Merging and Splitting Eigenspace Models

Peter Hall, David Marshall, and Ralph Martin

Abstract—We present new deterministic methods that given two eigenspace models—each representing a set of n-dimensional observations—will: 1) merge the models to yield a representation of the union of the sets and 2) split one model from another to represent the difference between the sets. As this is done, we accurately keep track of the mean. Here, we give a theoretical derivation of the methods, empirical results relating to the efficiency and accuracy of the techniques, and three general applications, including the construction of Gaussian mixture models that are dynamically updateable.

Index Terms—Eigenspace models, principal component analysis, model merging, model splitting, Gaussian mixture models.

1 Introduction

The contributions of this paper are: 1) a method for merging eigenspace models and 2) a method for splitting eigenspace models, in both of which we explicitly and accurately *keep track of the mean* of the observations. Methods for merging (updating) or splitting (downdating) eigenspace models exist [1], [2], [3], [4], [5], but they fail to handle a change in the mean. The methods we provide do update the mean, which is of crucial importance in classification problems—in such problems the mean represents the center of a cluster of observations in a given class.

An eigenspace model is a statistical description of a set of N observations in n-dimensional space; such a model may be regarded as a multidimensional Gaussian distribution. Geometrically, an eigenspace model is a hyperellipsoid: Its center is the mean of the observations; its axes point in directions along which the spread of observations is maximized, subject to orthogonality. The surface of the hyperellipsoid is a contour that lies at constant standard deviation from the mean. Often, the hyperellipsoid is almost flat along certain directions and, thus, can be modeled as having lower dimension than the space in which it is embedded.

Eigenspace models have a wide variety of applications, for example: classification for recognition systems [6], characterizing normal modes of vibration for dynamic models, such as the heart [7], motion sequence analysis [8], and the temporal tracking of signals [4].

Our motivation for this work arose in the context of building models of blood vessels for x-ray interpretation and building eigenspace models for many images. We would also like to incrementally build Gaussian mixture models, which use separate Gaussian distributions to describe data falling into several clusters or classes. Updating the means is a prerequisite in this case, as the mean represents the center of the distribution for each class; classification is based on the Mahalanobis distance, which measures the distance from the mean in units of standard deviation.

Eigenspace models are computed using either eigenvalue decomposition (EVD) (also called principal component analysis) or singular-value decomposition (SVD). We would like to distinguish between *batch* and *incremental* computation. In batch computation, all observations are used simultaneously to compute the eigenspace model. In an incremental computation, an existing eigenspace model is updated using new observations.

Previous research in incremental computation of eigenspace models has only considered adding *exactly one* new observation at a time to an eigenspace model [1], [2], [3], [4], [5]. None of these methods require the original observations to be retained; a description of the hyperellipsoid is sufficient information for incremental computation. Each of these previous approaches allows for a change in dimensionality of the hyperellipsoid so that a single additional axis is added if necessary. Only our previous work allows for a shift of the center of the hyperellipsoid [9], which proves crucial if the eigenspace model is to be used for classification.

Incremental methods do not need all observations at once—thus, reducing storage requirements and making large problems computationally feasible. Incremental methods *must* be used if not all observations are available simultaneously. These advantages are retained even if *low-dimensional* methods are used to compute the eigenspace [5], [10], which can be computed with a problem size no larger than $\min(n, N)$. We will return to low-dimensional methods later, in Section 2.2. Even if all observations are available, it is usually faster to compute a new eigenspace model by incrementally updating an existing one rather than by using batch computation [3], as our results show (see Section 5).

The disadvantage of incremental methods is their accuracy compared to batch methods. When only a few incremental updates are made, the inaccuracy is small and is probably acceptable for the great majority of applications [9]. When many thousands of updates are made, as when eigenspace models are incremented with a single observation at a time, the inaccuracies build up, although methods exist to circumvent this problem [4]. In contrast, our methods allow a whole new *set* of observations to be added in a single step, thus reducing the total number of updates to an existing model.

Section 2 defines eigenspace models in detail, standard methods for computing them, and how they are used for representing observations. Section 3 discusses merging of eigenspace models, while Section 4 addresses splitting. Section 5 presents empirical results and Section 6 presents some applications of the work. Section 7 gives our conclusions.

2 Eigenspace Models

In this section, we define more precisely what we mean by *eigenspace models*, briefly discuss standard methods for their batch computation, and how observations can be represented using them. First, we establish our notation for the rest of the paper.

Vectors are columns, denoted by a single underline. Matrices are denoted by a double underline. The size of a vector, or matrix, is often important and, where we would like to emphasize this size, it is denoted by subscripts. Particular column vectors within a matrix are denoted by a superscript and a superscript on a vector denotes a particular observation from a set of observations, so we treat observations as column vectors of a matrix. As an example, $\underline{\underline{A}}^{i}_{mn}$ is the *i*th column vector in an $(m \times n)$ matrix. We denote matrices formed by concatenation using square brackets. Thus, $[\underline{\underline{A}}_{mn}\underline{b}]$ is an $(m \times (n+1))$ matrix, with vector $\underline{b}$ appended to $[\underline{\underline{A}}_{mn}$ as a last column.

2.1 Theoretical Background

Consider N observations, each a column vector $\underline{x}^{i} \in \Re^{n}$. We compute an eigenspace model as follows:

The mean of the observations is

• *P. Hall is with the School of Mathematical Science, Universtiy of Bath, Bath BA2 7AY, UK. E-mail: maspmh@maths.bath.ac.uk.*
• *D. Marshall and R. Matrin are with Department of Computer Science, University of Wales, Cardiff, PO Box 916, Cardiff CF2 3XF, Wales UK. E-mail: {dave, ralph}@cs.cf.ac.uk.*

Manuscript received 12 Aug. 1998; accepted 6 Oct. 1999.
Recommended for acceptance by Y.-F. Wang.
For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 107985.

$$\underline{\bar{x}}d = \frac{1}{N}\sum_{i=1}^{N}\underline{x}^i \quad (1)$$

and their covariance is

$$\underline{\underline{C}}_{nn} = \left(\frac{1}{N}\sum_{i=1}^{N}\underline{x}^i(\underline{x}^i)^T\right) - \underline{\bar{x}}\underline{\bar{x}}^T. \quad (2)$$

Note that $\underline{\underline{C}}_{nn}$ is real and symmetric. The axes of the hyperellipsoid and the spread of observations over each axis are the eigenvectors, $\underline{\underline{U}}_{nn}$, and eigenvalues, $\underline{\underline{\Lambda}}_{nn}$ of the eigenproblem

$$\underline{\underline{C}}_{nn} = \underline{\underline{U}}_{nn}\,\underline{\underline{\Lambda}}_{nn}\,\underline{\underline{U}}_{nn}^T \quad (3)$$

$\underline{\underline{\Lambda}}_{nn}$ is a diagonal matrix. The eigenvectors are orthonormal so that $\underline{\underline{U}}_{nn}^T\,\underline{\underline{U}}_{nn} = \underline{\underline{I}}_{nn}$, the $(n \times n)$ identity matrix. The ith eigenvector $\underline{U}^i$ and ith eigenvalue $\underline{\underline{\Lambda}}_{nn}^{ii}$ are associated, the eigenvalue is the length of the eigenvector, which is the ith axis of the hyperellipsoid.

