

Singular Value Decomposition, Eigenfaces, and 3D Reconstructions*

Neil Muller[†]
Lourenço Magaia[†]
B. M. Herbst[†]

Abstract. Singular Value Decomposition (SVD) is one of the most important and useful factorizations in linear algebra. We describe how SVD is applied to problems involving image processing. In particular, how SVD aids the calculation of so-called eigenfaces, which provide an efficient representation of facial images in face recognition. Although the eigenface technique was developed for ordinary grayscale images, the technique is not limited to these images. Imagine an image where the different shades of gray convey the physical three-dimensional structure of a face. Although the eigenface technique can again be applied, the problem is finding the three-dimensional image in the first place. We therefore also show how SVD can be used to reconstruct three-dimensional objects from a two-dimensional video stream.

Key words. singular value decomposition, facial recognition, 3D reconstruction

AMS subject classifications. 65-01, 65F15, 62H25, 68T10, 68U10

DOI. 10.1137/S0036144501387517

I. Introduction. The human face is probably one of the most familiar sights on earth. Yet it is relatively new. *Homo sapiens* with its modern human characteristics appeared only about 150,000 years ago. Its current dominating position on earth dates back to only about 40,000 ago with the demise of the Neanderthals [8, 18].

From an animal perspective the human face is a curious object: it is rather flat, without a muzzle, and basically hairless. On this flat, hairless background are arranged eyes, a mouth, a prominent nose, and a chin. Some of these characteristics make perfect sense, others remain a puzzle. Why, for instance, the peculiar downward pointing, ridge-like nose? Is it a leftover from an aquatic origin, as some will have it [19]? And what purpose serves the chin? Hairlessness can be explained: it facilitates the communication of facial expressions, which in turn are reflections of the feelings of the individual. The subtlety of emotions requires a wide range of expressions that is possible only through a special arrangement—a complicated system of facial muscles attached directly to the skin. Of course, the human face, with its multitude of expressions, has always fascinated artists. In fact, some of the first detailed anatomical studies of humans can be traced to artists when they started to reject the stylized Gothic portraiture of the fifteenth century in favor of a more realistic approach to what people actually look like. This found early culmination in the work of Leonardo da Vinci, who studied human anatomy not only for the sake of art but also as a

*Received by the editors April 6, 2001; accepted for publication (in revised form) May 28, 2004; published electronically July 30, 2004.

<http://www.siam.org/journals/sirev/46-3/38751.html>

[†]Department of Applied Mathematics, University of Stellenbosch, Matieland, Stellenbosch, 7602 South Africa (neil@dip.sun.ac.za, magaia@dip.sun.ac.za, herbst@dip.sun.ac.za).

scientific interest [13]. The interest of the modern scientist in the interaction of the facial structure with light, its moods and expressions, is no less than that of the artist, for it is precisely these characteristics that distinguish one individual from another. For a fascinating account of the human face, see [18].

The variations under different lighting conditions and the changing moods and expressions of the face pose severe challenges to any facial recognition system—so severe that humans apparently have a special section of their brain dedicated to this function. The challenge to scientists is irresistible, and many automated systems have been developed. Our personal favorite is based on so-called eigenfaces, conceptually a simple idea going back to Sirovich and Kirby [26]. The two-dimensional (2D) $p \times q$ grayscale images span an $m = pq$ -dimensional vector space. The idea behind eigenfaces is to find an orthonormal basis for the subspace containing all the *facial* images. The beauty of this concept lies in the fact that this subspace is of surprisingly low dimension, something on the order of 100, resulting in an enormous reduction in dimension independent of m , the resolution of the image. Very efficient systems have been developed based on this key idea, most notably by the Media Laboratory at MIT; see [22, 23, 24, 33].

One of the problems that any facial recognition system must confront is the fact that different images of the same face can vary enormously with respect to size, position, orientation, lighting conditions, and facial expression. Thus it may be hard to compare different images of the same face under different conditions. Facial recognition systems in general tend to include a preprocessing stage where the image is *normalized* with respect to the above-mentioned variations; i.e., the image is transformed to some standard size, position, etc. Size and position are relatively easy to deal with but the others are troublesome. Consider out-of-plane rotations: the extreme case would be a 180° rotation where only the back of the head is visible and all facial information is lost. In practice systems allow a limited amount of variation from a full frontal view. Other significant issues are how to normalize the illumination and neutralize facial expression. One possibility is to obtain a database of different illumination and expressions. Using techniques very much like the ones described in this paper, it is possible to compensate for the variations.

Some of the difficulties mentioned above are a direct consequence of working with projections of faces onto a 2D imaging medium. Note that in these images all information about the facial structure is indirect: Light interacts with the face, and this interaction is captured on film or other imaging media. Of course, the final image reflects the facial structure, as is evident from our, albeit limited, ability to reconstruct three-dimensional (3D) objects from their grayscale images (see, for example, [32]), but the final image remains highly dependent on the position and strength of the light source. Why then not directly work with a 3D image? It is possible to represent a 3D structure with a grayscale image, where the shades of gray have a direct geometric meaning. This permits us to use 2D techniques but sidestep the illumination problem—we still work with a grayscale image, but now the different shades of gray describe structure.

Although this makes perfect sense from a theoretical point of view, the most serious practical problem is obtaining the 3D image. Various approaches have been tried. The first is the shape-from-shading method in which a 3D reconstruction is attempted from a 2D grayscale image. This problem is ill-posed since many images give the same 2D projections, forcing one to impose external restrictions on the reconstruction. Another method imitates the human visual system and attempts a 3D

reconstruction from stereo images. Parenthetically, is it known to what extent the human recognition system depends on its stereo vision, for example, under adverse lighting situations? Another closely related method involves the 3D reconstruction from a stream of video images. Mathematically, the reconstruction is simple; for stereo images, elementary projective geometry does the trick, and for reconstructions from streams of video images, singular value decomposition (SVD) suffices (provided that a simple camera model is used, as explained below). The difficulty lies in the image processing—a large number of features must be located very accurately in all the images in order to compute a *disparity* map, i.e., the relative offset of the individual features from one frame to the next.

Since large parts of the face are practically featureless, it is clear that the problem is a challenging one. Note the ease with which this is accomplished by the human mind—the same laws that determine automated 3D reconstruction with a computer also apply to the human visual system. In particular, the human visual system also requires a disparity map, and evidence supports the belief that the human mind pointwise matches the images captured on the retinas of our two eyes. The computational complexity is staggering; see [25].

Of course, even 3D reconstruction does not address the problem of facial expression or the presence of a beard, mustache, or glasses.

At this point the reader will understand how difficult it is to develop an automated facial recognition system. Since there exist alternative biometric identification systems for iris scans, fingerprints, voice-prints, signatures, etc., one might well ask whether facial recognition systems are actually needed. There are situations however, where facial recognition remains the only viable method. For example, since it is a passive system requiring no cooperation from the subject it is particularly suitable for unobtrusively monitoring the movement of individuals in specific security-sensitive situations. In addition, since facial recognition is a familiar way for humans to recognize one another, it allows for easy operator intervention should that be required. It is not surprising that one more and more facial recognition systems implemented for practical applications.

In this paper we explain the basic idea behind two real-world applications: (i) facial recognition using eigenfaces, and (ii) 3D reconstructions of objects from streams of video data. In both cases SVD provides the key. Among the great matrix factorizations (like LU and QR), SVD is arguably the greatest. From a theoretical point of view, it summarizes many fundamental results of matrix linear algebra; see, for example, [27, 31]. There is no space for a comprehensive discussion of SVD in this paper. To improve the readability, however, and to keep the paper self-contained, we give a brief overview of how the fundamental results of matrix linear algebra follow directly from SVD; these results are fundamental to many different applications, not only those discussed in this paper. For example, our first application—facial recognition using eigenfaces—depends on the remarkable ability of SVD to reduce the dimension of the feature vector space. The second application on the reconstruction of 3D objects from streams of video data exploits the ability of SVD to reveal the effective rank of a matrix.

A sample set of facial images, divided into training and test sets, as well as some simple Matlab codes that illustrate both applications, are made available [6]. Note that the material includes a file, `Tcodes.tar`, with a number of useful Matlab codes, developed specifically for instruction at MIT and The MathWorks (see also [7]).

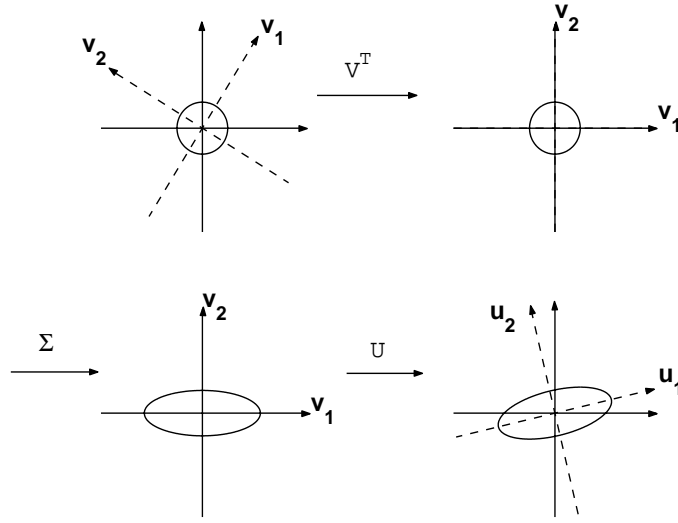


Fig. 1 *The geometrical meaning of SVD: The image of a circle under matrix multiplication is an ellipse.*

2. Singular Value Decomposition. We briefly review SVD and several of its properties that are important for the applications that follow. Throughout the paper, the discussion is restricted to real matrices. Among excellent references on SVD are [5, 12, 27, 31].

