Face Identification by Fitting a 3D Morphable Model using Linear Shape and Texture Error Functions

Sami Romdhani, Volker Blanz, and Thomas Vetter

University of Freiburg, Institüt für Informatik,
Georges-Köhler-Allee 52, 79110 Freiburg, Germany,
{romdhani, volker, vetter}@informatik.uni-freiburg.de

Abstract This paper presents a novel algorithm aiming at analysis and identification of faces viewed from different poses and illumination conditions. Face analysis from a single image is performed by recovering the shape and textures parameters of a 3D Morphable Model in an analysis-by-synthesis fashion. The shape parameters are computed from a shape error estimated by optical flow and the texture parameters are obtained from a texture error. The algorithm uses linear equations to recover the shape and texture parameters irrespective of pose and lighting conditions of the face image. Identification experiments are reported on more than 5000 images from the publicly available CMU-PIE database which includes faces viewed from 13 different poses and under 22 different illuminations. Extensive identification results are available on our web page for future comparison with novel algorithms.

1 Introduction

Images of faces do not depend on identity only, unfortunately. They are also influenced by illumination, by the pose from which the face is viewed and by its expression. Hence a face analysis system is confronted with the perplexing task of separating different sources of image variation. The recent FERET tests revealed that recognition performance drops significantly when pose and/or directed light are present in the input images [16]. Solving these two problems has become a major research issue. In this paper we present a novel algorithm able to identify faces in presence of combined pose and illumination variation.

A common approach for addressing these problems is to build an explicit generative model of the variations of face images. Then, analysis of a novel face image requires the fitting of an image generated by the model to the novel image, thereby coding the novel image in terms of model parameters. This fitting problem is in fact an optimisation problem: Its goal is to find the model parameters which minimise the difference between the image produced by the model and the novel image. The minimisation can be performed by a standard gradient descent algorithm [15] [6] or by a Levenberg-Marquardt [17] which is computationally expensive as it computes at each iteration the gradient of the image difference. This predicament lead first Gleicher [12], then Sclaroff and Isidoro [20] and Cootes *et al.* [8] to derive an optimisation-based algorithm in the specific context of model registration. This new approach, referred to as *Image Difference Decomposition* (IDD), makes the assumption that there is a linear dependency between the image difference (that is, the difference between the model image

Appeared in the European Conference on Computer Vision, 2002

and the input image) and the parameter update. Gleicher [12] applies this method to recover the projective transformation parameters of a small image region under motion. Sclaroff and Isidoro [20] endeavour to track the non-rigid motion of an image region. Cootes *et al.* [8] use IDD to fit a 2D face model to a novel face image.

Based on a theory of the complete image formation process (Section 2 and 3), we show that, in fact, the error function depends non-linearly on the parameters and the linear assumption made by IDD holds only on a small domain of variations (and not on the general problem which interests us, that of combined variation of pose and illumination). To approximate the non-linear image variations, induced by only the azimuth angle, Cootes *et al.* [9] had to learn five linear relations. If the number on non-linear degrees of freedom increases (elevation angle, direction of light, etc.), the number of linear relations required would grow exponentially. To overcome this problem, we use a 3D model that is able to describe faces viewed under arbitrary pose and illuminated from any direction. To fit the model, we do not assume a linear relation as in IDD, but we exploit, in Section 4, the fact that the error function is a non-linear mixing of two linear parts: A shape error function which linearly depends on the shape parameters of the model and a texture error which also depends linearly on the texture parameters. Hence, a *Linear Shape and Texture* (LiST) fitting algorithm is proposed, in Section 5, which uses these two linear functions to iteratively update the shape and texture parameters. Fitting and identification experiments are presented, in Section 7, on more than 5000 images from the publicly available CMU-PIE database which show faces viewed from 13 different poses and under 22 different lighting.

2 Morphable Model

In this section we briefly outline the Morphable Model. A detailed description can be found in [6] and [7]. The 3D Morphable Face Model is a model of faces represented in 3D where shape information is separated from texture information. The shape and texture models are learnt from a set of 200 exemplar faces which were acquired by a 3D *Cyberware*$^{\mathrm{TM}}$ scanner. This scanner allows the sampling of texture with little interference from any external lighting. Hence texture values can be regarded as albedo values which are characteristic of identity only. Building a 3D Morphable Face Model requires to transform the shape and texture spaces into vector spaces for which any convex combination of exemplar shapes and textures describes a realistic human face. This is achieved by setting the exemplar faces in full correspondence with respect to a reference shape. Correspondences between all exemplar face and the reference face are established by an optical flow algorithm [2]. This produces a 3D deformation field for each exemplar faces which is used to warp the textures onto the reference shape yielding a shape-free texture. This scheme introduces a consistent labelling of vertices across the whole set of exemplar faces. The geometry of a face is represented by a shape vector $\mathbf{s} = (X_1,\ Y_1,\ Z_1,\ X_2, \ldots, Y_N,\ Z_N) \in \mathcal{R}^{3N}$ that contains the $X,\ Y,\ Z$ coordinates of its N vertices. For simplicity the $R,\ G,\ B$ texture values are sampled at the same N points, therefore the texture vector of a face is represented by $\mathbf{t} = (R_1,\ G_1,\ B_1,\ R_2, \ldots, G_N,\ B_N) \in \mathcal{R}^{3N}$. Realistic novel shape and texture vectors, $\mathbf{s}$ and $\mathbf{t}$ can be generated by convex combination of the M exemplars, $\mathbf{s}^{ex}$ and $\mathbf{t}^{ex}$:

$$\mathbf{s} = \sum_{i=1}^{M} a_i \mathbf{s}_i^{ex}, \qquad \mathbf{t} = \sum_{i=1}^{M} b_i \mathbf{t}_i^{ex}, \qquad \sum_{i=1}^{M} a_i = \sum_{i=1}^{M} b_i = 1 \tag{1}$$