Typically, only $p \le \min(n, N)$ of the eigenvectors have significant eigenvalues and, hence, only p of the n eigenvectors need be retained. This happens when the observations are correlated so that the covariance matrix is, to a good approximation, rank-degenerate: Small eigenvalues are presumed to be negligible. Thus, an eigenspace model often spans a p-dimensional subspace of the n-dimensional space in which it is embedded. Different criteria for discarding eigenvectors and eigenvalues exist and these suit different applications and different methods of computation. Three common methods are: 1) Stipulate p as a fixed integer and so keep the p largest eigenvectors [5], 2) keep those p eigenvectors whose size is larger than an absolute threshold [3], and 3) keep the p eigenvectors such that a specified fraction of energy in the eigenspectrum (computed as the sum of eigenvalues) is retained.

Having chosen to discard certain eigenvectors and eigenvalues, we can recast (3) using block form matrices and vectors. Without loss of generality, we can permute the eigenvectors and eigenvalues such that $\underline{\underline{U}}_{np}$ are those eigenvectors that are kept and $\underline{\underline{\Lambda}}_{pp}$ their eigenvalues and $\underline{\underline{U}}_{nd}$ and $\underline{\underline{\Lambda}}_{dd}$ are those discarded, with $d = n - p$. We may rewrite (3) as:

$$\begin{aligned}\underline{\underline{C}}_{nn} &= [\underline{\underline{U}}_{np}\,\underline{\underline{U}}_{nd}]\begin{bmatrix}\underline{\underline{\Lambda}}_{pp} & \underline{\underline{0}}_{pd}\\ \underline{\underline{0}}_{dp} & \underline{\underline{\Lambda}}_{dd}\end{bmatrix}[\underline{\underline{U}}_{np}\,\underline{\underline{U}}_{nd}]^T\\ &\approx \underline{\underline{U}}_{np}\,\underline{\underline{\Lambda}}_{pp}\,\underline{\underline{U}}_{np}^T\end{aligned} \quad (4)$$

with error $\underline{\underline{U}}_{nd}\,\underline{\underline{\Lambda}}_{dd}\,\underline{\underline{U}}_{nd}^T$, which is small if $\underline{\underline{\Lambda}}_{dd} \approx \underline{\underline{0}}_{dd}$.

Thus, we define an *eigenspace model*, Ω, as the mean, a (reduced) set of eigenvectors, their eigenvalues, and the number of observations: $\Omega = (\underline{\bar{x}}, \underline{\underline{U}}_{np}, \underline{\underline{\Lambda}}_{pp}, N)$

2.2 Low-Dimensional Computation of Eigenspace Models

Low-dimensional batch methods are often used to compute eigenspace models and are especially important when the dimensionality of the observations is very large compared to their number. Thus, they may be used to compute eigenspace models that would otherwise be infeasible. Incremental methods also use a low dimensional approach.

In principle, computing an eigenspace model requires that we construct an $(n \times n)$ matrix, where n is the dimension of each observation. In practice, the model can be computed by using an $(N \times N)$ matrix, where N is the number of observations. This is an advantage in applications like image processing where, typically, $N \ll n$.

This can be done by considering the relationship between eigenvalue decomposition and singular value decomposition. This leads to a simple derivation for a low-dimensional batch method for computing the eigenspace model (for alternative derivation see [5], [10]).

Let $\underline{\underline{Y}}_{nN}$ be the set of observations shifted to the mean so that $\underline{Y}^i = \underline{x}^i - \underline{\bar{x}}$. Then, an SVD of $\underline{\underline{Y}}_{nN}$ is: $\underline{\underline{Y}}_{nN} = \underline{\underline{U}}_{nn}\,\underline{\underline{\Sigma}}_{nN}\,\underline{\underline{V}}_{NN}^T$, where $\underline{\underline{U}}_{nn}$ are the left singular vectors, which are identical to the eigenvectors previously given $\underline{\underline{\Sigma}}_{nN}$ is a matrix with singular values on its leading diagonal, with $\underline{\underline{\Lambda}}_{nn} = \underline{\underline{\Sigma]}}_{nN}\underline{\underline{\Sigma}}_{nN}^T/N$; and $\underline{\underline{V}}_{NN}$ are right singular vectors. Both $\underline{\underline{U}}_{nn}$ and $\underline{\underline{V}}_{NN}$ are orthonormal matrices. This can now be used to compute eigenspace models in a low-dimensional way, as follows: $\underline{\underline{Y}}_{nN}^T\,\underline{\underline{Y}}_{nN} = \underline{\underline{V}}_{NN}\,\underline{\underline{\Sigma}}_{nN}^T\,\underline{\underline{\Sigma}}_{nN}\,\underline{\underline{V}}_{NN}^T = \underline{\underline{V}}_{NN}\,\underline{\underline{S}}_{NN}\,\underline{\underline{V}}_{NN}^T$ is an $(N \times N)$ eigenproblem. $\underline{\underline{S}}_{NN}/N$ is the same as $\underline{\underline{\Lambda}}_{nn}$, except for the presence of extra trailing zeros on the main diagonal of $\underline{\underline{\Lambda}}_{nn}$. If we discard the small singular values, and their singular vectors, following the above, then remaining eigenvectors vectors are $\underline{\underline{U}}_{np} = \underline{\underline{Y}}_{nN}\,\underline{\underline{V}}_{Np}\,\underline{\underline{\Sigma}}_{pp}^{-1}$.

This result formed the basis of the incremental technique developed by Murakami and Kumar [5], but they did not allow for a change in origin. SVD methods for adding a single point, with no change in mean, were actually proposed quite early in the development of incremental eigenproblem analysis [2] and, more recently, Chandrasekaran et al. [3] have observed that a solution based on the matrix product $\underline{\underline{Y}}_{nN}^T\,\underline{\underline{Y}}_{nN}$, as above, is likely to lead to inaccurate results because of conditioning problems and they develop a method for incrementally updating SVD solutions with a single observation, again without change of mean. Our experiments confirm that SVD methods are generally more accurate than EVD methods, but also show that a change of mean is of crucial importance to classification applications [9].

Later work with SVD allowed block updating [8], but neither dimension or mean changed. We have shown that SVD can be generalized for block updating, with a change of mean and dimension, provided all right singular vectors are maintained. We have not seen this published by others, but do have a derivation which is too long to include in this paper. Full downdating of SVD is not possible in any direct way. The reason is that the order of the right singular vectors depends upon the order of input points. Downdating is to remove some of these points and, hence, their input order is required.

2.3 Representing Observations

High-dimensional observations may be approximated by a low-dimensional vector using an eigenspace model. An n-dimensional observation $\underline{x}_n$ is represented using an eigenspace model $\Omega = (\underline{\bar{x}}, \underline{\underline{U}}_{np}, \underline{\underline{\Lambda}}_{pp}, N)$ as a p-dimensional vector $\underline{g}_p$:

$$\underline{g}_p = \underline{\underline{U}}_{np}^T(\underline{x}_n - \underline{x}). \quad (5)$$

This shifts the observation to the mean, and then represents it by components along each eigenvector. This is called the Karhunen-Loève transform [11].

The n-dimensional *residue vector* is defined by:

$$\underline{h}_n = \underline{x}_n - \underline{\underline{U}}_{np}\,\underline{g}_p \quad (6)$$

and $\underline{h}_n$ is orthogonal to every vector in $\underline{\underline{U}}_{np}$. Thus, $|\underline{h}_n|$ is the residue error in the representation of $\underline{x}_n$ with respect Ω.