The SVD of a matrix X of size $m \times n$ is

$$(1) \quad X = U\Sigma V^T,$$

where U and V are $m \times m$ and $n \times n$ orthonormal matrices. Σ is an $m \times n$ diagonal matrix with the nonnegative singular values σ_j , $j = 1, \dots, \min(m, n)$, arranged in nonincreasing order along the diagonal. The columns of U and V are denoted by the vectors \mathbf{u}_j , $j = 1, \dots, m$, and \mathbf{v}_j , $j = 1, \dots, n$. An important geometrical interpretation of SVD is given in Figure 1 for $m = n = 2$: The image of the unit sphere under any $m \times n$ matrix multiplication is an ellipse. Considering the three factors of the SVD separately, note that V^T is a pure rotation of the circle. Figure 1 shows how the axes \mathbf{v}_1 and \mathbf{v}_2 are first rotated by V^T to coincide with the coordinate axes. Second, the circle is stretched by Σ in the directions of the coordinate axes to form an ellipse. The third step rotates the ellipse by U into its final position. Note how \mathbf{v}_1 and \mathbf{v}_2 are rotated to end up as \mathbf{u}_1 and \mathbf{u}_2 , the principal axes of the final ellipse. A direct calculation shows that $X\mathbf{v}_j = \sigma_j\mathbf{u}_j$. Thus \mathbf{v}_j is first rotated to coincide with the j th coordinate axis, stretched by a factor σ_j , and then rotated to point in the direction of \mathbf{u}_j . All of this is beautifully illustrated for 2×2 matrices by the Matlab code `eigshow.m` (see [6, 7]).

A direct consequence of the geometric interpretation is that the largest singular value, σ_1 , measures the “magnitude” of X (its 2-norm):

$$(2) \quad \|X\|_2 = \sup_{\|\mathbf{x}\|_2=1} \|X\mathbf{x}\|_2 = \sigma_1.$$

This means that $\|X\|_2$ is the length of the longest principal semiaxis of the ellipse.

Expressions for U, V , and Σ follow readily from (1),

$$(3) \quad XX^T U = U \Sigma \Sigma^T \quad \text{and} \quad X^T X V = V \Sigma^T \Sigma,$$

demonstrating that the columns of U are the eigenvectors of XX^T and the columns of V are the eigenvectors of $X^T X$.

The rank of X is the number of nonzero singular values. Thus if $\text{rank}(X) = r$, it is possible to rewrite the SVD in its reduced form,

$$(4) \quad X = U_+ \Sigma_+ V_+^T,$$

where Σ_+ is the $r \times r$ diagonal matrix with the r nonzero singular values of X , and U_+ and V_+ consist of the first r columns of U and V , respectively. It is straightforward to prove that the first r columns of U form an orthonormal basis for the column space of X , the first r columns of V form an orthonormal basis for the row space of X , the remaining $m - r$ columns of U form an orthonormal basis for the left null-space of X , and the last $n - r$ columns of V form an orthonormal basis for the null-space of X . Thus we arrive at the following important result:

- (a) *The column and row spaces of X both have dimension r . The null-space of X has dimension $n - r$, and the left null-space of X has dimension $m - r$.*
- (b) *The null-space of X is the orthogonal complement of the row space in \mathbb{R}^n . The left null-space of X is the orthogonal complement of the column space in \mathbb{R}^m .*

The construction of eigenfaces depends on these results. The 3D reconstruction depends on SVD to reveal the effective rank of a matrix. To wit, note that X can be written as the sum of r rank-1 matrices,

$$(5) \quad X = \sum_{j=1}^r \sigma_j \mathbf{u}_j \mathbf{v}_j^T.$$

This implies that the zero singular values may be ignored since they carry no “information.” What about singular values close to zero? Again the answer can be based on a geometric description. If one wants to approximate a hyperellipsoid with a line segment, the best one can do is to take the line segment as the longest axis of the ellipsoid. If one takes the longest and second longest axes of the ellipsoid, one gets the best approximation by a 2D ellipse, and so on. More precisely, let us approximate X with the ν -dimensional ellipsoid,

$$(6) \quad X_\nu = \sum_{j=1}^{\nu} \sigma_j \mathbf{u}_j \mathbf{v}_j^T.$$

Each term is associated with one of the principal directions \mathbf{u}_j of the hyperellipsoid. Since the difference $X - X_\nu$ is a matrix with singular values $\sigma_{\nu+1}, \dots, \sigma_r$, the 2-norm is again given by the largest singular value,

$$(7) \quad \|X - X_\nu\|_2 = \sigma_{\nu+1}.$$

In this expression $\sigma_{\nu+1} = 0$ if $\nu > r$. Indeed, it can be shown that X_ν is the best approximation to X (in the 2-norm sense) over all matrices with $\text{rank} \leq \nu$. If $\sigma_{\nu+1}$ is sufficiently small, it is safe to keep only ν singular values. In this case, we say that the effective rank of X is ν . What is meant by “sufficiently small” is problem specific. To jump ahead a little, for facial recognition we may wish to take “small” as indicating either visually perfect reconstruction, as in Figure 6(c), or just retaining sufficient information for identification purposes.

3. SVD and Covariances. We now approach SVD from a slightly different point of view. Assume that we are given n vectors \mathbf{x}_j , $j = 1, \dots, n$, each of dimension m , and zero average, $\mathbf{a} = \frac{1}{n} \sum_{j=1}^n \mathbf{x}_j = \mathbf{0}$. If the average is not zero, as in Figure 2, it can always be removed by subtracting the average \mathbf{a} from all the vectors.

Before, we were primarily interested in finding an orthonormal basis for the subspace spanned by these vectors. Instead, let us now investigate the “correlation” between these vectors. Geometrically, we ask for the direction that best approximates the distribution of the vectors. If we imagine a cloud of points inside a region of the 2D plane (including the origin, since we assume a zero average), we look for the direction of maximum variation. For data with nonzero average, subtracting the average amounts to looking for the direction of maximum variation relative to the average (\mathbf{u}_1 in Figure 2). Mathematically, we are searching for the direction \mathbf{u} such that

$$(8) \quad \mu = \max_{\|\mathbf{u}\|_2=1} \frac{1}{n} \sum_{j=1}^n (\mathbf{u}^T \mathbf{x}_j)^2.$$

Introducing the *covariance* matrix

$$(9) \quad C = \frac{1}{n} \sum_{j=1}^n \mathbf{x}_j \mathbf{x}_j^T,$$

it is possible to rewrite (8) as

$$(10) \quad \mu = \max_{\|\mathbf{u}\|_2=1} (\mathbf{u}^T C \mathbf{u}).$$

Since C is real and symmetric, it can be diagonalized with an orthonormal matrix U_C ,

$$(11) \quad C = U_C \Lambda_C U_C^T.$$

Here $\Lambda_C = \text{diag}(\lambda_1, \dots, \lambda_m)$, with the λ_j in decreasing order. (For an exercise, we leave it to the reader to show that all λ_j are nonnegative.) Thus,

$$(12) \quad \mu = \max_{\|\mathbf{u}\|_2=1} (\mathbf{u}^T U_C \Lambda_C U_C^T \mathbf{u}) = \max_{\|\mathbf{y}\|_2=1} (\mathbf{y}^T \Lambda_C \mathbf{y}).$$

Since U_C is orthonormal and $\mathbf{y} = U_C^T \mathbf{u}$, we know that $\|\mathbf{y}\|_2 = \|\mathbf{u}\|_2 = 1$. Thus we must calculate

$$(13) \quad \mu = \max_{\|\mathbf{y}\|_2=1} \sum_{j=1}^m \lambda_j |y_j|^2$$

subject to

$$(14) \quad \sum_{j=1}^m |y_j|^2 = 1.$$

Hence, $\mu = \lambda_1$ and $\mathbf{y} = \mathbf{e}_1 = [1 \ 0 \ 0 \ \dots \ 0]^T$. Since $\mathbf{u} = U_C \mathbf{y} = U_C \mathbf{e}_1$, the direction of maximum variation \mathbf{u} is exactly the eigenvector \mathbf{u}_1 of the covariance matrix C ,

belonging to the largest eigenvalue λ_1 . Moreover, λ_1 measures the variation in the direction of \mathbf{u}_1 .

Asking for the direction of maximum variation orthogonal to \mathbf{u}_1 , we must calculate (13), but this time subject to both (14) and $\mathbf{e}_1^T \mathbf{y} = 0$. This implies that the first component of \mathbf{y} equals zero, i.e., $y_1 = 0$. It then follows that (13) is calculated subject to $\sum_{j=2}^m |y_j|^2 = 1$. This is achieved by $\mathbf{y} = \mathbf{e}_2 = [0 \ 1 \ 0 \ \cdots \ 0]^T$ or $\mathbf{u} = U_C \mathbf{e}_2 = \mathbf{u}_2$, the eigenvector of C belonging to the second largest eigenvalue.

Continuing in this fashion we conclude: The first eigenvector of the covariance matrix C points in the direction of maximum variation, and the corresponding eigenvalue measures the variation in this direction; i.e., it is the *variance* in this direction. The subsequent eigenvectors point in the directions of maximum variation orthogonal to the previous directions, and the eigenvalues again measure the variations.

Finally, note that the same results are obtained by calculating the SVD of the matrix

$$(15) \quad X = \frac{1}{\sqrt{n}} [\mathbf{x}_1 \ \cdots \ \mathbf{x}_n].$$

The directions of maximum variation are then given by the columns of U and the variances (eigenvalues λ_j of C) λ_j by the squares of the singular values, i.e., $\lambda_j = \sigma_j^2$. In probability theory the singular values are known as the *standard deviations*. We illustrate these ideas with an example.