To take advantage of the inherent regularity of human faces and reduce the dimensionality of the shape and texture spaces, a popular statistical technique, Principal Component Analysis, is used. PCA is applied separately on the shape and texture spaces, thereby ignoring the correlation between shape and texture as opposed to other techniques [8]. We describe the application of PCA to shapes, its application to textures is straightforward. After subtracting their mean, $\overline{\mathbf{s}}$, the exemplars are arranged in a data matrix $\mathbf{A}$ and the eigenvectors of its covariance matrix $\mathbf{C}$ are computed using the Singular Value Decomposition [18] of $\mathbf{A}$:

$$\begin{gathered}\overline{\mathbf{s}} = \tfrac{1}{M}\textstyle\sum_{i=1}^{M} \mathbf{s}_i^{ex}, \qquad \mathbf{a}_i = \mathbf{s}_i^{ex} - \overline{\mathbf{s}}, \qquad \mathbf{A} = (\mathbf{a}_1, \mathbf{a}_2, \ldots, \mathbf{a}_M) = \mathbf{U}\mathbf{\Lambda}\mathbf{V}^{\mathrm{T}}, \\ \mathbf{C} = \tfrac{1}{M}\mathbf{A}\mathbf{A}^{\mathrm{T}} = \tfrac{1}{M}\mathbf{U}\mathbf{\Lambda}^2\mathbf{U}^{\mathrm{T}}\end{gathered} \tag{2}$$

The M columns of the orthogonal matrix U are the eigenvectors of the covariance matrix $\mathbf{C}$, and $\sigma_i^2 = \frac{\lambda_i^2}{M}$ are its eigenvalues, where the λ_i are the elements of the diagonal matrix Λ, arranged in decreasing order. Now, instead of describing a novel shape and texture as a convex combination of exemplars, as in Equation 1, we express them as a linear combination of shape and texture eigenvectors, $\mathbf{s}_i$ and $\mathbf{t}_i$, respectively:

$$\mathbf{s} = \overline{\mathbf{s}} + \sum_{i=1}^{M-1} \alpha_i \, \sigma_{s,i} \, \mathbf{s}_i \qquad \mathbf{t} = \overline{\mathbf{t}} + \sum_{i=1}^{M-1} \beta_i \, \sigma_{t,i} \, \mathbf{t}_i, \tag{3}$$

To achieve dimensionality reduction, the last eigenvalues (which are very small compared to the first eigenvalues) are set to zero. Hence, Equation 3, is reformulated as:

$$\mathbf{s} = \overline{\mathbf{s}} + S\boldsymbol{\alpha}, \qquad \mathbf{t} = \overline{\mathbf{t}} + T\boldsymbol{\beta} \tag{4}$$

where the columns of S and T are the most significant eigenvectors $\mathbf{s}_i$ and $\mathbf{t}_i$, re-scaled by their standard deviation. The Morphable Model shape and texture coefficients $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ constitute a pose and illumination invariant low-dimensional coding of the identity of a face as they are derived from characteristics pertaining only to the identity (3D shape and albedo). They will be used to identify faces in Section 7.2.

An additional and important benefit of PCA is that it provides an estimate of the probability densities of the shapes and textures, assuming that these densities are Gaussian:

$$p(\mathbf{s}) \sim e^{-\frac{1}{2}\|\boldsymbol{\alpha}\|^2}, \qquad p(\mathbf{t}) \sim e^{-\frac{1}{2}\|\boldsymbol{\beta}\|^2}. \tag{5}$$

3 Face Rendering

The Morphable Face Model is a generative model which can produce photo-realistic face images. In this section, we summarise the graphical rendering process which transforms face shape and texture vectors, $\mathbf{s}$ and $\mathbf{t}$, into a face image I. For more information see [6], or [7]. In this section, we use a hat ˆ to denote entities pertaining to a single vertex.

Shape Transformation Let us denote by $\hat{\mathbf{s}}$, the (X, Y, Z) coordinates of a vertex of the shape $\mathbf{s}$. The first rigid transformation is a translation by $\hat{\mathbf{t}_{3d}}$ and a rotation about the axes X, Y, Z of angles ϕ, γ and θ. The translation $\hat{\mathbf{t}}_{3d}$ is such as it sets the origin at the centre of the 3D shape $\mathbf{s}$. We denote by $\hat{\mathbf{w}}$ the result of this transformation:

$$\hat{\mathbf{w}} = \hat{R}_\phi \hat{R}_\gamma \hat{R}_\theta \; (\hat{\mathbf{v}} + \hat{\mathbf{t}}_{3d}), \qquad \hat{\mathbf{t}}_{3d} = \sum_j^N \hat{\mathbf{s}}_j \tag{6}$$

The 2D image coordinates of the vertex, denoted by $\hat{\mathbf{s}}_{2d}$, are then determined using an orthographic projection (as opposed to [6]) which is legitimate when the distance from the camera to the rendered head is much larger than its dimension. In practise this introduces only a minor error. After the projection a 2D translation, $\hat{\mathbf{t}}_{2d}$ and a re-scaling of f are applied:

$$\hat{\mathbf{s}}_{2d} = f \; \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} \hat{\mathbf{w}} + \hat{\mathbf{t}}_{2d} \tag{7}$$

Equations 4, 6 and 7 can be expressed as one equation for the entire set of vertices, clarifying the dependency between the 2D image coordinates vector $\boldsymbol{s}_{2d}$ and the shape parameters $\boldsymbol{\alpha}$:

$$\mathbf{s}_{2d} = f \; P \; R \; (\overline{\mathbf{s}} + S \; \boldsymbol{\alpha} + \mathbf{t}_{3d}) + \mathbf{t}_{2d} \tag{8}$$

where $\mathbf{t}_{3d}$ and $\mathbf{t}_{2d}$ are translation vectors of length $3N$ and $2N$ respectively formed by concatenating $\hat{\mathbf{t}}_{3d}$ and $\hat{\mathbf{t}}_{2d}$. R is a $3N \times 3N$ block diagonal matrix which performs the rotation $R_\phi R_\gamma R_\theta$ of all N vertices. P is the $2N \times 3N$ orthographic projection matrix for the ensemble of vertices. Note that, for rendering, a test of visibility must still be performed using a z-buffer method [11].

Illumination and Colour Transformation We assume that the face is illuminated by ambient light and one directed light. In the same fashion as the shape transformation aforementioned, we denote by $\hat{\mathbf{t}}$ the R, G, B colour values of one vertex of the texture $\mathbf{t}$ and by $\hat{\mathbf{n}}$ its unit-length normal (computed by use of the shape $\hat{\mathbf{s}}$). The illumination effects are modelled using Phong illumination [11]:

$$\hat{\mathbf{t}}_l = (\hat{A}_{amb} + \langle \hat{\mathbf{n}}, \mathbf{l} \rangle \; \hat{A}_{dir}) \hat{\mathbf{t}} + k_s \; \langle \mathbf{v}, \hat{\mathbf{r}} \rangle^\nu \; \hat{A}_{dir} \; \mathbf{1}, \qquad \hat{\mathbf{r}} = 2 \langle \hat{\mathbf{n}}, \mathbf{l} \rangle - \mathbf{l}, \;\; \|r\| = 1 \tag{9}$$

where $\hat{\mathbf{t}}_l$ is the resulting colour of the vertex, $\hat{A}_{amb}$ is a diagonal matrix containing the R, G, B colour values of the ambient light, $\mathbf{l}$ is the direction of the light, $\hat{A}_{dir}$ the diagonal matrix holding its colour, k_s, the specular reflectance, ν, the angular distribution of the specularities of human skin (or *shininess*), $\mathbf{v}$, the viewing direction, $\mathbf{1}$, a vector of ones and $\mathbf{r}$ is the direction of maximum specular reflection.