3 Merging Eigenspace Models

We now turn our attention to one of the two main contributions of this paper, merging eigenspace models.

We derive a solution to the following problem. Let $\underline{\underline{X}}_{nN}$ and $\underline{\underline{Y}}_{nM}$ be two sets of observations. Let their eigenspace models be $\Omega = (\underline{\bar{x}}, \underline{\underline{U}}_{np}, \underline{\underline{\Lambda]}}_{pp}, N)$ and $\Psi = (\underline{\bar{y}}, \underline{\underline{V}}_{nq}, \underline{\underline{\Delta}}_{qq}, M)$, respectively. The problem is to compute the eigenspace model $\Phi = (\underline{\bar{z}}, \underline{\underline{W}}_{nr}, \underline{\underline{\Pi}}_{rr}, P)$, for $\underline{\underline{Z}}_{n(N+M)} = [\underline{\underline{X}}_{nN}\underline{\underline{Y}}_{nM}]$ using only Ω and Ψ.

Clearly, the total number of new observations is $P = N + M$. The combined mean is:

$$\underline{\bar{z}} \;=\; \frac{1}{P}\Big(N\underline{\bar{x}} + M\underline{\bar{y}}\Big). \tag{7}$$

The combined covariance matrix is:

$$\begin{aligned}\underline{\underline{E}}_{nn} &= \frac{1}{P}\left(\sum_{i=1}^{N}\underline{x}^i(\underline{x}^i)^T + \sum_{i=1}^{M}\underline{y}^i(\underline{y}^i)^T\right) - \underline{z}\underline{z}^T\\ &= \frac{1}{(N+M)}(N\underline{\underline{C}}_{nn} + N\underline{\bar{x}}\underline{\bar{x}}^T + M\underline{\underline{D}}_{nn} + M\underline{\bar{y}}\underline{\bar{y}}^T) - \underline{z}\underline{z}^T\\ &= \frac{N}{P}\underline{\underline{C}}_{nn} + \frac{M}{P}\underline{\underline{D}}_{nn} + \frac{NM}{P^2}(\underline{\bar{x}} - \underline{\bar{y}})(\underline{\bar{x}} - \underline{\bar{y}})^T,\end{aligned} \tag{8}$$

where the first two terms combine scaled versions of $\underline{\underline{C}}_{nn}$ and $\underline{\underline{D}}_{nn}$, the covariance matrices for $\underline{\underline{X}}_{nN}$ and $\underline{\underline{Y}}_{nM}$, respectively, and the final term allows for a change of mean (it is such a final a term that is omitted by solutions not allowing for a change of mean).

We wish to compute the s eigenvectors and eigenvalues that satisfy:

$$\underline{\underline{E}}_{nn} = \underline{\underline{W}}_{ns}\,\underline{\underline{\Pi}}_{ss}\,\underline{\underline{W}}_{ns}^T, \tag{9}$$

where some eigenvalues are subsequently discarded to give r nonnegligible eigenvectors and eigenvalues. The problem to be solved is of size s and this is necessarily bounded by

$$\max(p, q) \leq s \leq p + q + 1. \tag{10}$$

We explain perhaps the surprising additional 1 in the upper limit later (Section 3.1.1), but, briefly, it is needed to allow for the vector difference between the means, $\underline{\bar{x}} - \underline{\bar{y}}$.

3.1 Method of Solution

This problem may be solved in three steps:

1. Construct an orthonormal basis set, $\underline{\underline{\Upsilon}}_{ns}$, that spans both eigenspace models and $\underline{\bar{x}} - \underline{\bar{y}}$. This basis differs from the required eigenvectors, $\underline{\underline{W}}_{ns}$, by a rotation, $\underline{\underline{R}}_{ss}$, so that:
$$\underline{\underline{W}}_{ns} = \underline{\underline{\Upsilon}}_{ns}\,\underline{\underline{R}}_{ss}. \tag{11}$$
2. Use $\underline{\underline{\Upsilon}}_{ns}$ to derive a new eigenproblem. The solution of this problem provides the eigenvalues, $\underline{\underline{\Pi}}_{ss}$, needed for the merged eigenmodel. The eigenvectors, $\underline{\underline{R}}_{ss}$, comprise the linear transform that rotates the basis set $\underline{\underline{\Upsilon}}_{ns}$.
3. Compute the eigenvectors, $\underline{\underline{W}}_{ns}$, as above and discard any eigenvectors and eigenvalues using the chosen criteria (as discussed above) to yield $\underline{\underline{W}}_{nr}$ and $\underline{\underline{\Pi}}_{rr}$.

We now give details of each step.

3.1.1 Construct an Orthonormal Basis Set

To construct an orthonormal basis for the combined eigenmodels, we must choose a set of orthonormal vectors that span three subspaces: 1) The subspace spanned by eigenvectors $\underline{\underline{U}}_{np}$, 2) the subspace spanned by eigenvectors $\underline{\underline{V}}_{nq}$, and 3) the subspace spanned by $(\underline{\bar{x}} - \underline{\bar{y}})$. The last of these is a single vector. It is necessary because the vector joining the center of the two eigenspace models need not belong to either eigenspace. This accounts for the additional 1 in the upper limit of the bounds of s in (10). To see this, consider a pair of two-dimensional eigenspaces which are embedded in a three-dimensional space. The eigenvectors for each eigenspace could define parallel planes that are separated by a vector perpendicular to each of them. Clearly, a merged model should be a 3D ellipse and the vector between the origins of the models must contain a component perpendicular to both eigenspaces.

A sufficient spanning set is:

$$\underline{\underline{\Upsilon}}_{ns} \;=\; [\underline{\underline{U}}_{np}, \underline{\underline{\nu}}_{nt}], \tag{12}$$

where $\underline{\underline{\nu}}_{nt}$ is an orthonormal basis set for that component of the eigenspace of Ψ which is orthogonal to the eigenspace of Ω and, in addition, accounts for that component of $(\underline{\bar{x}} - \underline{\bar{y}})$ orthogonal to both eigenspaces; $t = s - p$.

To construct $\underline{\underline{\nu}}_{nt}$ we start by computing the residues of each of the eigenvectors in $\underline{\underline{V}}_{nq}$ with respect to the eigenspace of Ω:

$$\underline{\underline{G}}_{pq} \;=\; \underline{\underline{U}}_{np}^T\,\underline{\underline{V}}_{nq} \tag{13}$$

$$\underline{\underline{H}}_{nq} \;=\; \underline{\underline{V}}_{nq} - \underline{\underline{U}}_{np}\,\underline{\underline{G}}_{pq}. \tag{14}$$

The $\underline{\underline{H}}_{nq}$ are all orthogonal to $\underline{\underline{U}}_{np}$ in the sense that $(\underline{\underline{H}}^i)^T\underline{\underline{U}}^j = 0$ for all i, j. In general, however, some of the $\underline{\underline{H}}_{nq}$ are zero vectors because such vectors represent the intersection of the two eigenspaces. These zero vectors are removed to leave $\underline{\underline{H}}_{nq'}$. We also compute the residue $\underline{h}$ of $\underline{\bar{y}} - \underline{\bar{x}}$ with respect to the eigenspace of Ω, using (6).

