Sequential Karhunen–Loeve Basis Extraction and its Application to Images

Avraham Levy and Michael Lindenbaum

***Abstract*—The *Karhunen–Loeve (KL) Transform* is an optimal method for approximating a set of vectors or images, which was used in image processing and computer vision for several tasks such as face and object recognition. Its computational demands and its batch calculation nature have limited its application. Here we present a new, sequential algorithm for calculating the KL basis, which is faster in typical applications and is especially advantageous for image sequences: the KL basis calculation is done with much lower delay and allows for dynamic updating of image databases. Systematic tests of the implemented algorithm show that these advantages are indeed obtained with the same accuracy available from batch KL algorithms.**

***Index Terms*—Karhunen–Loeve Transform, sequential algorithms, singular value decomposition.**

I. Introduction

THE *Karhunen–Loeve (KL) transform* [1] is a preferred method for approximating a set of vectors or images by a low dimensional subspace. The method provides the optimal subspace, spanned by the *KL basis*, which minimizes the MSE between the given set of vectors and their projections on the subspace. The KL transform has found many applications in traditional fields such as statistics [2] and communication [3]. In computer vision, it was used for a variety of tasks such as face recognition [4], [5], object recognition [6], motion estimation [7], [8], visual learning [9], and object tracking [10].

Typical computer vision applications calculate the KL basis of hundreds or thousands images, each of size M (= *Width* $\times$ *Height*) in the range of 10 K to 1 M. The basis is partial and typically includes only a few dozens vectors or less. Calculating the KL basis for N images of size M requires roughly $O(MN^2)$ operations. In many applications, this large computational demands may be prohibitive. In fact, for such applications, even the $O(MN)$ memory requirement may exceed the resources of common computers. Several attempts were already made to reduce the computational effort by using efficient image coding [11], [12].

Here, we suggest another approach to reducing the computational effort, relying on the relatively small dimension (denoted K) of the partial KL basis that is usually needed. Working in the context of image sets, where $M \gg N$, we propose a sequential algorithm that does not require to store the entire set of input images before proceeding to the calculation of the KL basis. Rather, it takes the images one by one (or in small groups—a possibility which has computational advantages), and updates an internal representation of the required KL basis.

The algorithm, named *Sequential Karhunen–Loeve algorithm* (SKL), essentially makes a sequence of Singular Value Decomposition (SVD) updating steps, leading to a low dimensional KL-basis of an image sequence. We optimize the number of images that should be added at every updating step, and derive the expected performance. In contrast to previous updating methods, we show that both the time and memory required are lower that those required by the standard batch algorithm, implying that the proposed algorithm is advantageous even when all the data is available before that algorithm starts. See Section III for the detailed algorithm.

A. Main Advantages of this New Algorithm

1) Reduced complexity and memory requirements: The proposed algorithm requires $O(MNK)$ operations and $O(MK)$ memory units in comparison to $O(MN^2)$ complexity and $O(MN)$ units of memory required by batch algorithms.

2) On-line features: Dealing with a sequence of images, the proposed algorithm may process the images as they arrive rather then wait for the end of the sequence.

3) Database update: The proposed, sequential, algorithm can easily handle continuous updating of the current eigenspace by new sets of images, without any requirement to keep the old images as well.

We found a few related works: Methods for SVD update, based on efficiently reducing an *arrowhead matrix* to tridiagonal form were proposed in [13], [14] but they do not lead to an efficient KL algorithm. A correlation-matrix-based method for updating the KL basis was suggested in the context of shape/motion recovery [15]. It requires however, that the correlation matrix rank is fixed and small (e.g., 3), and that the space dimension is low relative to the sequence length. Both demands are essential for efficiency but fail for images. The proposed SVD update stage shares some principles with the method suggested in the context of retrieving textual materials from scientific databases [16].

II. Batch, SVD Based, KL Algorithms

The discrete KL low dimension approximation problem is defined as follows. Given a data set of N vectors (images) $(A_i)_{i=1}^{N}$, find a set of K $(K < N)$ orthogonal unit vectors $(U_j)_{j=1}^{K}$ such that the projections of the data on the subspace spanned by the U_j vectors approximate the input vectors best. The meaning of "best approximation" is that the mean square error (over all input vectors) is minimal. The KL basis may be calculated either as the eigenvectors of the input vectors correlation matrix

Manuscript received July 28, 1998; revised February 13, 2000. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Patrick Bouthemy.

A. Levy is with HP Laboratories-Israel, Technion City, Haifa 32000, Israel (e-mail: avi@hpli.hpl.hp.com).

M. Lindenbaum is with the Department of Computer Science, Technion—Israel Institute of Technology, Haifa 32000 Israel (e-mail: mic@hpli.hpl.hp.com).

Publisher Item Identifier S 1057-7149(00)06137-6.

or using SVD: Let A be the data matrix whose columns are the input vectors A_i and compute the singular value decomposition $A = UDV^T$. The KL basis is spanned by the K most significant left singular vectors (the first K columns of U).