The second colour transformation allows to render various types of images such as photograph from different sources, grey-scale images or paintings [6] [7]. It uses a diagonal matrix, G, storing the R, G, B colour gains, a vector $\hat{\mathbf{o}}$ containing the R, G, B offsets and a matrix $M(c)$ which interpolates between the original colour values and the overall luminance depending on the colour contrast c. The resulting colour of the vertex rendered in the image at the position $\hat{\mathbf{s}}_{2d}$ is $\hat{\boldsymbol{t}}_M$:

$$\hat{\boldsymbol{t}}_M = G \; M(c) \; \hat{\mathbf{t}}_l + \hat{\mathbf{o}}, \qquad M(c) = c \; I + (1 - c) \begin{pmatrix} 0.3 & 0.59 & 0.11 \\ 0.3 & 0.59 & 0.11 \\ 0.3 & 0.59 & 0.11 \end{pmatrix} \tag{10}$$

Again, the dependency of the rendered texture over the model parameters $\boldsymbol{\beta}$ is clarified by summarising Equations 4, 9 and 10 as one:

$$\boldsymbol{t}_M = (A_{amb} + A_{dir})\,(\bar{\mathbf{t}} + T\boldsymbol{\beta}) + \mathbf{a}_{spec} + \mathbf{o} \tag{11}$$

where A_{amb} and A_{dir} are a $3N \times 3N$ block diagonal matrix representing the effects of colour transformation, G and $M(c)$, and, respectively, the ambient light $\hat{A}_{amb}$ and the lambertian part of the directed lights; $\mathbf{a}_{spec}$ contains the specular part of the light and $\mathbf{o}$ the R, G, B offset.

Texture Warping The image of a face is produced by setting the colour of the vertices, $\boldsymbol{t}_M$, at the positions $\mathbf{s}_{2d}$. Naturally, this is performed only for the visible vertices. As the values of $\mathbf{s}_{2d}$ are non-discrete, a rasterisation is performed. Also, to obtain a smooth and continuous image, a 2D to 2D mapping is achieved, whereby the colour of pixels between vertices is interpolated using the topology provided by a triangular mesh. These three steps constitute a warping operation, denoted by the function $\mathcal{W}(\cdot,\cdot)$:

$$I = \mathcal{W}(\mathbf{s}_{2d}, \boldsymbol{t}_M) \tag{12}$$

4 Inversion of the Face Image Formation Process

In Section 2 and 3 the problem of the *generation* of a photo-realistic face image is described. Algorithms of this kind found many applications in the field of Computer Graphics. Here, we are interested in the inverse problem, that of *recovering* the shape, texture, rigid transformation, colour transformation and lighting parameters from one image of a face, I_{input}. This vision problem is generally referred to as fitting. In this section, the theory motivating a novel fitting algorithm is detailed and in the next section the fitting algorithm is formally described.

Explaining an input image in terms of model parameters is usually performed using an *Analysis-by-Synthesis* loop whereby, at each iteration, an image is generated using the current estimate of the model parameters, then the difference between the model image and the input image is computed, and an update of the model parameters which reduces this difference is sought for. This is a minimisation problem whose cost function is the image difference between the image rendered by the model and the input image. The image difference, $\delta\boldsymbol{t}$ can be computed on the reference grid obtained from the shape estimate:

$$\mathbf{t}_I = \mathcal{W}^{-1}(\mathbf{s}_{2d}, I_{input}) \qquad \delta\boldsymbol{t} = \boldsymbol{t}_M - \mathbf{t}_I \tag{13}$$

where $\mathbf{t}_I$ is a texture vector formed by the R, G, B values sampled from the input image at the pixels given by the current shape estimate $\boldsymbol{s}_{2d}$. Such an error function was introduced by the Image Difference Decomposition algorithm [12] and used to update the parameters using a linear equation:

$$\delta\boldsymbol{p} = A\,\delta\boldsymbol{t} \tag{14}$$

where $\boldsymbol{p}$ regroups the model parameters which are to be recovered and A is a matrix learnt off-line. However this linear dependency is an assumption which holds in a small domain of variation of I_{input}. In the presence of pose and illumination variation, the image formation process, described in Section 3, allows the expression of Equation 13 as:

$$\delta \boldsymbol{t} = \frac{\partial \mathbf{t}_M}{\partial \boldsymbol{\alpha}} \delta \boldsymbol{\alpha} + \frac{\partial \mathbf{t}_M}{\partial \boldsymbol{\beta}} \delta \boldsymbol{\beta} + \frac{\partial \mathbf{t}_M}{\partial \boldsymbol{p}_i} \delta \boldsymbol{p}_i - \frac{\partial \mathcal{W}^{-1}(\boldsymbol{s}_{2d}, I_{input})}{\partial \mathbf{s}_{2d}} \left(\frac{\partial \mathbf{s}_{2d}}{\partial \boldsymbol{\alpha}} \delta \boldsymbol{\alpha} + \frac{\partial \mathbf{s}_{2d}}{\partial \boldsymbol{p}_r} \delta \boldsymbol{p}_r \right) \tag{15}$$

Several of these Jacobi matrices are not constant due to the non-linear influence of the shape and the illumination on the texture (see Equation 9 and 11). This makes Equation 13 non-linear and the assumption leading to Equation 14 violated. The non-constancy of the derivatives obliges their re-computation at each iteration of the fitting algorithm. However, we can still benefit from the regularity of the problem by realising that some of the derivatives are relatively inexpensive to compute: If $\boldsymbol{\alpha}$ and $\boldsymbol{p}_i$ are maintained constant, $\frac{\partial \mathbf{t}_M}{\partial \boldsymbol{\beta}}$ is constant and can be computed at relatively low cost and used to update $\boldsymbol{\beta}$. Hence, $\delta\boldsymbol{\beta}$ is obtained by solving a simple system of linear equations. Similarly, $\boldsymbol{\alpha}$, $\boldsymbol{p}_i$ and $\boldsymbol{p}_r$ are updated separately while maintaining the other parameters constant. This process is repeated iteratively until global convergence is obtained. The recovery of each of the parameters is detailed in the next two sections. It should be noted that each parameter update is biased because the other parameters are kept constant. To overcome this problem a regularisation technique, detailed in Section 4.3, is used.