$\underline{\underline{\nu}}_{nt}$ can now be computed by finding an orthonormal basis for $[\underline{\underline{H}}_{nq'}, \underline{h}]$, which is sufficient to ensure that $\underline{\underline{\Upsilon}}_{ns}$ is orthonormal. Gramm-Schmidt orthonormalization [12] may be used to do this:

$$\underline{\underline{\nu}}_{nt} = \mathrm{Orthonormalize}([\underline{\underline{H}}_{nq'}, \underline{h}]). \tag{15}$$

3.1.2 Forming a New Eigenproblem

We now form a new eigenproblem by substituting (12) into (11) and the result together with (8) into (9) to obtain:

$$\frac{N}{P}\underline{\underline{C}}_{nn} + \frac{M}{P}\underline{\underline{D}}_{nn} + \frac{NM}{P}(\underline{\bar{x}} - \underline{\bar{y}})(\underline{\bar{x}} - \underline{\bar{y}})^T = [\underline{\underline{U}}_{np}\;\underline{\underline{\nu}}_{nt}]\,\underline{\underline{R}}_{ss}\,\underline{\underline{\Pi}}_{ss}\,\underline{\underline{R}}_{ss}^T\,[\underline{\underline{U}}_{np}\;\underline{\underline{\nu}}_{nt}]^T.$$

Multiplying both sides on the left by $[\underline{\underline{U}}_{np}, \underline{\underline{\nu}}_{nt}]^T$, on the right by $[\underline{\underline{U}}_{np}, \underline{\underline{\nu}}_{nt}]$, and using the fact that $[\underline{\underline{U}}_{np}, \underline{\underline{\nu}}_{nt}]^T$ is a left inverse of $[\underline{\underline{U}}_{np}, \underline{\underline{\nu}}_{nt}]$, we obtain:

$$\begin{aligned}&[\underline{\underline{U}}_{np}\;\underline{\underline{\nu}}_{nt}]^T\left(\frac{N}{P}\underline{\underline{C}}_{nn} + \frac{M}{P}\underline{\underline{D}}_{nn} + \frac{NM}{P^2}(\underline{\bar{x}} - \underline{\bar{y}})(\underline{\bar{x}} - \underline{\bar{y}})^T\right)[\underline{\underline{U}}_{np}, \underline{\underline{\nu}}_{nt}]\\ &= \underline{\underline{R}}_{ss}\,\underline{\underline{\Pi}}_{ss}\,\underline{\underline{R}}_{ss}^T,\end{aligned} \tag{16}$$

which is a new eigenproblem whose solution eigenvectors constitute the $\underline{\underline{R}}_{ss}$ we seek and whose eigenvalues provide eigenvalues for the combined eigenspace model. We do not know the covariance matrices $\underline{\underline{C}}_{nn}$ or $\underline{\underline{D}}_{nn}$, but these can be eliminated, as we now show.

The first term in (16) can be approximated, using (4) and the fact that $\underline{\underline{U}}_{np}^T\,\underline{\underline{\nu}}_{nt} = \underline{\underline{0}}_{pt}$ to give

$$[\underline{\underline{U}}_{np}\underline{\underline{\nu}}_{nt}]^T\,\underline{\underline{C}}_{nn}\,[\underline{\underline{U}}_{np}, \underline{\underline{\nu}}_{nt}] \approx \begin{bmatrix}\underline{\underline{\Lambda}}_{pp} & \underline{\underline{0}}_{pt}\\ \underline{\underline{0}}_{tp} & \underline{\underline{0}}_{tt}\end{bmatrix}. \tag{17}$$

The second term in (16) can also be reduced. We have $\underline{\underline{D}}_{nn} \approx \underline{\underline{V}}_{nq}\,\underline{\underline{\Delta}}_{qq}\,\underline{\underline{V}}_{nq}^T$ (4) and $\underline{\underline{G}}_{pq} = \underline{\underline{U}}_{np}^T\underline{\underline{V}}_{nq}$ (13), which we use to obtain:

$$[\underline{\underline{U}}_{np}\;\underline{\underline{\nu}}_{nt}]^T\,\underline{\underline{D}}[\underline{\underline{U}}_{np}\;\underline{\underline{\nu}}_{nt}] \;\approx\; \begin{bmatrix}\underline{\underline{G}}_{pq}\,\underline{\underline{\Delta}}_{qq}\,\underline{\underline{G}}_{pq}^T & \underline{\underline{G}}_{pq}\,\underline{\underline{\Delta}}_{qq}\,\underline{\underline{\Gamma}}_{tq}^T\\ \underline{\underline{\Gamma}}_{tq}\,\underline{\underline{\Delta}}_{qq}\,\underline{\underline{G}}_{pq}^T & \underline{\underline{\Gamma}}_{tq}\,\underline{\underline{\Delta}}_{qq}\,\underline{\underline{\Gamma}}_{tq}^T\end{bmatrix}, \tag{18}$$

in which $\underline{\underline{\Gamma}}_{tq} = \underline{\underline{\nu}}_{nt}^T\underline{\underline{V}}_{nq}$, with $\underline{\underline{\nu}}_{nt}$, as in (15).

The final term in (16) we write as:

$$\begin{bmatrix} \underline{g}_p \underline{g}_p^T & \underline{g}_p \underline{\gamma}_t^T \\ \underline{\gamma}_t \underline{g}_p^T & \underline{\gamma}_t \underline{\gamma}_t^T \end{bmatrix}. \tag{19}$$

with $\underline{g}_p = \underline{\underline{U}}_{np}^T(\bar{\underline{x}} - \bar{\underline{y}})$, and $\underline{\gamma}_t = \underline{\underline{\nu}}_{nt}^T(\bar{\underline{x}} - \bar{\underline{y}})$.

So, the new eigenproblem to be solved may be approximated by

$$\begin{aligned} &\frac{N}{P}\begin{bmatrix} \underline{\underline{\Lambda}}_{pp} & \underline{\underline{0}}_{pt} \\ \underline{\underline{0}}_{tp} & \underline{\underline{0}}_{tt} \end{bmatrix} + \\ &\frac{M}{P}\begin{bmatrix} \underline{\underline{G}}_{pq}\underline{\underline{\Delta}}_{qq}\underline{\underline{G}}_{pq}^T & \underline{\underline{G}}_{pq}\underline{\underline{\Delta}}_{qq}\underline{\underline{\Gamma}}_{tq}^T \\ \underline{\underline{\Gamma}}_{tq}\underline{\underline{\Delta}}_{qq}\underline{\underline{G}}_{pq}^T & \underline{\underline{\Gamma}}_{tq}\underline{\underline{\Delta}}_{qq}\underline{\underline{\Gamma}}_{tq}^T \end{bmatrix} + \\ &\frac{NM}{P^2}\begin{bmatrix} \underline{g}_p \underline{g}_p^T & \underline{g}_p \underline{\gamma}_t^T \\ \underline{\gamma}_t \underline{g}_p^T & \underline{\gamma}_t \underline{\gamma}_t^T \end{bmatrix} = \underline{\underline{R}}_{ss}\underline{\underline{\Pi}}_{ss}\underline{\underline{R}}_{ss}^T. \end{aligned} \tag{20}$$

Each matrix is of size $s \times s$, where

$$s = p + t \leq p + q + 1 \leq \min(n, M + N).$$

Thus, we have eliminated the need for the original covariance matrices. Note this also reduces the size of the central matrix on the left hand side. This is of crucial computational importance because it makes the eigenproblem tractable in cases where the dimension of each datum is large, as is the case for image data.

