that increase with the square distances between each individual flow vector and its four neighbours. These interconnections have equal strength over the whole domain. The second condition is enforced by quadratic potentials that depend on the square distance between $(\delta h'(h,\phi), \delta\phi'(h,\phi))$ and $(\delta h(h,\phi), \delta\phi(h,\phi))$ in every position (h,ϕ). These potentials vary over the (h,ϕ) domain: If the gradient of colour and radius values, weighted in the way described above, is above a given threshold, the coupling factor is set to a fixed, high value in the direction along the gradient, and zero in the orthogonal direction. This allows the flow vector to move along an edge during the relaxation process. In areas with gradients below threshold, the potential is vanishing, so the flow vector depends on its neighbours only. With our choice of the threshold, only 5% of all flow-vectors were set free in the low-resolution step, but 85% in the final full-resolution computation.

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 can now be rendered to a new model image I^{model}. This model image is not fully specified by the model parameters c_i and b_i, but it also depends on some projection parameters p_j and on the parameters r_j of surface reflectance properties and illumination used for rendering. For the general problem of matching the model to a novel image I^{novel} we define the following error function

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

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 $(\mathbf{c},\mathbf{b},\mathbf{p},\mathbf{r})$. 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 $\mathbf{r}$ and also the projection parameters $\mathbf{p}$, such as viewpoint, are known. This assumption allows us to reduce the amount of rendering and also to use image modeling techniques developed earlier [15, 16]. 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 depicting faces in the requested spatial orientation and under the known illumination. After matching this flexible 2D model to the novel image (see below), 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 build the flexible 2D model, first we 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, and let positions within I_0 be parameterized by (u, v). Pixelwise correspondences between I_0 and each example image are mappings $\mathbf{s}_j : \mathcal{R}^2 \rightarrow \mathcal{R}^2$ 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 *2D shape vector* for the vectorized I_j.

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 from the projection P of the differences of 3D shapes between all 3D faces and the reference face, $PS_j - PS_0$.

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 $\mathbf{c} = [c_0, c_1, \ldots, c_M], \mathbf{b} = [b_0, b_1, \ldots, b_M]$ 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 [15, 16]. In 2D, the error function as defined in equation (4) is reduced to a function of the model parameters **c** and **b**.

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

In order to compute I^{model} (see equation (5)) the shape transformation ($\sum c_i s_i$) 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 **c** and **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 **c** and **b** are found by a stochastic gradient descent algorithm [17], 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 [11]. In addition to the textural pyramids, separate resolution pyramids are computed for displacement fields s in x and y.

Separate matching of facial subregions.

In the framework described above, the flexible face model has M degrees of freedom for texture and M for shape, if M is the number of example faces. The number of degrees of freedom can be increased by dividing faces into independent subregions which are optimized independently [18], for example into eyes, nose, mouth and a surrounding region. Once correspondence is established, it is sufficient to define these regions on the reference face. In the linear object class, this segmentation is equivalent to splitting down the vector space of faces into independent subspaces. The process of fitting the flexible face model to given images is modified in the following ways: First, model parameters **c** and **b** are estimated as described above, based on the whole image. Starting from these initial values, each segment is then optimized independently, with its own parameters **c** and **b**. The final 3D model is generated by computing linear combinations for each segment separately and blending them at the borders according to an algorithm proposed for images by [11, 19] .

5 Novel view synthesis

After matching the 2D image model to the novel image, the 2D model parameters **c** and **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