4.1 Shape Fitting

Updating the shape parameters $\boldsymbol{\alpha}$ from an image difference $\delta \boldsymbol{t}$ is difficult due to the combined influence of: (i) the shape error (ii) the normals which depends non-linearly on the shape and (iii) the texture error (see Equation 9 and 11). On the other hand, recovering the shape parameters $\boldsymbol{\alpha}$ from a shape error is easier: if f, ϕ, γ and θ are kept constant the relation between the shape $\mathbf{s}_{2d}$ and $\boldsymbol{\alpha}$ is linear (from Equation 8):

$$\frac{\partial \mathbf{s}_{2d}}{\partial \boldsymbol{\alpha}} = f \; P \; R \; S \tag{16}$$

So, the derivative is the 2D projection of the shape PCA matrix rotated and scaled by the current estimate of the pose and scale parameters, ϕ, γ, θ, and f. Note that this gradient is not learnt off-line nor numerically estimated on-line, it simply results from a matrix multiplication. Hence, updating $\boldsymbol{\alpha}$ from a shape error $\delta \mathbf{s}_{2d}$ requires only to solve a linear system of equations. Ideally the shape error would be directly obtained from the image (as for the texture error described in the next section). However, as the image does not contain any shape information, the shape error is estimated by applying an optical flow algorithm [2] between the input image and the model image rendered using the current parameters. We denote by $\mathbf{s}_{2d}^{of}$ the shape recovered by optical flow in the image frame.

Rotation, Translation and Scale Parameters Update Equation 16 is used to update the shape model parameter $\boldsymbol{\alpha}$ but is not appropriate to update the shape transformation parameters due to the non-linear interaction of the parameters. Therefore, to recover these parameters, a Levenberg-Marquardt optimisation [18] is used to minimise the geometrical error between $\mathbf{s}_{2d}^{of}$ and the model points [14]:

$$\arg \min_{f,\phi,\gamma,\theta,\boldsymbol{t}_{2d}} \| \mathbf{s}_{2d}^{of} - (f \; P \; R \, (\overline{\mathbf{s}} + S \; \boldsymbol{\alpha} + \mathbf{t}_{3d}) + \mathbf{t}_{2d}) \|^2 = (\tilde{f}, \tilde{\phi}, \tilde{\gamma}, \tilde{\theta}, \tilde{\boldsymbol{t}}_{2d}) \tag{17}$$

The parameters $\boldsymbol{\alpha}$ and $\mathbf{t}_{3d}$ are not optimised. The current value of $\boldsymbol{\alpha}$ is used and $\mathbf{t}_{3d}$ is uniquely determined by $\boldsymbol{\alpha}$ (as it is the centre of mass of $\boldsymbol{s}$). This optimisation is very fast as the search space is very low-dimensional (6).

Shape Model Parameters Update A Shape error, $\delta\mathbf{s}_{2d}$, is defined as the difference between the shape recovered by optical flow $\mathbf{s}_{2d}^{of}$ and the shape rendered using the new rotation and scaling parameters $\tilde{f}, \tilde{\phi}, \tilde{\gamma}, \tilde{\theta}, \tilde{\boldsymbol{t}}_{2d}$ and the current estimate of $\boldsymbol{\alpha}$. If the rotation and scaling parameters are kept constant, the dependency between $\delta\mathbf{s}_{2d}$ and $\boldsymbol{\alpha}$ is linear. Hence, the update of $\boldsymbol{\alpha}$ which explains the remaining of $\delta\mathbf{s}_{2d}$ (not accounted for by the rotation, translation and scaling) is obtained in a single step, by solving an over-constrained linear system of equations:

$$\delta\mathbf{s}_{2d} = f\ P\ R\ S\ \delta\boldsymbol{\alpha} \tag{18}$$

4.2 Texture Fitting

Illumination and Colour Transformation Update The update on $\boldsymbol{p}_i$ is performed while maintaining the other parameters constant. Unfortunately $\delta\boldsymbol{t}$ does not depend linearly on the $\boldsymbol{p}_i$ and, similarly to the $\boldsymbol{p}_r$ update aforementioned, we have to resort to a non-linear Levenberg-Marquardt minimisation [18] to recover the illumination and colour transformation parameters. The problem to be solved is:

$$\min_{A_{amb}, A_{dir}, G, c, \mathbf{o}} \parallel \delta\boldsymbol{t} \parallel^2 \tag{19}$$

with $\delta\boldsymbol{t}$ defined by Equation 13. The minimisation can be performed on the ensemble of the visible vertices or on a subset thereof to decrease computation time. Again, this optimisation is fast due to the low-dimensionality of the search space (11).

Texture Model Parameters Update If $\boldsymbol{\alpha}$, $\boldsymbol{p}_i$ and $\boldsymbol{p}_r$ are kept constant Equation 15 becomes:

$$\delta\boldsymbol{t} = (A_{amb} + A_{dir})\ T\ \delta\boldsymbol{\beta} \tag{20}$$

Similarly to the shape parameter update, the texture parameter update is obtained in a single step by solving an over-constrained linear system of equations performed by computing the pseudo-inverse of $(A_{amb} + A_{dir})\ T$. Again, to speed-up the processing, only a subset of the pixels can be used.