3.1.3 Computing the Eigenvectors

The matrix $\underline{\underline{\Pi}}_{ss}$ is the eigenvalue matrix we set out to compute. The eigenvectors $\underline{\underline{R}}_{ss}$ comprise a rotation for $\underline{\underline{\Upsilon}}_{ns}$. Hence, we use (11) to compute the eigenvectors for $\underline{\underline{\Pi}}_{ss}$. However, not all eigenvectors and eigenvalues need be kept and some ($s - r$ of them) may be discarded using a criterion, as previously discussed in Section 2. This discarding of eigenvectors and eigenvalues will usually be carried out each time a pair of eigenspace models is merged.

Notice that there are two sources of error. The first is rounding error introduced because of finite machine precision. The second source of error is introduced by truncation of the eigenmodel, i.e., the discarding of eigenvectors and eigenvalues. This is the dominant source and its precise behavior deserves further investigation, both theoretically and empirically.

3.1.4 Comments on the Solution

This solution commutes and is associative (provided error terms are disregarded). It has an additive identity $(0, 0, 0, 0)$. It allows one or more observations to be added at any one time: Indeed, the above form reduces to that derived previously [9] when either of the models has just one point. The resulting model tends to some stable solution as the number of points in either model grows large. We note that there is no lower bound on the number of eigenvectors that must be retained, in contrast to SVD methods where all right singular vectors must be kept to block update while shifting the mean. Hence, EVD is often more efficient in terms of memory consumption. The time complexity to compute the EVD on an $(n \times n)$ matrix is usually $O(n^3)$. Hence, the time complexity of our method is expected to be $O(s^3)$ and this is born out by experiment (see Section 5).

4 Splitting Eigenspace Models

Here, we show how to split two eigenspace models. Given an eigenspace model $\Phi = (\underline{z}, \underline{\underline{W}}_{nr}, \underline{\underline{\Pi}}_{rr}, P)$, we remove $\Psi = (\underline{y}, \underline{\underline{V}}_{nq}, \underline{\underline{\Delta}}_{qq}, M)$ from it to give a third model $\Omega = (\underline{x}, \underline{\underline{U}}_{np}, \underline{\underline{\Lambda}}_{pp}, N)$. We use $\underline{\underline{\Pi}}_{rr}$, because $\underline{\underline{\Pi}}_{ss}$ is not available in general.

We ask the reader to carefully note that splitting means removing a subset of observations; the method is the inverse of merging in this sense. However, it is impossible to regenerate information which was discarded when the overall model was created (whether by batch methods or otherwise). Thus, if we split one eigenspace model from a larger one, the eigenvectors of the remnant must still form some subspace of the larger.

The derivation for splitting follows in a very straightforward way by analogy to that of merging. Therefore, we state the results for splitting. Clearly, $N = P - M$. The new mean is:

$$\bar{\underline{x}} = \frac{P}{N}\bar{\underline{z}} - \frac{M}{N}\bar{\underline{y}}. \tag{21}$$

As in the case of merging, new eigenvalues and eigenvectors are computed via a new eigenproblem. In this case, it is:

$$\frac{P}{N}\underline{\underline{\Pi}}_{rr} - \frac{M}{N}\underline{\underline{G}}_{rp}\,\underline{\underline{\Delta}}_{pp}\,\underline{\underline{G}}_{rp}^T - \frac{M}{P}\underline{g}_r\,\underline{g}_r^T = \underline{\underline{R}}_{rr}\,\underline{\underline{\Lambda}}_{rr}\,\underline{\underline{R}}_{rr}^T, \tag{22}$$

where $\underline{\underline{G}}_{rp} = \underline{\underline{W}}_{nr}^T\,\underline{\underline{V}}_{nq}$ and $\underline{g}_r = \underline{\underline{W}}_{nr}^T(\bar{\underline{y}} - \bar{\underline{x}})$.

The eigenvalues we seek are the q nonzero elements on the diagonal of $\underline{\underline{\Lambda}}_{rr}$. Thus, we can permute $\underline{\underline{R}}_{rr}$ and $\underline{\underline{\Lambda}}_{rr}$, and write without loss of generality:

$$\begin{aligned} \underline{\underline{R}}_{rr}\,\underline{\underline{\Lambda}}_{rr}\,\underline{\underline{R}}_{rr}^T &= [\underline{\underline{R}}_{rp}\ \underline{\underline{R}}_{rt}]\begin{bmatrix} \underline{\underline{\Lambda}}_{pp} & \underline{\underline{0}}_{pt} \\ \underline{\underline{0}}_{tp} & \underline{\underline{0}}_{tt} \end{bmatrix}[\underline{\underline{R}}_{rp}\ \underline{\underline{R}}_{rt}]^T \\ &= \underline{\underline{R}}_{rp}\,\underline{\underline{\Lambda}}_{pp}\,\underline{\underline{R}}_{rp}^T, \end{aligned} \tag{23}$$

where $p = r - q$.

Hence, we need only identify the eigenvectors in $\underline{\underline{R}}_{rr}$ with nonzero eigenvalues and compute the $\underline{\underline{U}}_{np}$ as:

$$\underline{\underline{U}}_{np} = \underline{\underline{W}}_{nr}\,\underline{\underline{R}}_{rp} \tag{24}$$

In terms of complexity, splitting must always involve the solution of an eigenproblem of size r. An algorithm for splitting may readily be written out using a similar approach to that for merging.

5 Results

Here, we examine the efficiency and accuracy of our methods, compared to batch methods. We used a database of 400 face images (each of $112 \times 92 = 10,304$ pixels) available online[1] in the tests reported here—similar results were obtained in tests with randomly generated data. The gray levels in the images were scaled into the range $[0, 1]$ by division only, but no other preprocessing was done. We implemented all functions using commercially available software on a computer with standard configuration (Sun Sparc Ultra 10, 300 Hz, 64 Mb RAM). The physical limit of our computer allowed us to store up to 300 images in RAM.

For all tests, the experimental procedure used was to compute eigenspace models using a batch method [10] and compare these to models produced by merging or splitting other models also produced by the batch method. In each case, the largest of the three data sets contained 300 images. These were partitioned into two data sets, each containing a multiple of 50 images. We included the degenerate cases when one model contained zero images. Note that we tested both smaller models merged with larger ones and vice-versa.

The number of eigenvectors retained in any model, including a merged model, was set to be 100 as a maximum, for ease of

1. The Olivetti database of faces: http://www.cam-orl.co.uk/facedatabase.html.

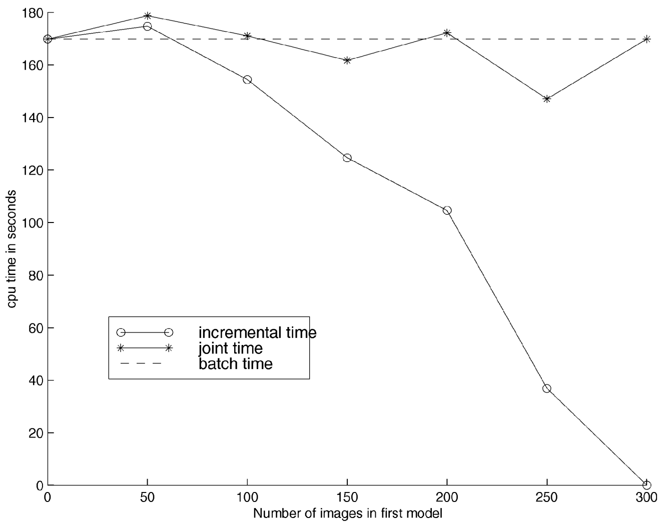

Fig. 1. Time to make a complete eigenspace model for a database of 300 images. The incremental time is the addition of the time to construct only the eigenspace to be added. The joint time is the time to compute both eigenspace models and merge them.

comparing results. (Initial tests using other strategies indicate that the resulting eigenspace model is little effected.)