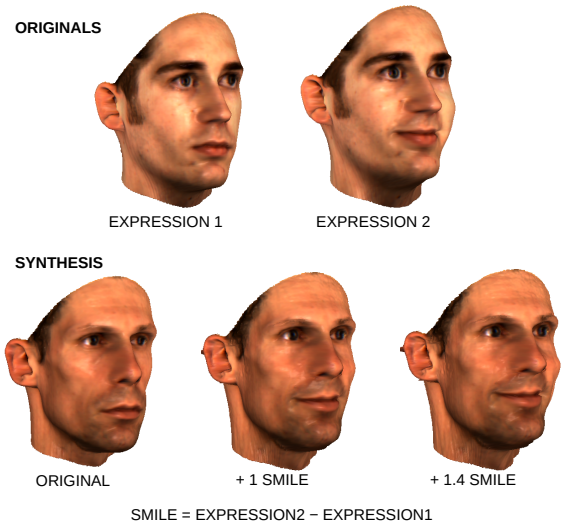

Fig. 2. *Correspondence between faces allows us to map expression changes from one face to the other. The difference between the two expressions in the top row is mapped on the left face in the lower row multiplied by a factor of 1 (center) or by 1.4 (lower right)*

in detail under the aspect of *linear object classes* in [4]. The output of the 3D flexible face model is 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.

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. In figure 2, the transformation for a smile is extracted from one person and then mapped onto the face of another person. Computing the correspondence between two examples of one person's face, one example showing the face smiling and the other showing the face in a neutral expression, 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 formally identical to the correspondence fields between different persons described earlier. Such a 'smile-vector' can now 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.

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).