4.3 Model Constraints

The fitting algorithm exploits the fact that the shape and image differences depend linearly on, respectively, the shape and texture model parameters, if the other parameters are maintained constant. This restriction introduces a bias in $\frac{\partial\mathbf{s}_{2d}}{\partial\boldsymbol{\alpha}}$ and in $\frac{\partial I}{\partial\boldsymbol{\beta}}$. Therefore updating $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ solely based on Equations 18 and 20 would lead to inaccuracies. These equations must be traded off with the prior probabilities of the parameters given by Equation 5. In the remaining of this section, we discuss only the shape parameters. The same is performed on the texture parameters. Equation 18 solves the following problem:

$$\min_{\delta\boldsymbol{\alpha}} \|\delta\mathbf{s}_{2d} - A\ \delta\boldsymbol{\alpha}\|^2 \tag{21}$$

where the matrix A is equal to $f\ P\ R\ S$. Now, instead of solving this problem, we favour solving the following problem:

$$\min_{\delta\boldsymbol{\alpha}} \left(\|\delta\mathbf{s}_{2d} - A\ \delta\boldsymbol{\alpha}\|^2 + \eta\ \|\boldsymbol{\alpha}_{cur} + \delta\boldsymbol{\alpha}\|^2\right) \tag{22}$$

where $\boldsymbol{\alpha}_{cur}$ is the shape model parameter vector before the update. This formula is motivated by the fact that the prior probability of $\boldsymbol{\alpha} = \boldsymbol{\alpha}_{cur} + \delta\boldsymbol{\alpha}$ directly depends on its norm. The parameter η trades off the prior probability and the fitting quality. Applying singular value decomposition to A yields $A = Udiag(w_i)V^{\mathrm{T}}$, where w_i are the singular values of A. It can be shown that Equation 22 is minimal for:

$$\delta\boldsymbol{\alpha} = V\ diag(\frac{w_i}{w_i^2+\eta})\ U^{\mathrm{T}}\ \delta\mathbf{s}_{2d} - V\ diag(\frac{\eta}{w_i^2+\eta})\ V^{\mathrm{T}}\ \boldsymbol{\alpha}_{cur} \tag{23}$$

Note that (i) if $\eta = 0$, the update computed by Equation 23 is the same as the one obtained by Equation 18 and that (ii) computing the update using Equation 18 is not much more expensive as computing the update using Equation 23.

Value of $\boldsymbol{\eta}$ The value of η depends on the magnitude of $\|\delta\mathbf{s}_{2d}\|^2$ relative to the magnitude of $\|\delta\boldsymbol{\alpha}\|^2$ (Equation 22). As $\delta\boldsymbol{\alpha}$ also depends on η (Equation 23), it appears, at first, that these values must be tuned through an iterative process. However, $\|\delta\mathbf{s}_{2d}\|^2$ and $\|\delta\boldsymbol{\alpha}\|^2$ are related by the singular values of A:

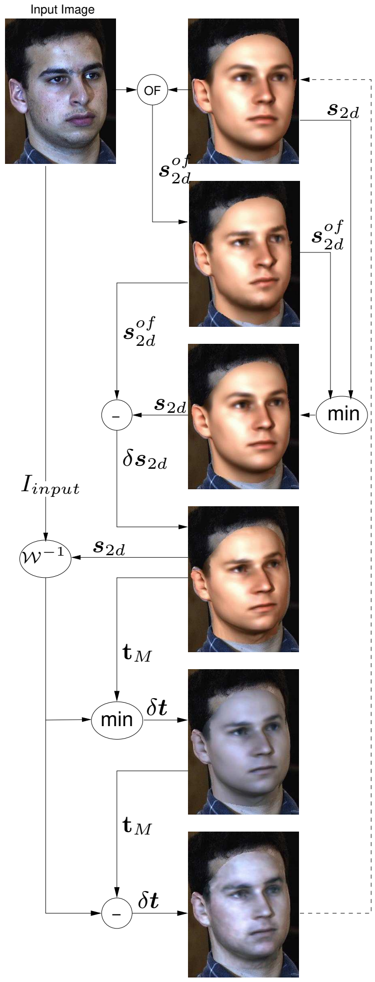
$$w_n^2 < \frac{\|\delta\mathbf{s}_{2d}\|^2}{\|\delta\boldsymbol{\alpha}\|^2} < w_1^2 \tag{24}$$

where w_n and w_1 are, respectively, the lowest and the largest singular values of A. Hence, the following algorithm is used to tune η: At the beginning of the fitting algorithm, A is strongly biased due to the fact that the other parameters (f, ϕ, γ and θ) are far from their optimum, hence we set η to the square of one of the first singular values. Then, as the other parameters reach their optimum, the confidence in A increases, and η is decreased on the spectrum of the singular values of A. More specific detail about this process is given in the next Section.

5 LiST - a Linear Shape and Texture Fitting Algorithm

Based on the inversion of the image formation process detailed in the previous section, we present, in this section, the fitting algorithm which is summarised in Figure 1. The key feature of this fitting algorithm is the exploitation of linear relations to update the shape and texture parameters. Therefore we refer to this algorithm as the *Linear Shape and Texture* (LiST) fitting algorithm.

1. Model Estimate: Before starting the algorithm, the parameters are initialised to their average values ($\boldsymbol{\alpha} = \boldsymbol{\beta} = 0$) except for the pose parameters f, ϕ, γ, θ and $\boldsymbol{t}_{2d}$, which must be estimated by use of other means (manually, using a face-pose detector or using additional 3D camera and head position information as done in our experiments (see Section 7)). Also, the direction of the light must be known as this parameter is not recovered by the algorithm. Based on these parameters, a model image is generated overlaying the input image. It is used as input of the optical flow algorithm.
2. Optical Flow: A coarse-to-fine optical flow algorithm [2] is performed to recover the shape deformation between the input image and the current model estimate. For the first iterations there is a large difference between the two images (mostly due to the fact that the illumination and colour parameters are not recovered yet). This may lead to inaccuracies in the optical flow algorithm. Therefore the algorithm is not applied on the raw RGB values but on the images filtered by a Laplacian kernel. Also, the accuracy of the flow recovered at one pixel depends on the magnitude of the Laplacian at that pixel. Therefore only the flow of the salient pixels whose Laplacian is higher than a threshold (the salient pixels) is used.



1. Model Estimate: This image shows the current model estimate overlaying the input image. It is used as one of the two input images of the optical flow algorithm.

2. Optical Flow: A coarse-to-fine optical flow algorithm is performed aiming at recovering the shape deformation, $\boldsymbol{s}_{2d}$, between the input image and the current model estimate

3. Rotation, translation and scale update: f, ϕ, γ, θ and $\boldsymbol{t}_{2d}$ are updated by solving:

$$\min_{f,\phi,\gamma,\theta,\boldsymbol{t}_{2d}} \|\mathbf{s}_{2d}^{of} - (fPR(\overline{\boldsymbol{s}} + S\boldsymbol{\alpha} + \boldsymbol{t}_{3d}) + \boldsymbol{t}_{2d})\|^2$$

using Levenberg-Marquardt.

4. Shape update: $\boldsymbol{\alpha}$ is updated by solving the following linear system of equations:

$$\delta \boldsymbol{s}_{2d} = (fPRS)\, \delta\boldsymbol{\alpha}$$

Equation 23 is used to solve the system.