5.1 Timing

When measuring CPU time we ran the same code several times and chose the smallest value to minimize the effect of other concurrently running process. Initially, we measured the time taken, using the batch methods, for data sets of different sizes. Results show a cubic dependency on the number of data points, as expected.

5.1.1 Merging

Experiments show that the time taken to *merge* two previously constructed models depends on the cube of the size of eigenproblem in (20). Because the eigenproblem we solve can be approximately as large as the total number of eigenvectors in both eigenmodels, it tends to be more expensive to merge two large eigenspaces (about 70 seconds for eigenspaces each of dimension 100) than merging eigenspaces of different sizes (about 5 seconds to merge a single observation with a model of dimension 100) . The order of eigenspaces makes no significant difference to the time complexity.

The total time taken to compute an eigenspace model from 300 images using the batch method and our merging method is compared in Fig. 1. The total time is a combination of construction and merging, as follows: The *incremental time* is the time needed to compute and then merge one eigenspace model to an existing one. As might be expected, incremental time falls as the additional number of new images falls. The *joint time* is the time to compute both eigenmodels and then merge them. The joint time is approximately constant and similar to the total time to compute a batch solution. However, recall that we experimented within the physical limits of our computer: When we built an eigenmodel using 400 images, the effects of memory paging were clearly visible: The "batch" time taken rose to over 800 seconds, whereas the joint time to produce an equivalent model tool less than 400 seconds.

5.1.2 Splitting

Time complexity for splitting eigenspaces should depend principally on the size of the large eigenspace from which the smaller space is being removed and the size of the smaller eigenspace should have little effect. This is because the size of the new eigenproblem to be solved depends on the size of the larger space and therefore dominates the complexity. These expectations are born out experimentally. We computed a large eigenmodel using 300 images, as before. We then removed smaller models of sizes between 50 and 250 images inclusive, in steps of 50 images. At most, 100 eigenvectors were kept in any model. The average time taken was approximately constant and ranged between 9 and 12 seconds, with a mean time of about 11.4 seconds. These figures are much smaller than those observed for merging because the large eigenspace contains only 100 eigenvectors. Thus, the matrices involved in the computation were of size (100×100), whereas in merging the size was at least (150×150) and other computations were involved (such as computing an orthonormal basis).

5.2 Similarity and Performance

We made a variety of similarity measures, intended to measure the similarity between a pair of eigenmodels, one incrementally computed, the other batch computed. We also measured some performance metrics, again for a pair of eigenmodels.

5.2.1 Merging

We first compared the means of the models produced by each method using Euclidean distance. This distance is greatest when the models to be merged have the same number of input images (150 in this case) as fall smoothly to zero when either of the models to be merged is empty. The value at maximum is typically very small and we measured it to be 3.5×10^{-14} units of gray level. This compares favorably with the working precision of our software, which is 2.2×10^{-16}.

We next compared the directions of the eigenvectors produced by each method, the error measured by the mean angular deviation of corresponding eigenvectors. Ignoring the degenerate cases, when one of the models is empty, the angular deviation has a single minimum when the eigenspace models were built with about the same number of images and grows larger as the sizes of the two models separate. This may be because, when a small model is added to a large model, its information tends to be swamped. However, the angular deviation to be very small on average, about 0.3 degrees at maximum.

The sizes of eigenvalues from both methods were compared next, more precisely the mean realtive absolute error was measured ($|a-b|/a$ for eigenvalues a and b). In general, we observed that the smaller eigenvalues had larger errors, as might be expected as they contain relatively little information and so are more susceptible to noise. The measure rises to a single peak when the number of input images in both models is the same (contrast with the angular error for eigenvectors) Even so, the maximal value is small, 7×10^{-3}.

We chose several performace metrics, such as residue error, Mahalanobis distance, and likelihood. Space permits we report only one. The mean difference in residue error is typically small, about 10^{-6} units of gray level per pixel, clearly below any noticeable effect. The measure is smallest when the eigenspaces merges are of equal size and grows as the size of the merged spaces differes (which compares with the form for angular deviation). Other performance metrics show similar behavior.

5.2.2 Splitting

Similar measures for splitting were computed using exactly those conditions described for testing the timing of splitting and for exactly those characteristics described for merging. In each case, a model to be subtracted was computed by a batch method and removed from the overall model by our splitting procedure. Also, a batch model was made for purposes of comparison with the residual data set. In all that follows, the phrase "size of the removed eigenspace" means the number of images used to construct the eigenspace removed from the eigenspace built from 300 images.

The Euclidean distance between the means of the models produced by each method grows monotonically as the size of the removed eigenspace falls and never exceeds about 1.5×10^{-13} gray-level units. Splitting is slightly less accurate in this respect than merging.

The mean angular deviation between corresponding eigenvector directions rises in similar fashion, from about 0.6 degrees when the size of the removed eigenspace is 250 to about 1.1 when the removed eigenmodel is of size 100. This represented a maximum in the deviation error because an error of about 1 degree was obtained when the removed model is of size 50. Again, these angular deviations are somewhat larger than those for merging.

The mean difference in eigenvalues shows the same general trend. Its maximum is about 0.5 units of gray level when the size of the removed eigenspace is 50. This is a much larger error than in the case of merging, but is still relatively small compared to a maximum eigenvalue of about 100. As in the case of merging, the deviation in eigenvalue grows larger as the size (importance) of the eigenvalue falls.

Difference in reconstruction error rises as the size of the removed eigenspace falls. Its size is of the order 10^{-4} units of gray level per pixel, which again is negligible. The overall trend is clear, accuracy and performance grew worse against any measure we used as the size of the eigenmodel being removed falls.

6 Applications

We now turn to applications of our methods. We have experimented with building point distribution models [13] of three-dimensional blood vessels and texture classification, while others have used similar methods for updating image motion parameters [8], selecting salient views [3], and building large image databases [3]. In this paper, we feel it is appropriate to discuss applications that are more general in nature; the intention is to furnish the reader with a practically useful appreciation of the characteristics of our methods and avoid the being diverted by any specific application.

An obvious application of our methods is to build an eigenspace for many images when there are too many to store in memory at once. This might arise in the case of very large databases and has been previously suggested [3]. Intuition suggests that images in the database will be better represented by the model if all of them are used in its construction; EVD (and SVD) fits a hyperplane to the data in the least-squares sense. Experiment bears this out—an eigenmodel built from a subset of data (as might be the only choice with batch methods) is often a poor model: better to use incrmental methods to build an eignmenodel of the full set.

We now turn to more substansive applications.