Example. Let us consider the problem of finding a face in an image. One possible (rather naive) approach would be to identify a number of objects that might qualify and to measure their width and height. It is easy to imagine that the measurements of any face should fall within a certain range. Anything outside the “typical” range can be discarded. Anything inside the range can then be investigated in more detail for further face-like characteristics. The problem is to find the “typical” range of measurements for faces.

Although this example is contrived—the number of features we measure is too low—it is not entirely unrealistic. In fact, one of the earliest identification systems, and at the time a serious rival for fingerprints, was based on a comprehensive measurements of individuals [2]. The system developed by Alphonse Bertillion in France during the 1870s employed eleven separate measurements of an individual: height, length, and breadth of head, length and breadth of ear, length from elbow to end of middle finger, lengths of middle and ring fingers, length of left foot, length of the trunk, and length of outstretched arms from middle fingertip to middle fingertip. Apart from being able to distinguish between different individuals, it also allowed a classification system that enabled Bertillion to quickly locate the file of a criminal, given just the measurements. The system was so successful that France was one of the last countries to adopt fingerprints for personal identification; see [2].

Figure 2 shows 48 actual measurements of a number of faces from a student population. The width is measured from ear-to-ear and the height from the chin to between the eyes. As you might have guessed, the measurements are distributed around a mean. We are interested in describing this distribution. The mean of the 48 measurements $\mathbf{x}_j = [x_j, y_j]^T$ is given by $\mathbf{a} = \frac{1}{48} \sum_{j=1}^{48} \mathbf{x}_j$ and the matrix X is defined by

$$X = \frac{1}{\sqrt{48}} [\mathbf{x}_1 - \mathbf{a}, \ \dots, \ \mathbf{x}_{48} - \mathbf{a}].$$

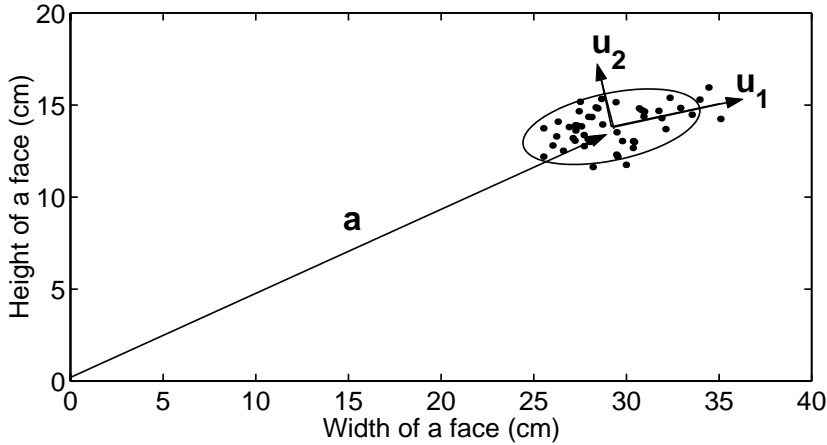


Fig. 2 The distribution of width and height measurements of a number of faces.

The reduced SVD, $X = U_+ \Sigma_+ V_+^T$, gives us the desired information. The two columns of U_+ give us the directions of maximum variation and the two singular values the magnitudes of the standard deviations in these two directions,

$$(16) \quad \begin{aligned} U_+ &= [\mathbf{u}_1 \quad \mathbf{u}_2] = \begin{bmatrix} 0.9778 & -0.2095 \\ 0.2095 & 0.9778 \end{bmatrix}, \\ \Sigma_+ &= \begin{bmatrix} \sigma_1 & 0 \\ 0 & \sigma_2 \end{bmatrix} = \begin{bmatrix} 2.43 & 0 \\ 0 & 0.91 \end{bmatrix}. \end{aligned}$$

The distribution described by U_+ and Σ_+ is illustrated by the ellipse shown in Figure 2. The ellipse is centered on the mean \mathbf{a} and its principal axes are given by the columns $[\mathbf{u}_1 \quad \mathbf{u}_2]$ of U_+ . Its principal semi-axes are $2\sigma_1$ and $2\sigma_2$ in order to enclose most of the measurements; a more detailed explanation follows below.

Let us now suppose that we are given the width and height measurements of an object in an image and are asked to decide whether this object can possibly be a face. It should be clear that one cannot simply use the distance from the mean—the shape of the ellipse clearly gives a better indication of the *nature* of the distribution. Intuitively we understand that the further a measurement is away from the ellipse, the less chance it has of being a face. One can be a little more precise by assuming that the measurements are *normally* distributed, a quite reasonable assumption. For our 2D measurements this means that we assume the distribution,

$$p(\mathbf{x}) = \frac{1}{2\pi|C|^{\frac{1}{2}}} \exp\left(-\frac{1}{2}(\mathbf{x} - \mathbf{a})^T C^{-1}(\mathbf{x} - \mathbf{a})\right),$$

where $C = XX^T$ is the covariance matrix we used before and $|C|$ its determinant. This means that the probability that a measurement falls inside a region denoted by A is given by

$$P = \int_A p(\mathbf{x}) d\mathbf{x}.$$

In particular, if we want to find the probability of a measurement inside the k standard deviation ellipse, we integrate over that ellipse centered at \mathbf{a} . A change of variables

simplifies the problem. To wit, first we factorize C into two 2×2 matrices, $C = SS^T$. We explain how to do this in section 7; for now, note that although C is also given by the product matrix $C = XX^T$, S and X are not the same. Factorizations of this type are not unique. The change of variables $\mathbf{x} = S\mathbf{y}$ then leads to the following expression for the probability,

$$P = \frac{1}{2\pi} \int_{C^k} \exp\left(-\frac{1}{2}\mathbf{y}^T\mathbf{y}\right) d\mathbf{y},$$

where C^k denotes the circle of radius k centered at the origin. This can be integrated numerically for the ellipse shown in Figure 2, for example ($k = 2$), to find that about 86% of the measurements fall inside the two standard deviation ellipse. Said differently, if a measurement falls on, or just outside of, the ellipse shown in the figure, the chances that it is a face is less than 14%—probably worth our while to do a more detailed investigation. If, on the other hand, the measurement falls outside the three standard deviation ellipse, the chances that it is a face is less than 1.2%. Clearly, the further a measurement falls outside the ellipse, the less chance it has of actually representing a face. In this way it might be possible to discard a large number of objects with measurements sufficiently far outside the ellipse (depending how careful we need to be not to miss out on any possible faces), thereby avoiding further time-consuming detailed investigations.

4. Eigenfaces. The idea behind the eigenface technique is to extract the relevant information contained in a facial image and represent it as efficiently as possible. Rather than manipulating and comparing faces directly, one manipulates and compares their representations.

Assume that the facial images are represented as 2D arrays of size $m = p \times q$. Obviously, m can be quite large, even for a coarse resolution such as 100×100 , $m = 10,000$. By “stacking” the columns, we can rewrite any $m = p \times q$ image as a vector of length m . Thus, we need to specify m values in order to describe the image completely. Therefore, all $p \times q$ sized images can be viewed as occupying an $m = pq$ -dimensional vector space. Do facial images occupy some lower-dimensional subspace? If so, how is this subspace calculated? In the previous sections SVD was used to find an orthonormal basis for the column space of a matrix. We shall use this technique again.

Consider n vectors with m components, each constructed from a facial image by stacking the columns. This is the training set and the individual vectors are denoted by \mathbf{f}_j , where $j = 1, \dots, n$. Obviously, it is impossible to study every single face on earth, so the training set is chosen to be representative of all the faces our system might encounter. This does not mean that all *faces* one might encounter are included in the training set; we merely require that all faces are adequately *represented* by the faces in the training set. For example, it is necessary to restrict the deviation of any individual face from the average face. Some individuals may be so unique that our system simply cannot cope. See [21] for a detailed analysis of issues relating to training sets. Obviously, the training set must be developed with care. Also, typically $n \ll m$.

As described in section 1, one should ensure that the faces are normalized with respect to position, size, orientation, and intensity. All faces must have the same size, be at the same angle (upright is most appropriate), and have the same lighting intensity, etc. This requires some nontrivial image processing (see [14]). Assume that all of this is done.

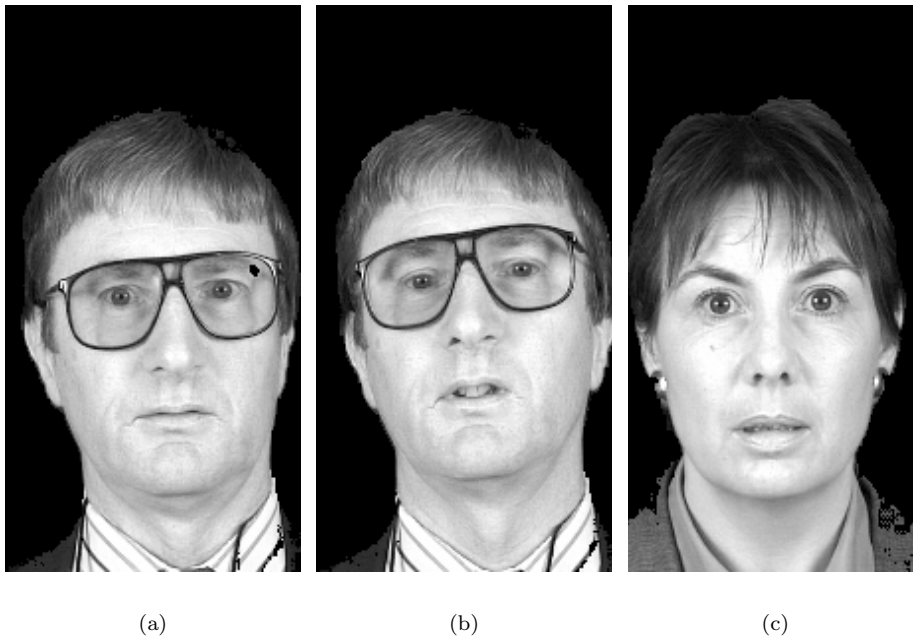


Fig. 3 *Three sample faces. (a) Image in the training set. (b) Same individual as (a), but the image not in the training set. (c) Image of an individual not in the training set.*

The average face captures essential (general) face information. Also, since all the values for the faces are nonnegative and the facial images are well removed from the origin, the average face can be treated as a uniform bias of all the faces. Thus we subtract it from the images, as explained in section 3. The average and the deviations, also referred as the caricatures, are

$$(17) \quad \mathbf{a} = \frac{1}{n} \sum_{j=1}^n \mathbf{f}_j,$$

$$(18) \quad \mathbf{x}_j = \mathbf{f}_j - \mathbf{a}.$$

We illustrate the procedure using the University of Surrey database [34]. This consists of a large number of RGB color images taken against a blue background. Using color separation it is easy to remove the background, and the images were then converted to grayscale by averaging the RGB values. A training set consisting of 600 images (3 images of each of 200 individuals) was constructed according to the Lausanne Protocol Configuration 1 [17]. We should point out that the normalization was done by hand and is not particularly accurate, as will become evident in the experiments. Figure 3 shows three different images of two different persons—the first two images are of the same person, where the first image is part of the training set. The second image is not inside the training set, taken on a different day, and one should note the different pose as well as facial expression. The third image is of a person not in the training set.