5. Illumination & colour transformation update: Update A_{amb}, A_{dir}, G, c and $\mathbf{o}$ by solving:

$$\min_{A_{amb},A_{dir},G,c,\mathbf{o}} \|\delta\boldsymbol{t}\|^2$$

using Levenberg-Marquardt.

6. Texture update: $\boldsymbol{\beta}$ is updated by solving the following linear system of equations:

$$\delta\boldsymbol{t} = (A_{amb} + A_{dir})\, T\, \delta\boldsymbol{\beta}$$

Equation 23 is used to solve the system.

Figure 1. This figure visually demonstrate the six steps of the first iteration of the LiST fitting algorithm.

3. Rotation, translation and scale update: f, ϕ, γ, θ and $\boldsymbol{t}_{2d}$ are updated by minimising the difference between the shape recovered by optical flow and the model shape over the set of salient pixels. This is achieved by solving Equation 17. As the cost function is non-linear, the minimisation is carried out by a Levenberg-Marquardt algorithm [18].

4. Shape update: $\boldsymbol{\alpha}$ is updated by solving the linear Equation 18 by use of Equation 23. The value of η is chosen based on the iteration number. At the beginning of the algorithm, the shape derivative is strongly biased due to the fact that the other parameters are far from their optimum. Hence a strong weight on the prior probability must be chosen. Therefore, at the first iteration, η is set to $w_{i_1}^2$, where w_i are the singular values of the gradient matrix, $i_1 = 0.1\, n$ and n is the number of singular values (i.e. the number of shape parameters). At the second iteration,

$i_2 = 0.2 \cdot n$, and so forth. At the tenth iteration and thereafter, the cost function is not influenced by the prior probabilities anymore.

5. Illumination and colour transformation update: Similarly to step 3, A_{amb}, A_{dir}, G, c and $\mathbf{o}$ are updated by solving Equation 19 using a Levenberg-Marquardt algorithm [18].
6. Texture update: Similarly to step 4, the texture parameters $\boldsymbol{\beta}$ are updated by solving the linear Equation 20 by use of Equation 23. The selection of η is the same as explained at step 4. After this step, all the parameters have been successively updated and a new iteration is started at step 1. The iterative algorithm is stopped when $\|\delta \boldsymbol{t}\|^2$ obtained after this iteration becomes stationary.

6 Discussion

Building on our earlier 3D morphable model fitting using stochastic gradient descent (SGD) [5], we now developed LiST, an algorithm which takes advantage of the linear parts of the model. Applied to the problem of face recognition, we achieved with LiST a similar performance than the SGD algorithm in a more efficient manner (5 times faster). LiST departs from SGD in that it estimates different shape and texture errors which are used to update the shape and texture model parameters separately. The accuracy of the shape estimate depends on the goodness of the texture estimate and vice-versa. Therefore, it is paramount to interleave shape and texture update as the convergence of one helps the convergence of the other. The ability of LiST to recover out of the image plane rotation and directed illumination (see experiments in Section 7) originates from the fact that it is based on a model which closely approximates physical reality.

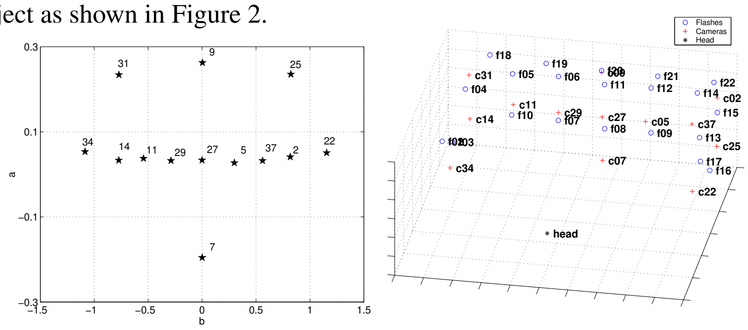
LiST bears some resemblance with the Active Shape Model (ASM) [10] and with the Vectorizer [3]. The ASM recovers the 2D shape of objects, represented as a sparse number of landmark points, using an equation similar to the Equation 18. Apart from the shape representation (dense physically-based 3D shape as opposed to sparse 2D shape), the main difference between LiST and ASM lies in the shape error estimation. As the ASM lacks a texture model, the shape is recovered by searching the image for edges along the normals of the 2D shape. By contrast, LiST uses the full estimate of the texture model to constraint the optical flow shape recovery, thereby using more image information than the ASM.

The Vectorizer [3] recovers the shape using optical flow between the input image and a model-generated image in a similar fashion to steps 1 and 2 of LiST. However, the Vectorizer lacks a shape model and hence the shape yielded by optical flow is not constrained by a shape prior probability as is achieved by LiST via Equations 18 and 23.

As mentioned in Section 4, the main difference between Image Difference Decomposition algorithms and LiST is the utilisation by LiST of an explicit shape error which allows us to take advantage of the linearities of the shape model to recover the shape parameters. On the other hand, in IDD the shape error is implicitly represented in the texture error which is, as explained in Section 4, a source of non-linearities. As these non-linearities are modelled by a constant Jacobi matrix, IDD is not designed to be used on a general problem such as face image fitting exhibiting pose and light variations. To model only one non-linear degree of freedom (the azimuth angle) Cootes *et al.* had to learn five constant gradient matrices [9]. In presence of combined pose and illumination variation the number of constant Jacobi matrices required to approximate the non-linearities would increase exponentially.

7 Experiments

In this sections, the fitting and identification performances are investigated on the Pose Illumination and Expression (PIE) database from CMU [21] which exhibits combined variations of pose and directed light. None of the 68 individuals in the PIE database is in the training set of 3D scans. We selected two portions of the database to carry out our tests: The first part includes pose variations at ambient light: each individual is imaged from 13 different poses (884 images overall). The second portion presents variation with respect to both pose and directed illumination and with an ambient light. Each individual is imaged at 3 poses and illuminated from 22 different directions (4488 images overall). The cameras and flashes are distributed in an hemisphere in front of the subject as shown in Figure 2.



(a) (b)

Figure 2. PIE database camera and light positions. (a) A plot of the azimuth and altitude angles of the cameras, along with the camera ID number. 9 of the 13 cameras sample a half circle at roughly head height ranging from a full left to a full right profile view(+/- 60 degrees); 2 cameras were placed above and below the central camera; and 2 cameras were positioned in the corners of the room. (b) 3D locations of the cameras, the flashes and the head. Courtesy of [21] and [13].