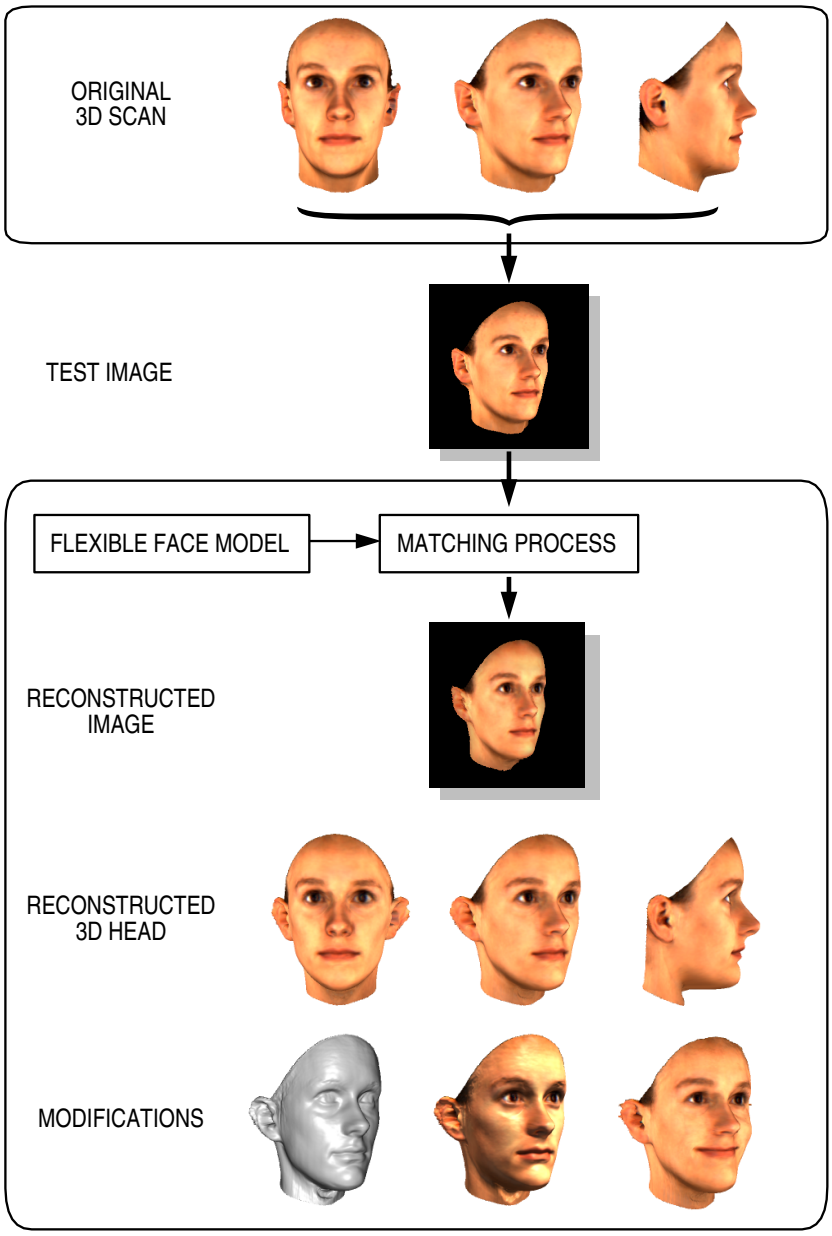

Fig. 3. *Three-dimensional reconstruction of a face from a single two-dimensional image of known orientation and illumination. The test image is generated from an original 3D scan (top) which is not part of the training set of 100 faces. Using the flexible face model derived from the training set, the test image can be reconstructed by optimizing parameters in a 2D matching process. These model parameters also describe the estimated 3D structure of the reconstructed head. The bottom row illustrates manipulations of surface properties, illumination and facial expression (left to right).*

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.
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. From the test set, images were rendered showing the faces 30° from frontal, and using mainly ambient light. The image size used in the experiments was 256-by-256 pixel and 8 bit per color channel.

Fig. 4. *In order to recover more of the characteristics of a test image (left), after optimizing model parameters for the whole face (center), separate sets of parameters are optimized for different facial subregions independently (right). As subregions, we used eyes, nose, mouth and the surrounding area. Improvements can be seen in the reconstructed shape of lips and eyes.*

7 Results

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

Figure 3 shows an example of a three-dimensional face reconstruction from a single image of known orientation and illumination. After matching the model to the test image, the model parameters were used to generate a complete three-dimensional face reconstruction. Figure 3 illustrates the range of images that can be obtained when a full 3D head is reconstructed, including both manipulations of viewpoint, surface material or illumination, and changes of internal properties such as facial expression.

At present, evaluation of the three-dimensional face reconstructions from single images is only based on visual inspection. Out of 100 reconstructions, 72

faces were highly similar and often hard to distinguish from the original in a 30° view. In 27 cases, persons were still easy to identify, but images displayed some differences, for example in the shape of cheeks or jaw. Only one reconstruction showed a face clearly unlike the original, yet a face that could very well exist.

We rated the example shown in figure 3 as highly similar, but within this category it is average. While the reconstructed head appears very similar to the original image in a 30° view, it is not surprising to notice that front and profile views reveal a number of differences.

Since texture vectors in the flexible 2D face model only code points that are part of the face, no particular background colour is specified. Performance should be roughly the same for any background that produces clear contours, and indeed results are almost identical for test images with black and light brown backgrounds.

The flexible model approach can also be applied to faces that are partially occluded, for example wearing sunglasses, or for persons with unusual features such as beards or moles. In each iteration of the matching process, contributions are ignored for those pixels which have the largest disagreement in color values with respect to the original image [15]. Along with a relatively stable reconstruction of all visible areas, the system yields an estimate of the appearance of occluded areas, based on the information available in the image and the internal structure of the face model (figure 5). Moreover, the approach allows detection of conspicuous image areas. Conspicuousness of an individual pixel can be measured in different ways, for example by plain disagreement of reconstructed colour values with the original image, by the required change in model parameters to fit the original (used in figure 5), or by the loss in overall matching quality for fitting this specific pixel within the flexible face model. The second and third measures take into account that in some regions small changes of the model parameters lead to considerable changes in color value, while other regions show only small variations.

Fig. 5. *Reconstruction of a partly occluded face. In the matching process, pixels with large disagreement in colour values were ignored. No segmentation was applied. The approach allows detection of conspicuous image areas (right, see text).*

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 can now be described or coded by means of the model parameters reconstructing the image. Prior knowledge of the three-dimensional appearance of faces, derived from an example set of faces, allows new images of a face to be predicted. The results presented in this paper are preliminary. We 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, render an image from this model and then perform a comparison with the target image. This requires several changes in our matching procedure to keep the computational costs tolerable.