We need a basis for the space spanned by the \mathbf{x}_j . These basis vectors will become the building blocks for reconstructing any face in the future, whether or not the

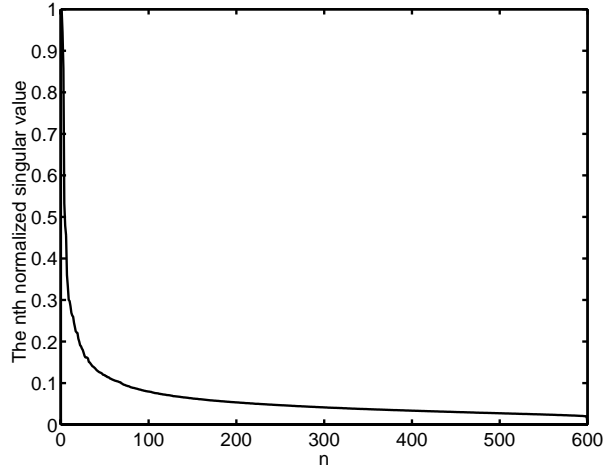


Fig. 4 Graph of the normalized singular values.

face is in training set. In order to construct an orthonormal basis for the subspace spanned by the faces in the training set, we define the $m \times n$ matrix X with columns corresponding to the faces in the training set,

$$(19) \quad X = \frac{1}{\sqrt{n}} \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \dots & \mathbf{x}_n \end{bmatrix},$$

where the constant $\frac{1}{\sqrt{n}}$ is introduced for convenience as in section 3.

From our discussion in section 2, a basis for the column space of X consists of the columns of U associated with the nonzero singular values.

In Figure 4, we plot the singular values normalized by σ_1 ; i.e., we plot σ_n/σ_1 . Although our database is not well normalized, the decay is reasonably fast, and one would expect (using (7)) a reasonable reconstruction in the 2-norm using about 150 singular values, i.e., $\nu = 150$. Equivalently, one concludes, rather arbitrarily in this case, that the effective dimension of the column space of X is about 150.

Recall that the columns of U are the eigenvectors of XX^T . Since

$$(20) \quad XX^T = \frac{1}{n} \sum_{j=1}^n \mathbf{x}_j \mathbf{x}_j^T =: C,$$

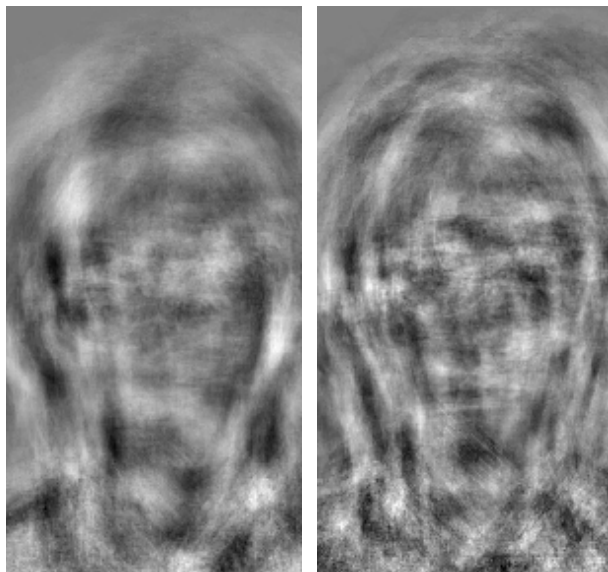
C is the (biased) sample covariance matrix from section 3. Thus, the first ν eigenvectors of the covariance matrix form the sought after basis. These basis vectors, \mathbf{u}_j , $j = 1, \dots, \nu$, are the eigenfaces. Some of the eigenfaces are shown in Figure 5. Note how the higher eigenfaces have more detail added to them. This is in general the case—the lowest eigenfaces contain general face-like information; the detail required to distinguish an individual provided is by the higher-order eigenfaces. For a more detailed explanation, see [21]. In this case one should point out that the almost complete lack of structure of the first eigenface is an indication that our training set is not well normalized.

If our training set were representative, all facial images would be in a 150-dimensional linear subspace. However, the faces in our training set were selected



(a) First eigenface.

(b) Tenth eigenface.



(c) Fiftieth eigenface.

(d) Hundredth eigenface.

Fig. 5 *Some of the eigenfaces.*

arbitrarily and not optimized to be as widely representative as possible. More importantly, our normalization is crude, and we'll describe a simple experiment indicating just how sensitive the system is to the normalization. One should also realize that for an automated facial recognition system, visually perfect reconstruction is not required. Reconstructions that include sufficient information to distinguish between different individuals suffice. Estimates in the literature range from 40 to about 100 based on experiments with a heterogeneous population (see [3]), indicating that faces are described very efficiently in terms of eigenfaces—the effective dimension of the linear vector space containing the facial images is of order 100 (rather than $m = pq$) for a general $p \times q$ image.

One might ask how important it is to subtract the average \mathbf{a} . The gain is probably clear for the example in section 3, where the vectors are all 2D. The advantage for high-dimensional facial images might not be that clear. Imagine that the facial images are clustered around the average face, far from the origin. The first eigenface indeed points in the direction of the average face. The problem is that the rest of the eigenfaces are constrained to be orthogonal to this rather arbitrary direction, reducing their ability to describe the distribution of the faces clustered around the average face. Experiments show that if the average is not removed, the first eigenface is indeed closely related to the average, but the singular values decrease slightly slower.

One striking feature of the calculation of the SVD is the relative sizes of m and n . Because $m = pq$ it can easily become very large, the number n of images in the training set is typically one or two orders of magnitude smaller. The question arises whether one can exploit this fact for numerical purposes. This turns out to be a little tricky. The main problem is that one has to be very careful to ensure that numerical errors do not destroy the orthogonality of the orthogonal matrices. For example one might be tempted to follow the procedure prescribed by Fukunaga [11, p. 39] and first calculate the reduced V_+ from the $n \times n$ symmetric eigenvalue problem (3), using any of the efficient methods available for calculating the eigensystem of symmetric matrices, including the QR algorithm, Rayleigh quotient iteration, divide-and-conquer methods, etc. See [5, 12] for more details. Although there is already a loss in accuracy by computing $X^T X$, it is the next step where one can go seriously wrong. For example, it is tempting to calculate U_+ from

$$(21) \quad XV_+ = U_+\Sigma_+,$$

i.e. $U_+ = XV_+\Sigma_+^{-1}$. This is not a good idea. Note, for example, that roundoff error destroys the orthogonality of the columns of U_+ .

In order to exploit the fact the $n \ll m$ in a numerically stable way, we suggest the following procedure. Given an $m \times n$ matrix X with $n \ll m$, we use another of the great matrix factorizations, namely, the QR factorization with column permutations (the basis of the QR algorithm mentioned above), to calculate $XP = QR$, where Q is an $m \times m$ matrix with orthonormal columns and R is an $m \times n$ upper triangular matrix. P is a permutation matrix rearranging the columns of X such that the diagonal entries of R are in nonincreasing order of magnitude. Since the last $m - n$ rows of R consist of zeros, we can form the reduced QR factorization by keeping the first n columns of Q and the first n rows of R . Thus we obtain $X = Q_+R_+$, where R_+ is an $n \times n$ upper triangular matrix. The next step is to calculate the SVD of $R_+ = U_R\Sigma V^T$. Thus we get $XP = (Q_+U_R)\Sigma V^T$ or $XP = U\Sigma V^T$, with U the product of two orthonormal matrices, $U = Q_+U_R$.

5. Using Eigenfaces. In the previous section we discussed the reasons why faces can be efficiently represented in terms of eigenfaces. To obtain the actual representation is quite straightforward. The idea is to project any given face orthogonally onto the space spanned by the eigenfaces. More precisely, given a face \mathbf{f} we need to project $\mathbf{f} - \mathbf{a}$ onto the column space of $U_\nu := [\mathbf{u}_1 \cdots \mathbf{u}_\nu]$. This means that we wish to solve

$$(22) \quad U_\nu \mathbf{y} = \mathbf{f} - \mathbf{a}$$

in a least squares sense. Since the columns of U_ν are orthonormal, it follows that the eigenface *representation* is given by

$$(23) \quad \mathbf{y} = U_\nu^T (\mathbf{f} - \mathbf{a}).$$

This representation captures all the features associated with the face \mathbf{f} , and instead of comparing faces directly, we rather compare the features \mathbf{y} .