The following variation on the SVD algorithm, denoted R-SVD [17], is efficient when $M \gg N$.

1) Compute the QR decomposition $A = QR$ where Q is column-orthonormal $M \times N$ matrix and R is $N \times N$ upper triangular matrix ($4MN^2$ operations by the modified Gram-Schmidt algorithm [17]).
2) Compute the SVD of R, $R = U'DV^T$ by any standard SVD algorithm ($O(N^3)$ operations).
3) Calculate $U = QU'$ ($2MN^2$ operations).

$A = UDV^T$ is the SVD of A and the total computational complexity of the R-SVD algorithm is $6MN^2 + O(N^3)$.

III. Sequential KL/SVD Algorithm

Our approach to the analysis of large image sets (or sequences) is to use a sequential iterative algorithm which avoids both the storage of all the image data and its simultaneous processing. The algorithm takes the images in small "blocks" and updates an SVD internal representation of the required KL basis.

Based on our assumption that $M \gg N$, the proposed algorithm is based on the R-SVD algorithm. The key point in the R-SVD algorithm, which makes it useful as a basis for the sequential approach, is the fact that its complexity is due to the QR decomposition 1) and the update phase 3) while the contribution of the actual SVD computation 2) is negligible.

A. Partitioning the R-SVD Algorithm

The core of the proposed SKL algorithm is based on partitioning the SVD of a large matrix into two steps. Recall that the SVD of a $M \times L$ matrix B represents it as a product of three matrices, $B = UDV^T$, where U is the $M \times L$ columns orthogonal matrix of the (left) singular vectors, D is $L \times L$ diagonal matrix whose diagonal elements are the singular values of B and V is an $L \times L$ orthonormal matrix whose columns are the right singular vectors.

Let $B = UDV^T$ be the SVD of a matrix B. Efficiently calculating the SVD of a larger matrix $B^* = (B|E)$, where E is an $M \times P$ matrix consisting of P additional columns (images) can be done as follows.

- Perform an orthonormalization process on the matrix $(U|E)$, yielding the columns orthonormal matrix $U' = (U|\tilde{E})$.
- Let $V' = \left(\begin{array}{c|c} V & 0 \\ \hline 0 & I_P \end{array}\right)$ be a $(L+P) \times (L+P)$ matrix, where I_P is the P dimensional identity matrix.
- Let $D' = U'^T B^* V' = \left(\begin{array}{c} U^T \\ \hline \tilde{E}^T \end{array}\right)(B|E)\left(\begin{array}{c|c} V & 0 \\ \hline 0 & I_P \end{array}\right) = \left(\begin{array}{c|c} U^T BV & U^T E \\ \hline \tilde{E}^T BV & \tilde{E}^T E \end{array}\right)$. Note that $U^T BV = D$ and $\tilde{E}^T BV = 0$. Note also that the P rightmost columns of D' are the new vectors, added to B, represented in the orthonormal basis spanned by the columns of U'.
- Calculate the SVD $D' = \tilde{U}\tilde{D}\tilde{V}^T$. Then, the SVD of B^* is

$$B^* = U'(\tilde{U}\tilde{D}\tilde{V}^T)V'^T = (U'\tilde{U})\tilde{D}(\tilde{V}^T V'^T)$$

where $\tilde{D}$ is a diagonal $(L+P) \times (L+P)$ matrix, $U'\tilde{U}$ is an $M \times (L+P)$ column-orthonormal matrix and $\tilde{V}V'$ is an $(L+P) \times (L+P)$ orthonormal matrix.

Thus, calculating the SVD of B^* requires to calculate U', V', D' and the SVD of the small matrix D'. This update process is the core of the following sequential algorithm.

B. SKL Algorithm

The Sequential Karhunen-Loeve (SKL) algorithm is based on partitioning the data into blocks of column vectors (images). It starts by calculating the SVD of the first block. Then, at every step, another block of columns is added and the updated SVD is calculated using the "partitioned R-SVD" algorithm (Section III-A). Being interested only in a low dimensional approximation to the KL basis, the algorithm both deletes basis vectors (columns in the updated U matrices), corresponding to very small singular values and sets a limit on their number.

1) Input and parameters:

A	$M \times N$ data matrix.
P	Number of columns in each block.
K	Maximal number of columns for the U_i matrices.
ϵ	Lower bound for the singular values that are retained.
ff	Optional "forgetting factor" coefficient, which can reduce the contribution of previous blocks. See remark below.

2) Initialization—The 0th stage: Form the matrix A_0 from the first $K_0 = K$ columns of the matrix A. Compute the R-SVD $A_0 = U_0 D_0 V_0^T$ and keep U_0 and D_0.