7.1 Fitting

Before fitting a face image, the face must be detected, its pose estimated and the direction of the light must be known. During execution of the algorithm, it refines the estimate of the face location and pose but leaves the approximation of the light direction constant. The initial value of all the other parameters is their average value. For the experiments on the PIE database, the face pose was estimated automatically from the geometrical information provided by the CMU database. Indeed, the 3D position of the cameras, the light sources and the head were recorded at the time of the data acquisition [21]. These data allow us to recover an initial estimate of the azimuth of the face (ϕ) and the direction of the light $\mathbf{l}$. The two other angles determining the pose of the face (θ and γ) were left to their average value (0). 2D translation and size parameters, $\boldsymbol{t}_{2d}$ and f, were estimated using the 2D positions of the two eyes and the mouth also provided by the maker of the database. As explained in [13], the 2D locations were automatically recovered using FaceIt's face finding module and manually verified.

On average, the algorithm converges in 8 iterations. The time required per iteration is less than a minute on a Pentium III 800 MHz. More than the half of this time is consumed in the optical flow computation. We anticipate an important speed up of the algorithm if we apply it in a coarse to fine approach, however the impact of this implementation on the algorithm performances must be carefully evaluated.

Figure 3 shows five fitting examples. The first line depicts the initial values of the parameters, the second represents the fitted model image and the last shows the original input images. Due to legal reasons, we are allowed to show images of very few individuals only. Unfortunately, this subset of individual is not representative of the whole database as many persons wore glasses and have facial hair.

Figure 3. Examples of Fitting. **Top row**: Initial parameters, **Middle row**: Fitting result, **Bottom row**: Input image. The fourth column is an example of bad fitting.

7.2 Identification

The fitting procedure provides shape and texture coefficient which, as explained in Section 2, depends of the identity only. Identification of a novel face image is performed by matching the coefficients recovered from the image with the coefficients recovered from the image of a known individual. The matching criterion used is a simple nearest neighbour classification rule using the following correlation-based similarity measure:

$$d = \frac{\langle \boldsymbol{\alpha}, \boldsymbol{\alpha}' \rangle}{\|\boldsymbol{\alpha}\| \; \|\boldsymbol{\alpha}'\|} + \frac{\langle \boldsymbol{\beta}, \boldsymbol{\beta}' \rangle}{\|\boldsymbol{\beta}\| \; \|\boldsymbol{\beta}'\|} \tag{25}$$

where the primed symbols pertain to the gallery image (known individuals) and the non-primed to the probe image (novel images). We use the *closed universe* model whereby every probe image shows an individual that can be found in the gallery. To enable past and future algorithm assessment, we made publicly available our identification results on our web page[1]: for each probe image we provide the ordered list of gallery images along with the distance d between the gallery images and the probe image.

[1] `http://graphics.informatik.uni-freiburg.de/publications/list`

Pose variations Here, the performances of the fitting algorithm are evaluated with respect to identification across 13 different poses with ambient light. The gallery contains a single image per individual at the same pose. Each pose is used in turn in the gallery. Table 1 shows the complete confusion matrix. Each of the 13×13 cells refer to an identification experiment of 68 probe images using a gallery of 68 individuals. The last column of the table shows the percentage of correct identification for a particular gallery view averaged on the entire probe set. The only reported results of this set of experiments (performed on the same images using the same initial parameters (eyes and mouth coordinates)) is in [13] using Visionics FaceIt, one of the most successful commercial face recognition system. FaceIt finished as the top performer in the Face Recognition Vendor Test 2000 [4]. Visionics claim that FaceIt can handle pose variation up to 35 degrees in all direction. Our identification results outperform FaceIt in all 169 cells. For front gallery and probe views, our algorithm outperforms FaceIt by a few percents and the improvement of our algorithm compared to FaceIt increases as the probe and gallery pose increase, averaging to more than 60 % correct identification augmentation. **These results clearly show that, for pose-invariant face identification, a 3D representation is valuable.**

azimuth	-66	-47	-46	-32	-17	0	0	0	16	31	44	44	62	
altitude	3	13	2	2	2	15	2	1.9	2	2	2	13	3	
Probe Pose	34	31	14	11	29	09	27	07	05	37	25	02	22	mean
Gallery Pose														
34	-	81	84	78	60	56	54	41	43	44	50	53	53	61
31	91	-	96	93	87	82	79	78	69	65	63	57	47	77
14	88	97	-	100	93	78	85	82	68	62	54	62	51	79
11	76	94	99	-	94	85	94	91	74	65	50	53	44	78
29	60	84	90	96	-	94	96	93	78	68	59	54	40	78
09	53	79	81	88	94	-	97	90	87	84	76	69	43	80
27	49	78	87	93	97	99	-	99	99	94	75	76	50	84
07	35	75	81	96	93	90	99	-	93	82	62	62	38	77
05	46	78	85	88	90	93	99	94	-	93	91	85	63	85
37	47	81	78	82	82	85	93	79	94	-	91	90	78	83
25	50	71	72	60	71	84	85	65	82	85	-	88	84	77
02	49	56	65	60	62	71	78	68	75	90	91	-	81	73
22	56	43	51	44	43	49	47	50	57	71	74	84	-	59

Table 1. Confusion table for pose variation. Each row of the confusion table shows the recognition percentage on each of the probe poses given a particular gallery pose. The camera position, indicated by its azimuth and altitude angles is shown in Figure 2. The table layout is courtesy of [13].

Combined pose & light variations Here, we used the portion of the PIE database containing 4488 images of faces of 68 individuals viewed from 3 poses (front (camera 27), side (cam. 5) and profile (cam. 22)) and illuminated from 21 different directions. Again only a single image per individual is used in the gallery. Each pose is used in turn in the gallery. We chose light number 12 for the gallery as the fitting is generally fair at that condition. Table 2 shows the confusion matrix for each gallery broken down into probe views and illuminations. It appears that the best generalisation performances are obtained with a side-view gallery. As expected, the poorest performances are obtained by a profile-view. This is probably due to the lack of features at that view: only one eye visible, no nostril edges, only half a mouth visible. The results for generalisation across light variation for similar or near-by poses (front and side) are promising. The decrease in identification performance for profile views is similar to that experienced with ambient light only (see previous paragraph). Therefore we conjecture that this performance drop is due more to pose variation than to light variation. To augment our understanding of the effects of combined pose-illumination variations on identification accuracy, an image database with more poses should be used.