The eigenface *reconstruction* of \mathbf{f} is given by

$$(24) \quad \tilde{\mathbf{f}} = U_\nu \mathbf{y} + \mathbf{a}.$$

For a facial recognition system a good reconstruction is not required. It is only necessary that the features distinguish between different individuals. If, on the other hand, our interest is in the *compression* of facial images, a good visual reconstruction becomes important. Again the idea is to store only the representation \mathbf{y} and then reconstruct the face from the eigenface database when necessary. Anyway, it is impossible not to show the reader the reconstructions visually! The eigenface reconstruction of the first face in Figure 3 (the face in the training set) is shown in Figure 6, using 40, 100, and 450 eigenfaces. The root mean square (rms) errors of the three reconstructions—the 2-norm of the differences between the original and the reconstructions, or, equivalently, the scaled (by σ_1) magnitudes of the first neglected singular values—are given by 0.1306, 0.0793, and 0.0302. Certainly up to 100 eigenfaces, the reconstruction is visually not particularly good, despite the fact that the singular values (rms values) are already relatively small. The 2-norm is clearly not a good norm in which to measure visual correspondence.

We do the same for the image of the first person not in the training set (the second image of Figure 3), and the results are shown in Figure 7. Although the result is not visually pleasing, the important question is whether enough information is available to recognize the individual.

In Figure 8 we show the reconstruction of the third face of Figure 3, a face not in the training set, again using 40, 100, and 450 eigenfaces. Although the reconstruction is visually not particularly good, the individual is already recognizable using 100 eigenfaces. In fact this reconstruction is better than that of Figure 7. The reason is that most of the images in the training set are facing the camera directly. Thus, although not in the training set, the individual with a similar pose is better reconstructed in Figure 8 than the individual in the training set but with a pose not represented in the training set.

The final experiment demonstrates the sensitivity of the system to the normalization. In this experiment we shifted the first image of Figure 3 two pixels down from its original position. Figure 9 shows its reconstruction using 450 eigenfaces. The result is clearly a significant deterioration of the visually perfect reconstruction we obtained in Figure 6.

Up to this point the grayscale images have been 2D representations of the interaction of light with a 3D structure. This poses several difficulties. For example, since

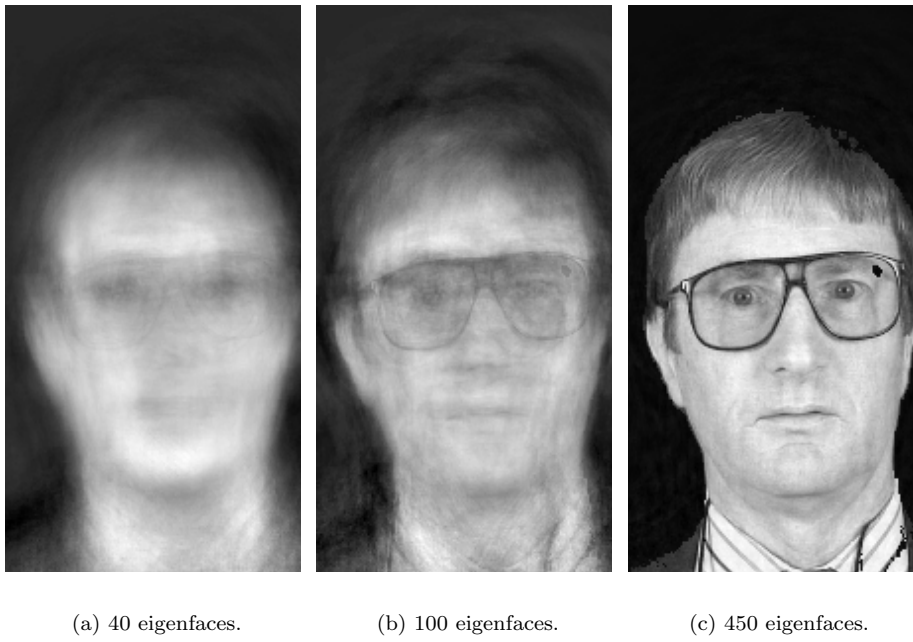


Fig. 6 Various reconstructions of the first face in Figure 3.



Fig. 7 Various reconstructions of the second face in Figure 3.

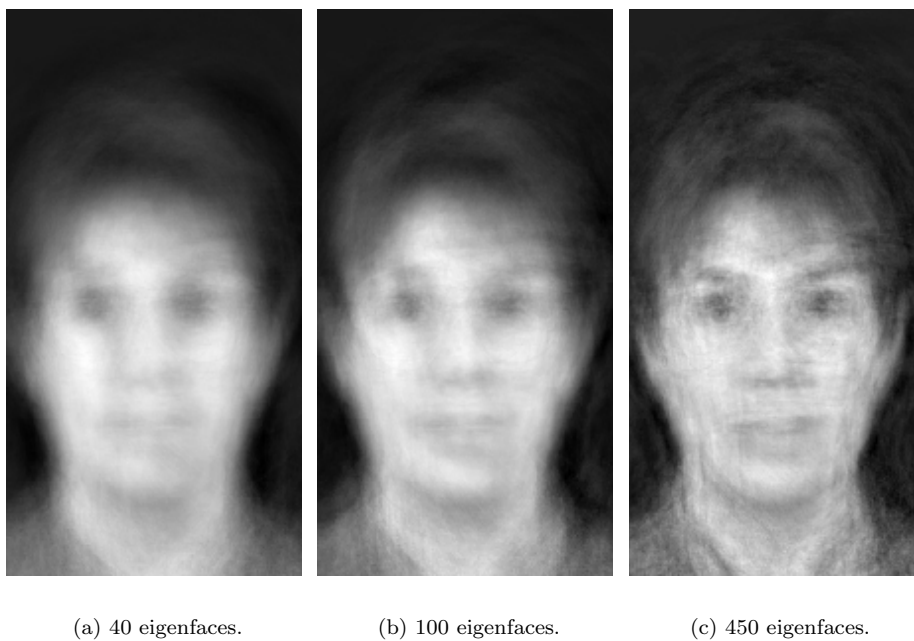


Fig. 8 *Various reconstructions of the third face in Figure 3.*



Fig. 9 *Reconstruction of shifted image.*

no 3D information is available, it is hard to correct for essentially 3D distortions such as out-of-plane rotations. We had more problems reconstructing the image with an out-of-plane rotation of a person inside the training set than that of a person not in the training set but directly facing the camera. Of course facial expression also plays an important role, and is not easily corrected for. Grayscale images also depend heavily on the strength and direction of the light source, and it is not easy to compensate for different lighting conditions [14]. Working with grayscale images where the different shades of gray directly represent 3D structure removes these problems. As pointed out in the introduction, the problem becomes one of reconstructing the 3D image. In the following sections we explain how SVD can again be used, provided a simple camera model is used.

6. Orthographic Camera Model. It is well-known that 3D objects can be reconstructed from stereo pairs of 2D images. For example, the human 3D vision system relies heavily on the presence of *two* eyes. 3D information is also present in a *sequence* of images, such as a video film or, indeed, human vision with one eye closed. Imagine that a person moves about and that we capture views of the face from different angles. These views certainly contain 3D information. The problem is to reconstruct a 3D face from these images.

In general the idea is to identify a number of features on the object that can be tracked over the different video frames. These features should also be selected in such a way that the object can be reconstructed from them. There are different ways of selecting and tracking features; a particularly useful one is due to Lucas and Kanade [16, 32].

A face is a complicated 3D object requiring a large number of feature points for its reconstruction. In addition, it also contains large areas such as the forehead where features are difficult to identify. Although not an impossible task (see, for example, [9]), it is much easier to illustrate the main ideas through simple geometric objects.

The problem we have to solve is an inverse one—from the projections on the video film we need to reconstruct the original object. For this we need to know how the projection was created in the first place; i.e., we need to have a model of the camera. A reasonable model is the so-called pinhole camera with its perspective projection; see, for example, [15]. Although it is possible to do a reconstruction using this camera model, the problem becomes nonlinear, requiring advanced techniques such as the Kalman filter or, more precisely, one of its nonlinear variants; see [1]. Fortunately Kanade and coworkers [28, 29, 30, 4, 20] developed a useful simplification, requiring only the techniques discussed in this paper.

Imagine objects far from the camera or a focal length that approaches infinity. In that case, to a good approximation, the object is projected onto the film parallel to the Z - or optical axis, as shown in Figure 10, also known as an orthographic projection. We'll see in a moment that this simple camera model allows one to solve the problem using linear methods, in essence, by using a matrix factorization (see [30]).

Assume that we observe only one object and, importantly, that the object is a rigid body. The importance of the rigid body assumption lies in the fact that the only movements that rigid bodies can perform are translations and rotations. Translations are easily described by means of a 3D translation vector. Rotations, on the other hand, can be described by *rotation* matrices. Noting that an arbitrary rotation of an object in 3D has three degrees of freedom—it rotates through an *angle* of rotation (one degree of freedom) around its normalized *axis* of rotation (another two degrees of freedom)—one looks for 3×3 matrices with three degrees of freedom. Since 3×3

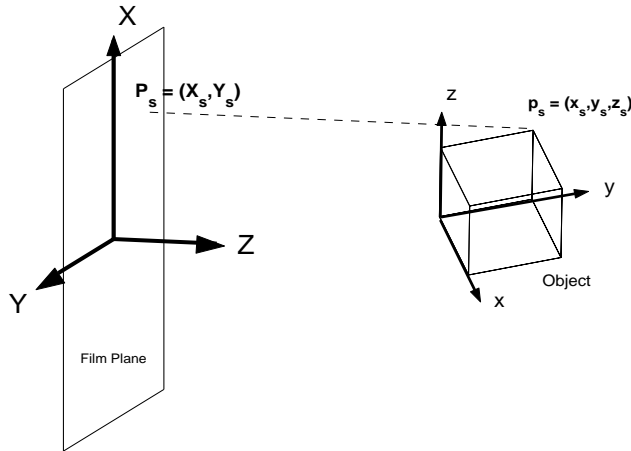


Fig. 10 Orthographic projection.

real, orthonormal matrices, satisfying $RR^T = I = R^T R$, have exactly three degrees of freedom, they are perfect candidates. Since $\det(R) = \pm 1$ in general, there is some ambiguity. A rotation matrix is one with $\det(R) = 1$ —the negative sign involves a reflection as well as a rotation. In general one has to be able to construct the rotation matrix given an axis and angle of rotation or, given a rotation matrix, find the axis and angle of rotation. There is a beautiful result due to Rodrigues (see, for example, [10]) that enables us to do this quite easily.