6.1 A Security Application

We consider a security application based on verification by classification—which requires a good estimate of the mean. The scenario is that of a company wishing to efficiently store photographs of its thousands of employees. We chose to store the data using an eigenmodel—the images can be projected into the eigenmodel and stored with tens rather than thousands of numbers. Conventional batch methods cannot be used to make the eigenmodel because not all images can fit into memory at once. Additionally, the database requires changing each year as employees come and go.

Our methods allow the eigenspace to be constructed and maintained. An initial eigenmodel is constructed by building several eigenspaces, each as large as possible, and merging them.

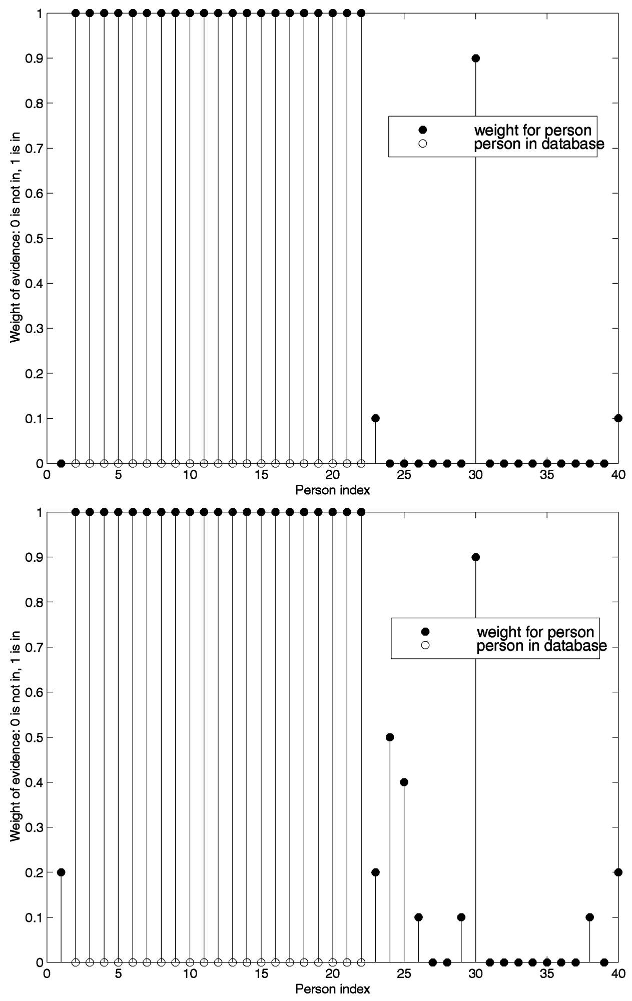

Fig. 2. Weight of evidence measures after a change: batch (top), incremental (below).

Thereafter, the eigenmodel can be maintained by simply merging or splitting eigenmodels as required.

We illustrate this with the database of faces used previously. We constructed an eigenmodel from a selection of 21 people, there being 10 photographs for each person. To recognize an individual, a new photograph was given a "weight of evidence" between 0 (not in the database) and 1 (in the database). To compute this weight, we used the maximum Mahalanobis distance (using Moghaddam and Pentland's method [6]) of all photographs used to construct the eigenmodel. Each new photograph was then judged as *in* if its Mahalanobis distance was less than this maximum. Since each person has 10 photographs associated with them, we can then compute a weight for each person as the fraction of their photographs classified as in.

This crude measure is sufficient to demonstrate that we can update image databases for classification using *some* measure—and this is our aim here.

We initialized the eigenmodel with the first 21 people (200 images). We then made a change by adding the 22nd person and removing the first—arbitrary but convenient choices. Fig. 2 show the "weight of evidence" measured after this change. The upper plot shows the measure for the images against a batch model. The lower plot shows the same measure for the same images. We notice that both models produce some false positives in the sense that some people who should not be classified as in have a weight larger than zero. We notice too that the incrementally computed eigenspace gives rise to more false positives than the eigenmodel

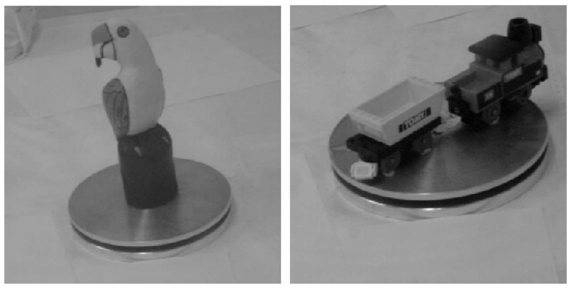
Fig. 3. Example images of each toy.

computed via batch methods—in line with earlier observations on subtraction. However, the weight-of-evidence factor is less than one in every case, no matter how the eigenmodel was computed, and this fact (or some other more sophisticated test and preprocessing) could be used to eliminate false positives—but the point here is not to develop a fully operational and robust security application, but to demonstrate the potential of our methods in classification.

We conclude that *additive* incremental eigenanalysis is safe for classification metrics, but that *subtractive* incremental eigenanalysis needs a greater degree of caution.

6.2 Dynamic Gaussian Mixture Models

We are interested in using our methods to construct dynamic Gaussian mixture models (GMMs). GMMs are increasingly common in the vision literature and a method for their dynamic construction would be useful. The methods presented in this paper make this possible. We note that block updating and maintainance of the mean are prerequisites for dynamic GMMs. Here, we focus on merging existing GMMs and outline how to construct a dynamic GMM from a library of photographs.

We partition data into sets, and for each set construct a GMM as follows: First, use all the data in a set to build an eigenmodel (using incremental methods if necessary). Second, project each datum in the set into the eigenmodel. Third, construct a GMM from the projected data, using the EM algorithm for this [14]. Finally, represent each Gaussian in the mixture using an eigenmodel. Hence, each GMM is a hierarchy of eigenspace models. In this regard, they are similar to a hierarchy of models proposed to improve the specificity of eigenmodels [15]. No two Gaussians need have the same dimension.

To merge GMMs, we first merged their base eigenspaces. Second, we transformed all dependent eigenmodels from each previous model into the new basis eigenspace. Finally, we merged those new dependent eigenspaces that were sufficiently close. We found that a simple volume measure to be adequate for most cases. The volume of a hyperellipse with semiaxes $\underline{A}$ (each element the square root of an eigenvalue), of dimension M, and at characteristic radius s (square root of the Mahalanobis distance) is $\frac{s^M|\underline{A}|\pi^{M/2}}{\Gamma(\frac{M}{2}+1)}$. We permanently merged a pair of eigenmodels in the GMM if the sum of their individual volumes was greater than their volume when merged. We found this works well enough, dimensionality problems notwithstanding.

As an example, we used photographs of two distinct toys, each photographed at 5 degree angles on a turntable. Hence, we had 144 photographs. Examples of these photographs can be seen in Fig. 3. The photographs were input in four groups of 36 photographs. For each group, we made an eigenmodel, projected the photographs into the eigenmodel, and used these projections to construct a GMM of 18 clusters. The Gaussians making up the mixture were represented by an eigenmodel. Hence, we had four GMMs, which we wanted to merge into a large GMM.