Conditions in our current matching experiments were simplified in two ways. Firstly, all images were rendered from our test set of 3D-face scans. Secondly, projection parameters and illumination conditions were known. The extension of the method to face images taken under arbitrary conditions, in particular to any photopraph, will require several improvements. Along with a larger number of free parameters in the matching procedure, model representations need to be more sophisticated, especially in terms of the statistical dependence of the parameters. Reliable results on a wider variety of faces, such as different age groups or races, can only be obtained with an extended data set.

While the construction of 3D-models from a single image is very difficult and often an ill-posed problem in a bottom-up approach, our example-based technique allows us to obtain satisfying results by means of a maximum likelihood estimate. The ambiguity of the problem is reduced when several images of an object are available, a fact that is exploited in stereo or motion-based techniques. In our framework, we can make use of this additional information by simultaneously optimizing the model parameters $\mathbf{c}$ and $\mathbf{b}$ for all images, while the camera and lighting parameters $\mathbf{p}$ and $\mathbf{r}$ are adjusted for each image separately.

The method presented in this paper appears to be complementary to non model-based techniques such as stereo. While our approach is limited to results within a fixed model space, these techniques are often not reliable in areas with little structure. For the future, we plan to combine the benefits of both techniques.

References

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

2. D. Beymer, A. Shashua, and T. Poggio, "Example-based image analysis and synthesis," A.I. Memo No. 1431, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1993.
3. A. Lanitis, C. Taylor, T. Cootes, and T. Ahmad, "Automatic interpretation of human faces and hand gestures using flexible models," in *Proc. International Workshop on Face and Gesture Recognition* (M.Bichsel, ed.), (Zürich, Switzerland), pp. 98–103, 1995.
4. T. Vetter and T. Poggio, "Linear objectclasses and image synthesis from a single example image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 733–742, 1997.
5. D. Beymer and T. Poggio, "Image representation for visual learning," *Science*, vol. 272, pp. 1905–1909, 1996.
6. F. Parke, "A parametric model of human faces," doctoral thesis, University of Utah, Salt Lake City, 1974.
7. T. Akimoto, Y. Suenaga, and R. Wallace, "Automatic creation of 3D facial models," *IEEE Computer Graphics and Applications*, vol. 13, no. 3, pp. 16–22, 1993.
8. J. Barron, D. Fleet, and S. Beauchemin, "Performance of optical flow techniques," *Int. Journal of Computer Vision*, pp. 43–77, 1994.
9. J. Bergen, P. Anandan, K. Hanna, and R. Hingorani, "Hierarchical model-based motion estimation," in *Proceedings of the European Conference on Computer Vision*, (Santa Margherita Ligure, Italy), pp. 237–252, 1992.
10. J. Bergen and R. Hingorani, "Hierarchical motion-based frame rate conversion," technical report, David Sarnoff Research Center Princeton NJ 08540, 1990.
11. P. Burt and E. Adelson, "The Laplacian pyramide as a compact image code," *IEEE Transactions on Communications*, no. 31, pp. 532–540, 1983.
12. B. K. P. Horn and B. G. Schunck, "Determining optical flow," *Artificial Intelligence*, vol. 17, pp. 185–203, 1981.
13. E. Hildreth, *The Measurement of Visual Motion*. Cambridge: MIT Press, 1983.
14. H. H. Nagel, "Displacement vectors derived from second order intensity variations in image sequences.," *Computer Vision, Graphics and Image Processing*, vol. 21, pp. 85–117, 1983.
15. M. Jones and T. Poggio, "Model-based matching by linear combination of prototypes," a.i. memo no., Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1996.
16. T. Vetter, M. J. Jones, and T. Poggio, "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, 1997.
17. P. Viola, "Alignment by maximization of mutual information," A.I. Memo No. 1548, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1995.
18. T. Vetter, "Synthestis of novel views from a single face image," *International Journal of Computer Vision*, no. in press.
19. P. Burt and E. Adelson, "Merging images through pattern decomposition," in *Applications of Digital Image Processing VIII*, no. 575, pp. 173–181, SPIE The International Society for Optical Engeneering, 1985.

This article was processed using the LaTeX macro package with LLNCS style