Proceeding with our description of rigid objects, we choose a reference frame (x, y, z) fixed to the object and describe the object in terms of n feature points,

$$(25) \quad \mathbf{p}_s = \begin{bmatrix} x_s \\ y_s \\ z_s \end{bmatrix}, \quad s = 1, \dots, n.$$

The *shape* matrix, describing the shape of the object in the object coordinate system, is therefore written as

$$(26) \quad S = [\mathbf{p}_1 \quad \mathbf{p}_2 \quad \dots \quad \mathbf{p}_n].$$

The feature points are really all we know about the object, and for a full reconstruction some kind of interpolation or subdivision is required. In the case of a wire-frame cube as in Figure 10, the eight vertexes are natural features, allowing a perfect reconstruction.

We now allow the object to rotate; in section 8 translation is handled similarly, using so-called homogeneous coordinates. The feature points of a rigid body do not change relative to a coordinate system fixed to the object. Now, imagine that the object rotates with respect to another coordinate system fixed to the camera. Each feature point is projected onto the film plane of the camera, and as the object rotates, the features points are projected onto slightly different positions in consecutive frames of the video sequence. The information about the 3D structure of the object is contained in the offsets of the features points in the different frames.

Consider one of the feature points, \mathbf{p}_s , of the object, represented in the *object coordinates* fixed to the object as given by (25). If the rotation of the object coordinate system with respect to that of the camera at the time of the t th video frame is given by the rotation matrix,

$$(27) \quad R_t = \begin{bmatrix} \mathbf{i}_t^T \\ \mathbf{j}_t^T \\ \mathbf{k}_t^T \end{bmatrix},$$

where

$$\begin{bmatrix} \mathbf{i}_t^T \\ \mathbf{j}_t^T \\ \mathbf{k}_t^T \end{bmatrix} = \begin{bmatrix} i_{xt} & i_{yt} & i_{zt} \\ j_{xt} & j_{yt} & j_{zt} \\ k_{xt} & k_{yt} & k_{zt} \end{bmatrix},$$

then the feature point \mathbf{p}_s is given in the camera coordinate system by \mathbf{P}_{ts}^c , where

$$(28) \quad \mathbf{P}_{ts}^c = R_t \mathbf{p}_s.$$

A little later we'll exploit the fact that rotation matrices are orthonormal, i.e., $R_t^T R_t = I$. This means that the rows (and columns) are orthogonal and normalized. In particular, we'll make use of

$$(29) \quad \begin{aligned} \mathbf{i}_t^T \mathbf{i}_t &= 1 = \mathbf{j}_t^T \mathbf{j}_t = \mathbf{k}_t^T \mathbf{k}_t, \\ \mathbf{i}_t^T \mathbf{j}_t &= 0 = \mathbf{i}_t^T \mathbf{k}_t = \mathbf{j}_t^T \mathbf{k}_t. \end{aligned}$$

This gives six relationships between the nine elements of the rotation matrix R_t , leaving three degrees of freedom. These three degrees of freedom are exactly what is needed to describe the angle and normalized axes of rotation.

Since our simple camera projects objects onto the film by translates parallel to the Z -axis of the camera coordinate system, a point $\mathbf{P}_{ts}^c = [X_{ts}, Y_{ts}, Z_{ts}]^T$ in camera coordinates projects onto $\mathbf{P}_{ts} = [X_{ts}, Y_{ts}]^T$ on the film. This can be written as

$$\mathbf{P}_{ts} = O_Z \mathbf{P}_{ts}^c = O_Z R_t \mathbf{p}_s,$$

where O_Z is the orthographic projection matrix,

$$(30) \quad O_Z = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}.$$

If we do this for all n features, we measure the following features in frame t ,

$$(31) \quad W_t = \begin{bmatrix} X_{t1} & X_{t2} & \cdots & X_{tn} \\ Y_{t1} & Y_{t2} & \cdots & Y_{tn} \end{bmatrix} = O_Z R_t S,$$

with S given by (26). If we write

$$(32) \quad M_t = O_Z R_t = \begin{bmatrix} i_{xt} & i_{yt} & i_{zt} \\ j_{xt} & j_{yt} & j_{zt} \end{bmatrix},$$

equation (31) becomes

$$(33) \quad W_t = M_t S.$$

All that remains to be done is to collect all the observations over f frames into a single observation matrix W ,

$$(34) \quad W = \begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_f \end{bmatrix} = \begin{bmatrix} X_{11} & X_{12} & \cdots & X_{1n} \\ Y_{11} & Y_{12} & \cdots & Y_{1n} \\ \vdots & & & \vdots \\ X_{f1} & X_{f2} & \cdots & X_{fn} \\ Y_{f1} & Y_{f2} & \cdots & Y_{fn} \end{bmatrix}.$$

From the expression for a single frame (33) we obtain

$$(35) \quad W = MS,$$

where M is the *motion* matrix assembled from (32),

$$(36) \quad M = \begin{bmatrix} M_1 \\ \vdots \\ M_f \end{bmatrix} = \begin{bmatrix} i_{x1} & i_{y1} & i_{z1} \\ j_{x1} & j_{y1} & j_{z1} \\ i_{x2} & i_{y2} & i_{z2} \\ j_{x2} & j_{y2} & j_{z2} \\ \vdots & & \\ i_{xf} & i_{yf} & i_{zf} \\ j_{xf} & j_{yf} & j_{zf} \end{bmatrix}.$$

Thus, given the observation matrix W , the mathematical problem is to find M and S ; i.e., the problem is reduced to factorizing the observation matrix into motion and shape matrices.

Let us take another look at the structures of the motion and shape matrices, M and S . Note that their maximum rank is 3. This means that the maximum rank of the observation matrix W should also be 3. In practice, however, W is obtained by tracking features over a number of frames, introducing all kinds of tracking and measurement errors, with the result that the actual rank of the $2f \times n$ matrix W will be higher than 3. We should therefore try to extract the best possible rank-3 approximation of W . We have seen that this is where the SVD is particularly effective.

It is possible for S to have rank 1, 2, or 3, where the latter indicates a full 3D object. Rank 2 corresponds to a planar object, rank 1 to a line. For instance, for a planar object we can always choose the object coordinate systems such that the plane of the object is given by $z = 0$. The motion matrix M can have rank 2 or 3. Rank 3 indicates a full 3D rotation, and rank 2 describes a rotation around the Z -axis. Since a rotation around the Z -axis presents only one side of the object to the film, the 3D object is perceived as a planar object—all this information is encoded in the rank of the observation matrix W . To reiterate, due primarily to measurement errors, rank may not be exact, and one needs a strategy to determine the effective rank of W . The key is SVD.

7. Reconstructing 3D Images. In the previous section we showed that we need to factorize the observation matrix W into a $2f \times 3$ motion matrix M and a $3 \times n$ shape matrix S . For this purpose we calculate the full SVD of the observation matrix,

$$(37) \quad W = U\Sigma V^T.$$

As pointed out at the end of the previous section, the rank of W is in general higher than 3. Thus we choose the 3 largest singular values and form the rank-3 approximation of W ,

$$(38) \quad \widetilde{W} = U_+ \Sigma_+ V_+^T,$$

where

$$(39) \quad \Sigma_+ = \text{diag}(\sigma_1, \sigma_2, \sigma_3)$$

and U_+ and V_+ consist of the first 3 columns of U and V , respectively. The expression (38) for \widetilde{W} is therefore the best rank-3 approximation of W in the sense of (7).

Thus, we can write

$$(40) \quad \widetilde{W} = \widetilde{M} \widetilde{S},$$

where

$$(41) \quad \widetilde{M} = U_+ \Sigma_+^{\frac{1}{2}} \quad \text{and} \quad \widetilde{S} = \Sigma_+^{\frac{1}{2}} V_+^T.$$

Although \widetilde{W} is written in the correct form, \widetilde{M} and \widetilde{S} are not yet the desired motion and shape matrices. In fact, the factorization is not unique. Indeed, let A be any invertible 3×3 matrix; then

$$(42) \quad \widetilde{W} = (\widetilde{M}A) (A^{-1}\widetilde{S})$$

is another factorization. We now exploit the freedom provided by A to get the motion matrix M in the correct form; i.e., we choose A in such a way that

$$(43) \quad M = \widetilde{M}A$$

has all the properties of the motion matrix. Now it is time to recall that the motion matrix M consists of orthogonal rows in the sense of (29). Setting $Q = AA^T$, the orthonormality relations (29) are written with the help of (43), (32), and (43) as

$$(44) \quad \begin{aligned} \widetilde{m}_{2t-1}^T Q \widetilde{m}_{2t-1} &= 1 = \widetilde{m}_{2t}^T Q \widetilde{m}_{2t}, \\ \widetilde{m}_{2t-1}^T Q \widetilde{m}_{2t} &= 0, \quad t = 1, \dots, f, \end{aligned}$$

where \widetilde{m}_k^T is the k th row of \widetilde{M} . Since Q is a symmetric 3×3 matrix it has 6 unknown entries that can be determined in a least squares sense from the $3f$ equations (44).