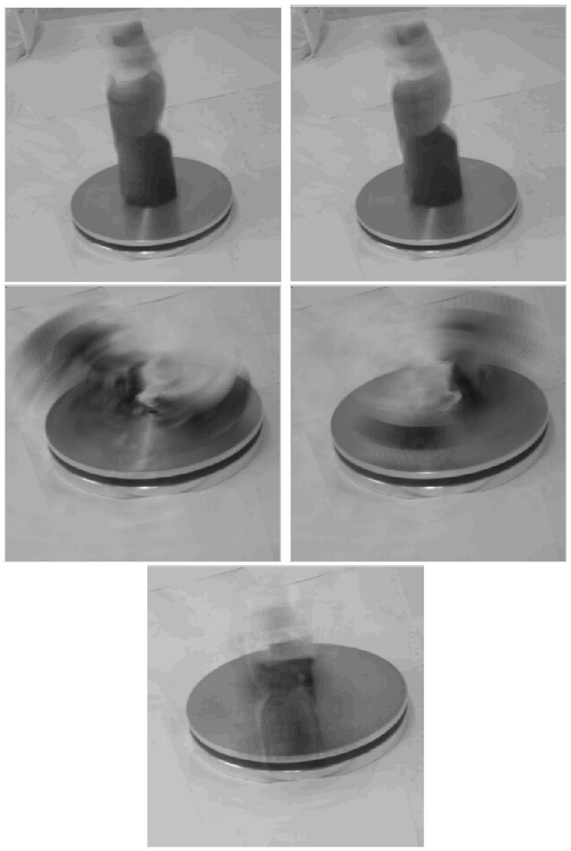
Fig. 4. Dynamic Gaussian Mixture Models, showing five examples of the 22 cluster centers. These are arranged to show clusters for each toy and the space between them.

To merge the GMMs, we first added added together the four eigenspaces to make a complete eigenspace. Next, we transformed each of the GMM clusters into this space, thus bringing the *ensemble* of clusters into a common space. Each Gaussian cluster in the mixture model in the new space was represented by an eigenmodel. We then merged the cluster, pairwise, using our volume criterion. Hence, we were able to reduce the number of Gaussians in the mixture to 22.

These clusters tend to model different parts of the cylindrical trajectories of the original data projected into the large eigenspace. Examples of cluster centers are shown in Fig. 4, where the two toys can be clearly seen in different positions. Such clusters may be used to identify the toy and its pose, for example [16], [17]. In addition, we found a few clusters occupying the space "in between" the two toys—an example of which is seen in Fig. 4. This artifact of clustering appears to derive from the high dimensionality of the space that the clusters are in, rather than being a side-effect of our method. These clusters might in future be removed because no picture matches well against them.

7 Conclusion

We have shown that merging and splitting eigenspace models is possible, allowing sets of new observations to be processed as a whole. The theoretical results are novel and our experimental results show that the methods are wholly practical, computation times are feasible, and often advantageous compared to batch methods. Batch and incremental eigenspaces are very similar, so performance characteristics, such as residue error, differ little. Our methods are useful in many applications and we have illustrated a few of a general nature.

We have concluded that the merging of eigenspaces is stable and reliable, but advise caution when splitting. Thus, splitting is the principle weakness of our methods and it is interesting to ask whether the process can be made more reliable.

We should point to several omissions from this work. We have not performed analytic error analysis, relying instead on experiment. Most of the errors arise from discarding eigenvectors and, eigenvalues. To the best of our knowledge, the work is unique and so, we have not compared our method to others. However, in a previous paper, we considered the inclusion of a single new datum and were able to make comparisons [9]. The conclusions there were that SVD tends to be more accurate and that updating the mean is crucial for classification applications. We note that in this paper, we have demonstrated that adding one datum each time is much less accurate than adding complete spaces. We have omitted fuller discussion of block update and downdate using SVD. We have presented general applications rather than any one, but have highlighted important areas such as dynamic GMMs.

We would expect our methods to find much wider applicability than those we have mentioned; updating image motion parameters [8], selecting salient views [3], and building large image databases [3] are two applications that exist already. We now use our methods routinely to construct eigenmodels that would be impossible by any other means and this has allowed us to experiment with image compression methods. Also, we have experimented with image segmentation, building models of three-dimensional blood vessels and texture classification. We believe that dynamic Gaussian mixture models provide an interesting future path.

REFERENCES

[1] J.R. Bunch, C.P. Nielsen, and D.C. Sorenson, "Rank-One Modification of the Symmetric Eigenproblem," *Numerische Mathematik,* vol. 31, pp. 31-48, 1978.

[2] J.R. Bunch and C.P. Nielsen, "Updating the Singular Value Decomposition," *Numerische Mathematik,* vol. 31, pp. 111-129, 1978.

[3] S. Chandrasekaran, B.S. Manjunath, Y.F. Wang, J. Winkler, and H. Zhang, "An Eigenspace Update Algorithm for Image Analysis," *Graphical Models and Image Processing,* vol. 59, no. 5, pp. 321-332, Sept. 1997.

[4] R.D. DeGroat and R. Roberts, "Efficient, Numerically Stablized Rank-One Eigenstructure Updating," *IEEE Trans. Acoustics, Speech, and Signal Processing,* vol. 38, no. 2, pp. 301-316, Feb. 1990.

[5] H. Murakami and B.V.K.V. Kumar, "Efficient Calculation of Primary Images from a Set of Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 4, no. 5, pp. 511-515, Sept. 1982.

[6] B. Moghaddam and A. Pentland, "Probabilistic Visual Learning for Object Representation," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 19, no. 7, pp. 696-710, July 1997.

[7] C. Nastar and N. Ayache, "Frequency-Based Nonrigid Motion Analysis: Application to Four Dimensional Medical Images," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 18, no. 11, pp. 1,067-1,079, Nov. 1996.

[8] S. Chaudhuri, S. Sharma, and S. Chatterjee, "Recursive Estimation of Motion Parameters," *Computer Vision and Image Understanding,* vol. 64, no. 3, pp. 434-442, Nov. 1996.

[9] P. Hall, D. Marshall, and R. Martin, "Incrementally Computing Eigenspace Models," *Proc. British Machine Vision Conf.,* pp. 286-295, 1998.

[10] L. Sirovich and M. Kirby, "Low-Dimensional Procedure for the Characterization of Human Faces," *J. Optical Soc. Am., A,* vol. 4, no. 3, pp. 519-524, Mar. 1987.

[11] Y.T. Chien and K.S. Fu, "On the Generalised Karhunen-Loève Expansion," *IEEE Trans. Information Theory,* vol. 13, pp. 518-520, 1967.

[12] G.H. Golub and C.F. Van Loan, *Matrix Computations.* Johns Hopkins, 1983.

[13] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham, "Training Models of Shape from Sets of Examples," *Proc. British Machine Vision Conf.,* pp. 9-18, 1992.

[14] A. Dempster, N. Laird, and D. Rubin, "Maximum Likelihood from Incomplete Data via the em Algorithm," *J. Royal Statistical Soc. B,* vol. 39, pp. 1-38, 1977.

[15] T. Heap and D. Hogg, "Improving Specificity in PDMS Using a Heirarchical Approach," *Proc. British Machine Conf.,* pp. 80-89, 1997.

[16] H. Murase and S.K. Nayar, "Visual Learning and Recognition of 3D Objects from Appearance," *Int'l J. Computer Vision,* vol. 14, no. 1, pp. 5-24, 1995.

[17] H. Borotschnig, L. Paltta, M. Prantl, and A. Pinz, "Active Object Recognition in Parametric Eigenspace," *Proc. British Machine Vision Conf.,* pp. 629-638, 1998.