	Front Gallery			Side Gallery			Profile Gallery		
light	Front	Side	Profile	Front	Side	Profile	Front	Side	Profile
ambient	97	84	48	88	91	49	57	72	87
2	71	60	22	50	69	31	29	49	51
3	95	78	28	84	87	26	63	54	54
4	98	83	45	85	96	47	54	51	66
5	98	91	65	94	97	65	51	53	81
6	98	89	65	94	100	65	65	62	84
7	98	92	65	96	99	71	57	65	87
8	98	94	48	99	100	71	63	78	96
9	100	97	57	100	100	75	69	88	100
10	98	89	58	93	99	71	60	60	78
11	100	97	72	97	100	82	66	75	93
12	-	98	78	99	-	91	75	78	-
13	98	97	77	100	100	91	82	85	100
14	98	98	83	99	100	91	82	90	100
15	98	97	80	97	100	93	81	90	100
16	95	94	71	99	99	84	87	93	100
17	97	89	75	91	94	87	76	85	100
18	97	85	58	88	96	60	43	47	78
19	97	86	54	88	99	57	51	50	81
20	100	97	72	99	100	75	59	71	85
21	100	98	58	100	100	81	74	79	97
22	97	97	78	97	100	90	74	90	100
mean	97	91	60	93	96	71	65	71	86

Table 2. Identification results in presence of combined pose and illumination variation. The first column lists the directed light number. The three groups of triple columns show the results obtained for three different galleries: Front, side and profile. For each gallery, the percentage of correct identification for a pose-illumination condition is displayed. See Figure 2 for flash light positions.

8 Conclusion

This paper presented LiST, a novel algorithm for fitting a 3D Morphable Model to face images. LiST exploits the linearity of the model in the recovery of shape and texture parameters leading to a robust and efficient algorithm. The shape parameters are recovered using linear equations from a shape error estimated by optical flow (using the input image and the current texture). The texture parameters are also computed using linear equations from a texture error estimated using the current shape. LiST was evaluated on the difficult problem of combined pose-illumination variation face identification. State of the art results were obtained. However they should be confirmed on a database containing more individuals. LiST opens up avenues for further research:

Initial parameters robustness: Now that the usefulness of our algorithm has been shown on face identification applications, we want to evaluate the performance of our

algorithm with respect to noisy initial parameter estimation.
Efficiency: An efficient way of speeding up the algorithm is to implement it in a coarse to fine fashion. We are planning to evaluate the impact of a coarse to fine implementation on the algorithm performances.
Illumination Recovery Recently, Basri *et al.* [1] and Ramamoorthi *et al.* [19] have shown that the diffuse part of the illumination was well approximated by a 9D linear space. We want to leverage these findings in our algorithm and recover the light parameters also using linear equations. Additionally, this method would alleviate the constraints on the initial light direction requirements.
Acknowledgement: We thank Simon Baker and Ralph Gross for providing us with PIE, a great face image database and for the support they gave us in the use of it. We also thank Nikos Canterakis for interesting discussions. This material is based upon work supported by the European Research Office of the US Army under contract No. N68171-01-C-9000.

References

1. Ronen Basri and David Jacobs. Lambertian reflectance and linear subspace. In *8th International Conference on Computer Vision*, volume 2, pages 383–390, 2001. **See section** 8
2. J.R. Bergen and R. Hingorani. Hierarchical motion-based frame rate conversion. Technical report, David Sarnoff Research Center Princeton NJ 08540, 1990. 2, 4.1, 2
3. D. Beymer and T. Poggio. Image representation for visual learning. *Science*, 272, 1996. 6
4. D. M. Blackburn, M. Bone, and P. J. Philips. Face recognition vendor test 2000: Evaluation report. Technical report, DoD Counterdrug Technology Development Program, 2000. 7.2
5. V. Blanz, S. Romdhani, and T. Vetter. Face identification across different poses and illuminations with a 3d morphable model. In *Proc. of the 5th IEEE Conf. on AFGR*, 2002. 6
6. V. Blanz and T. Vetter. A morphable model for the synthesis of 3D-faces. In *SIGGRAPH 99 Conference Proceedings*, Los Angeles, 1999. Addison Wesley. 1, 2, 3, 3, 3
7. Volker Blanz. *Automatische Rekonstruction der dreidimensionalen Form von Gesichtern aus einem Einzelbild*. PhD thesis, Universität Tübingen, Germany, 2001. 2, 3, 3
8. T. Cootes, G. Edwards, and C. Taylor. Active appearance model. In *ECCV*, 1998. 1, 2
9. T. Cootes, K. Walker, and C. Taylor. View-based active appearance models, 2000. 1, 6
10. T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham. Active shape models - their training and application. *Computer Vision and Image Understanding*, 1995. 6
11. J.D. Foley and A. van Dam. *Fundamentals of interactive computer graphics*. The systems programming series. Addison-Wesley, Reading, Ma, 1984. 3, 3
12. M. Gleicher. Projective registration with difference decomposition, 1997. 1, 4
13. Ralph Gross, Jianbo Shi, and Jeff Cohn. Quo vadis face recognition? In *Third Workshop on Empirical Evaluation Methods in Computer Vision*, 2001. 2, 7.1, 7.2, 1
14. R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000. 4.1
15. M. Jones and T. Poggio. Multidimensional morphable models: A framework for representing and matching object classes. In *Proceedings of the Sixth ICCV*, 1998. 1
16. P. J. Phillips, P. Rauss, and S.Der. Feret (face recognition technology) recognition algorithm development and test report. ARL-TR 995, U.S. Army Research Laboratory, 1996. 1
17. F. Pighin, R. Szeliski, and D.H. Salesin. Resynthesizing facial animation through 3d model-based tracking. In *Proceedings of the 7th ICCV*, pages 143–150, 1999. 1
18. Vetterling Press, Teukolsky and Flannery. *Numerical recipes in C : the art of scientific computing*. Cambridge University Press, Cambridge, 1992. 2, 4.1, 4.2, 3, 5
19. Ravi Ramamoorthi and Pat Hanrahan. A signal-processing framework for inverse rendering. In *SIGGRAPH 2001 Conference Proceedings*, pages 117–128, 2001. 8
20. S. Sclaroff and J. Isidoro. Active blobs. In *6th ICCV*, 1998. 1
21. T. Sim, S. Baker, and M. Bsat. The cmu pose, illumination and expression (pie) database of human faces. Technical Report CMU-RI-TR-01-02, CMU, 2000. 7, 2, 7.1