Once Q is calculated, another factorization is required to obtain A . This problem is also undetermined. Indeed, A has 9 unknown entries, and we only know the 6 elements of the symmetric matrix Q . So, 3 elements of A remain undetermined. The reason for this is that we are free to choose the object's coordinate system—since we do not yet allow translation, we are free to choose its rotational orientation. Thus, through factorization of Q we find A only up to an arbitrary rotation. More precisely, if $Q = \widetilde{A}\widetilde{A}^T$, then $Q = (\widetilde{A}R)(\widetilde{A}R)^T$ is another factorization for any rotation matrix R . Any 3×3 rotation matrix has precisely 3 arbitrary elements; these are the 3 undetermined elements of A . We therefore have a choice, and one possibility is to require the object to be oriented as in the first frame.

Since Q is symmetric it can be diagonalized with an orthonormal matrix U_Q (see, e.g., [12]),

$$Q = U_Q \Lambda_Q U_Q^T,$$

where one again notes that the eigenvalues of $Q = AA^T$ are all nonnegative. If $\tilde{A} = U_Q \Lambda_Q^{\frac{1}{2}}$, then $A = \tilde{A}R^T$ for any 3×3 rotation matrix R (the transpose of R is used for convenience). Choosing A such that the object is oriented as in the first frame, R follows from

$$(45) \quad \mathbf{w}_1 = O_Z R \tilde{A}^{-1} \tilde{S},$$

where \mathbf{w}_1 denotes the first two rows of the observation matrix (34). Looking more carefully at (45) we note that it consists of six linear equations (the left-hand side is a 2×3 matrix) in six unknowns ($O_Z R$ selects the first two rows of the 3×3 matrix R). Since \tilde{A} and \tilde{S} are known, we can solve the resulting 6×6 linear system. For a rotation matrix this then determines the last row uniquely.

The final step is to calculate the motion and shape matrices from

$$(46) \quad M = \tilde{M} \tilde{A} R^T \quad \text{and} \quad S = R \tilde{A}^{-1} \Sigma_+^{\frac{1}{2}} V_+^T.$$

We now summarize the algorithm, assuming that the rigid object only rotates (translation is described in the next section):

1. Select n features on the object and track their images on the film over f frames.
2. Assemble the x and y coordinates of the features on the f frames into a single $2f \times n$ observation matrix W .
3. Compute the SVD of $W = U \Sigma V^T$.
4. Determine the effective rank of W by inspecting the singular values. For nondegenerate motion $\text{rank}(W) = 3$, in which case form the best rank-3 approximation of W , namely, $\tilde{W} = U_+ \Sigma_+ V_+^T$.
5. Construct $\tilde{M} = U_+ \Sigma_+^{\frac{1}{2}}$.
6. Calculate the symmetric matrix Q as the least squares solution of (44).
7. Diagonalize Q , i.e., $Q = U_Q \Lambda_Q U_Q^T$.
8. Set $\tilde{A} = U_Q \Lambda_Q^{\frac{1}{2}}$.
9. Calculate R by solving the linear system of equations implied by (45).
10. Compute $M = \tilde{M} \tilde{A} R^T$ and $S = R \tilde{A}^{-1} \Sigma_+^{\frac{1}{2}} V_+^T$.

8. Rotation and Translation. For simplicity we have assumed up to now that we are dealing only with pure rotations. Now we incorporate translations into the model. The ideas as well as much of the formalism are the same as for pure rotations if we use the so-called homogeneous coordinates. If during the t th frame a translation \mathbf{c}_t occurs, the observation is

$$(47) \quad \mathbf{w}_t = O_Z (R_t S + \mathbf{c}_t).$$

In homogeneous coordinates (see [15] for an excellent account of the applications of projective geometry to computer vision) each feature point is written as

$$(48) \quad \mathbf{p}_s = \begin{bmatrix} x_s \\ y_s \\ z_s \\ 1 \end{bmatrix},$$

resulting in a homogeneous shape matrix of the form,

$$(49) \quad S = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \\ y_1 & y_2 & \cdots & y_n \\ z_1 & z_2 & \cdots & z_n \\ 1 & 1 & \cdots & 1 \end{bmatrix}.$$

The advantage of the homogeneous coordinates is that the translation can now be incorporated into the motion matrix as

$$(50) \quad M = \begin{bmatrix} i_{x1} & i_{y1} & i_{z1} & c_{x1} \\ j_{x1} & j_{y1} & j_{z1} & c_{y1} \\ \vdots & \vdots & \vdots & \vdots \\ i_{xf} & i_{yf} & i_{zf} & c_{xf} \\ j_{xf} & j_{yf} & j_{zf} & c_{yf} \end{bmatrix} =: [M_r \quad \mathbf{c}],$$

with the result that the observation matrix is again written as a product $W = MS$. We now proceed along the same lines as with pure rotation, and again the SVD of W is calculated. In this case the maximum rank of W in the nondegenerate case is 4. Accordingly we extract the best rank-4 approximation of W and factorize into \widetilde{M} and \widetilde{S} . The calculation of A requires a bit more care. For $\text{rank}(W) = 4$, A is 4×4 and can be partitioned as

$$(51) \quad A = [A_r \quad \mathbf{a}_c],$$

where A_r is a 4×3 matrix related to rotation and \mathbf{a}_c is a vector related to translation. More specifically, $\widetilde{M}A_r = M_r$, where M_r is the rotational part of the motion matrix in (50). Thus A_r is again obtained from the constraints (44), but now with $Q = A_r A_r^T$.

In order to determine \mathbf{a}_c we note that we are free to choose the translational orientation of the object's coordinate system. Since the centroid of the object maps to the centroid of the observations under an orthographic projection, the centroid of the observations is

$$(52) \quad \bar{\mathbf{w}} = \begin{bmatrix} \frac{1}{n} \sum_{s=1}^n X_{s1} \\ \frac{1}{n} \sum_{s=1}^n Y_{s1} \\ \vdots \\ \frac{1}{n} \sum_{s=1}^n X_{sf} \\ \frac{1}{n} \sum_{s=1}^n Y_{sf} \end{bmatrix}.$$

If

$$(53) \quad \bar{\mathbf{p}} = \frac{1}{n} \sum_{s=1}^n \mathbf{p}_s$$

is the centroid of the object, then

$$(54) \quad \bar{\mathbf{w}} = \widetilde{M} [A_r \quad \mathbf{a}_c] \begin{bmatrix} \bar{\mathbf{p}} \\ 1 \end{bmatrix}.$$

The simplest choice for the origin of the object coordinate system is $\bar{\mathbf{p}} = 0$. Consequently,

$$(55) \quad \bar{\mathbf{w}} = \widetilde{M} \mathbf{a}_c.$$

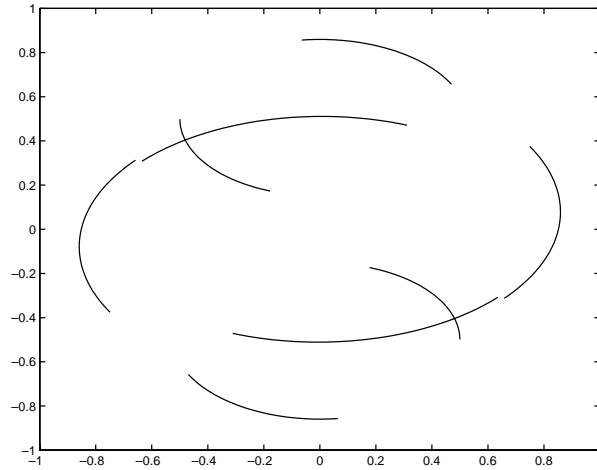


Fig. 11 *Observations of a rotated cube.*

This overconstrained problem for \mathbf{a}_c is again solved with the least squares method.

The algorithm can be extended to handle several objects moving simultaneously, but independently. Tracking the features is considerably harder and requires one to determine to which object each feature belongs. This is a nontrivial task for which more details are given in [4].

9. Example. For our first example, we generate a cube and then rotate it mathematically. The 8 vertexes of the cube form natural features, and during the rotation their images are recorded on an imaginary film under an orthographic projection. Now imagine that the shutter of the camera is kept open while recording the projections of the feature points so that each one of the 8 vertexes traces a specific trajectory on the film plane, as shown in Figure 11. This is the only information available about the object, and the movement of the vertexes illustrated in Figure 11 are then assembled in the observation matrix W . For this example W turned out to be 100×8 , implying that the 8 vertexes are tracked through 50 frames. It is from this measurement matrix that the motion and shape is reconstructed. It is not possible to illustrate the construction of the motion matrix, but the shape matrix, hence the object, is shown in Figure 12; see also [6]. It may be worth pointing out that only the features are reconstructed and that the lines connecting the vertexes in Figure 12 can be drawn only because we have advanced knowledge of the object—it is of course possible to connect the vertexes in a different manner where it might not be so obvious that we are dealing with a cube.

For our second example we use 3D data obtained from scanning the face of a mannequin. The facial data was then rotated mathematically and the feature points orthographically projected onto the film. In our first experiment the feature points were chosen by hand, shown as red dots in Figure 13(a). The motion of the feature points on the film is shown in Figure 14. Note that the 3D information is encoded in the different path lengths traced by the different feature points on the film. The fact that the features follow straight lines is the result of a pure rotation around the vertical axis. The reconstruction is shown in Figure 13(b), which was obtained by a cubic interpolation between the selected features. Since a small number of feature

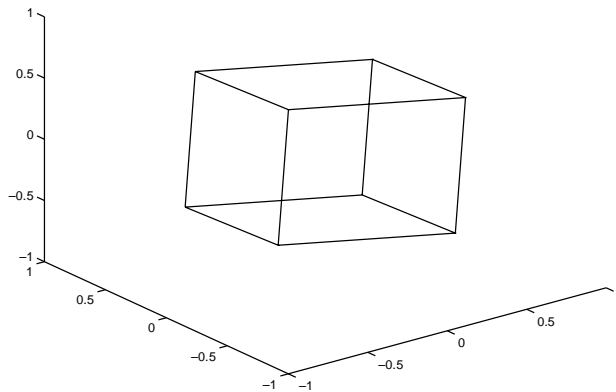


Fig. 12 *Reconstructed cube.*



(a) The original image with the selected feature points shown in red.

(b) The reconstructed face.

Fig. 13 *The original and reconstructed faces.*

points were used, the reconstruction is not particularly good.

The result using all the grid values as features (see Figure 15(a)) is shown in Figure 15(b). Since we track all the features mathematically as they move across the image plane, the reconstruction is perfect.

In order to convert these ideas to a practical facial identification system, a sufficient number of features should be identified on the face for a good reconstruction of the face. Tracking these features in a video sequence containing different views of the face would then allow one to do a 3D reconstruction as described above. The 3D image can then be displayed as a grayscale image, at which stage the eigenface procedure can be applied. As pointed out repeatedly, the main practical difficulties lie with the image processing—identifying and tracking a sufficient number of features to allow a facial reconstruction.

10. Conclusions. The examples described in this paper are testimony to the power of SVD—it forms the mathematical basis of a powerful facial recognition sys-

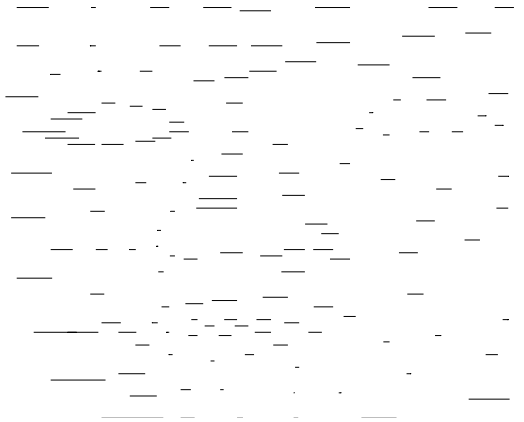
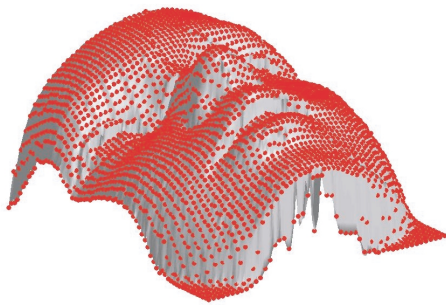
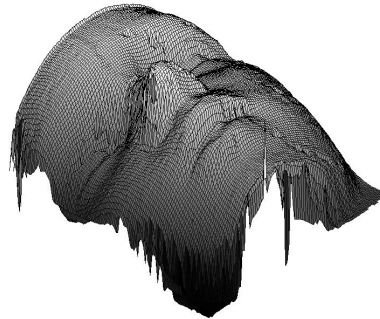


Fig. 14 *The motion of the feature points across the film plane.*



(a) The original image with the selected feature points shown in red.



(b) The reconstructed face.

Fig. 15 *The original and reconstructed faces.*

tem as well as 3D reconstructions from orthographic projections. The real difficulties lie in the image processing. For facial recognition the images need to be well normalized with respect to lighting, scale, rotation, etc. Normalization is not always easy, especially when one has little control over the situation.

All 3D reconstructions, stereo as well as the reconstruction described in this paper, rely on our ability to accurately identify the same features in the different images. We have carefully avoided this problem in our discussion by manipulating the 3D objects mathematically. This is of course not feasible in practice, and feature tracking remains an area of active research.

Acknowledgments. We are grateful to Elizabeth Jessup, Willy Hereman, and Gilbert Strang and to an anonymous referee for suggesting the algorithm for calculating the SVD in section 4. We would also like to thank all of the referees for many invaluable suggestions and comments. We are grateful to Dirk Wagener, who pro-

duced Figures 13–15, and to Ingrid Engelbrecht, who brought to our attention the contributions of early artists to anatomical studies.

REFERENCES

- [1] A. AZARBAYEJANI AND A. P. PENTLAND, *Recursive estimation of motion, structure, and focal length*, IEEE Trans. Pattern Anal. Machine Intelligence, 17 (1995), pp. 562–575.
- [2] C. BEAVAN, *Fingerprints. The Origins of Crime Detection and the Murder Case That Launched Forensic Science*, Hyperion, New York, 2001.
- [3] J. BRINK, C. NIEUWOUTD, AND E. BOTHA, *Facial image compression using the Karhunen-Loève transform*, in Proceedings of the Ninth Annual South African Workshop on Pattern Recognition, 1998.
- [4] J. P. COSTEIRA, *A Multibody Factorization Method for Motion Analysis*, Ph.D. thesis, Technical University of Lisbon, 1995.
- [5] J. W. DEMMEL, *Applied Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
- [6] *Department of Applied Mathematics*, University of Stellenbosch, ftp site, ftp://dip.sun.ac.za/.
- [7] *Department of Mathematics*, MIT, Cambridge, MA, ftp site, http://web.mit.edu/18.06/www/.
- [8] J. DIAMOND, *The Third Chimpanzee*, Harper Perennial, New York, 1992.
- [9] A. ERIKSSSEN, *3-d Face Recognition*, Master's thesis, University of Stellenbosch, South Africa, 1999.
- [10] O. FAUGERAS, *Three-Dimensional Computer Vision*, MIT Press, Cambridge, MA, 1993.
- [11] K. FUKUNAGA, *Statistical Pattern Recognition*, 2nd ed., Johns Hopkins University Press, Baltimore, MD, 1989.
- [12] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computation*, 3rd ed., Academic Press, Boston, 1996.
- [13] K. HEGER, *The Illustrated History of Surgery*, Harold Starke, London, 1989.
- [14] P. W. HALLINAN, G. G. GORDON, A. L. YUILLE, P. GIBLIN, AND D. MUMFORD, *Two- and Three-Dimensional Face Patterns of the Face*, A. K. Peters, Natick, MA, 1999.
- [15] R. HARTLEY AND A. ZISSERMAN, *Multiple View Geometry in Computer Vision*, Cambridge University Press, Cambridge, UK, 2000.
- [16] B. LUCAS AND T. KANADE, *An iterative image registration technique with an application to stereo vision*, in Proceedings of the 7th International Joint Conference on Artificial Intelligence, Vancouver, 1981, pp. 674–679.
- [17] J. LUETTIN AND G. MAÏTRE, *Evaluation protocol for the extended M2VTS database (XM2VTSDB)*, IDIAP-COM 05, IDIAP, 1998.
- [18] D. MCNEILL, *The Face*, Penguin Books, London, 1998.
- [19] E. MORGAN, *The Aquatic Ape Hypothesis*, Souvenir Press, London, 1997.
- [20] T. MORITA AND T. KANADE, *A sequential factorization method for recovering shape and motion from image streams*, IEEE Trans. Pattern Anal. Machine Intelligence, 19 (1997), pp. 858–867.
- [21] N. MULLER, *Facial Recognition, Eigenfaces and Synthetic Discriminant Functions*, Ph.D. thesis, University of Stellenbosch, South Africa, 2000.
- [22] A. PENTLAND, B. MOGHADDAM, AND T. STARNER, *View based and modular eigenspaces for face recognition*, in Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1994.
- [23] A. PENTLAND, T. STARNER, N. ETCOFF, A. MASOIU, O. OLIYIDE, AND M. TURK, *Experiments with Eigenfaces*, Technical Note 194, Perceptual Computing Group, MIT Media Laboratory, Cambridge, MA, 1992.
- [24] J. PHILLIPS, H. MOON, P. RAUSS, AND S. RIZVI, *The FERET September 1996 database and evaluation procedure*, in Proceedings of the First International Conference and Video-based Biometric Person Authentication, 1997.
- [25] S. PINKER, *How the Mind Works*, Penguin Books, London, 1998.
- [26] L. SIROVICH AND M. KIRBY, *Low-dimensional procedure for the characterization of human faces*, J. Opt. Soc. Amer. A., 4 (1987), pp. 519–524.
- [27] G. STRANG, *Introduction to Linear Algebra*, 2nd ed., Wellesley-Cambridge Press, Cambridge, MA, 1998.
- [28] C. TOMASI AND T. KANADE, *Shape and Motion from Image Streams: A Factorization Method*, Technical Report CS-90-166, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1991.
- [29] C. TOMASI AND T. KANADE, *Shape and Motion from Image Streams: A Factorization Method – Detection and Tracking of Points Features*, Technical Report CS-91-132, School of Com-

- puter Science, Carnegie Mellon University, Pittsburgh, PA, 1991.
- [30] C. TOMASI AND T. KANADE, *Shape and motion from image streams under orthography*, *Internat. J. Comput. Vision*, 9 (1992), pp. 137–154.
 - [31] L. N. TREFETHEN AND D. BAU III, *Numerical Linear Algebra*, SIAM, Philadelphia, 1997.
 - [32] E. TRUCCO AND A. VERRI, *Introductory techniques for 3D computer vision*, Prentice-Hall, Englewood Cliffs, NJ, 1998.
 - [33] M. TURK AND A. PENTLAND, *Eigenfaces for recognition*, *J. Cognitive Neurosci.*, 3 (1991), pp. 71–86.
 - [34] UNIVERSITY OF SURREY, *The XM2VTS Face Database*, available online from <http://www.ee.surrey.ac.uk/Research/VSSP/xm2vtsdb/>.