3) ith stage: While some data of A have not been processed yet, do the following.

a) Read the matrix A_i, containing the next P columns of A, and append it to the matrix $ff \cdot U_{i-1}D_{i-1}$. Calculate the QR decomposition $(ff \cdot U_{i-1}D_{i-1}|A_i) = U'_{i-1}D'_{i-1}$ of the combined matrix, where U'_{i-1} is an $M \times (K_{i-1}+P)$ column orthonormal matrix and D'_{i-1} is upper diagonal of size $(K_{i-1}+P) \times (K_{i-1}+P)$ (in $4MP(K+P)$ operations).
b) Compute the SVD of D'_{i-1} as the product $D'_{i-1} = \tilde{U}_{i-1}\tilde{D}_{i-1}\tilde{V}^T$ (in negligible $O(K+P)^3$ operations).
c) Let $\sigma_i^1, \sigma_i^2, \cdots, \sigma_i^K$ be the K largest elements of the diagonal matrix $\tilde{D}_{i-1}$ (the singular values). Delete the rest of the singular values as well as the energetically negligible singular values σ_i^j, $j \leq K$, satisfying $(\sigma_i^j)^2 < \epsilon \sum_{k=1}^{K} (\sigma_i^k)^2$. Let D_i be the diagonal matrix whose elements are the remaining K_i singular values.
d) Remove from $\tilde{U}_{i-1}$ all the columns that corresponds to singular values that were removed above, and let $\hat{U}_{i-1}$ be the resulting matrix. Calculate $U_i = U'_{i-1}\hat{U}_{i-1}$ (in $2MK(K+P)$ operations).

4) Output: The matrix U_i at the final stage, after all columns of A has been processed.

5) Remarks:

1) Note that at stage a), the matrix $U_{i-1}D_{i-1}$ is already column orthogonal. Therefore, the QR decomposition [17] can be performed on the columns of A_i only.
2) In most image processing applications the maximal size, K, of the approximation basis is relatively small and the

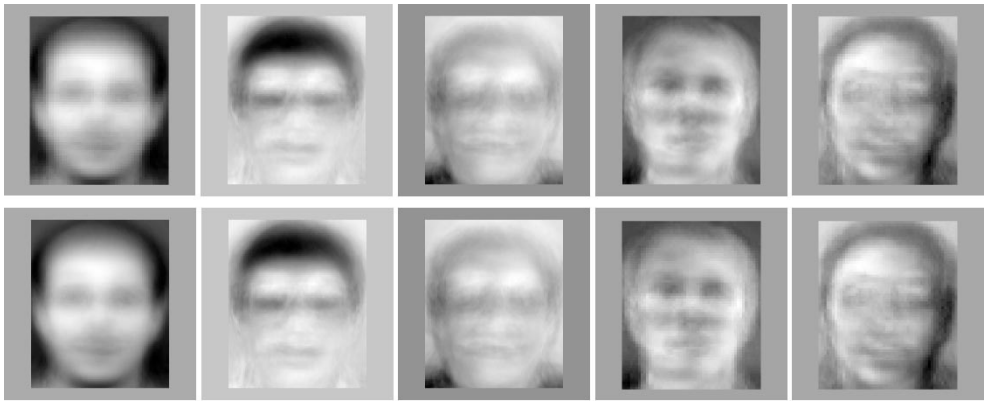


Fig. 1. The first five vectors of the KL basis (eigen-faces) produced by the standard KL algorithm (top) and the corresponding vectors produced by the proposed SKL. Scale inversion was done for some gray images to make the similarity clearer.

TABLE I
SKL TESTS ON EMSEMBLES OF FACES AND OBJECTS IMAGES. THE SKL-ERROR AND KL-ERROR ARE THE AVERAGE FRACTIONS OF THE TEST IMAGE (VECTOR) ENERGY, WHICH ARE NOT INCLUDED IN THE CORRESPONDING SUBSPACES. THE ENERGY RATIO IS THE RATIO OF THE IMAGE ENERGY CAPTURED (CONTAINED) IN THE SKL BASIS AND THE IMAGE ENERGY CONTAINED IN THE KL BASIS

Database	*Images no.*	*Basis dim.*	*SKL-Error*	*KL-Error*	*Energy-Ratio*
ORL	400	40	0.013168	0.013001	0.999830
ORL	400	20	0.018815	0.018804	0.999988
COIL-20	180	10	0.125238	0.122030	0.996346
COIL-20	380	20	0.123589	0.117479	0.993076

approximation is just sufficient for the task. Therefore, it usually makes sense to set ϵ to a very small positive value (not zero), implying that $K_i = K$ for each stage.

3) Often, when considering a time sequence of images, we wish to approximate the more recent images better. Setting the optional forgetting factor ff to a value smaller than one, enables this option.
4) The ith stage of the algorithm can be described concisely as follows: Given the basis matrix U_{i-1}, the (diagonal) singular value matrix D_{i-1} and the i-th batch of P input vectors, denoted A_i, compute the K most significant singular vectors of the matrix $(U_{i-1}D_{i-1}|A_i)$. The somewhat longer stage described at item (3) above has computational advantage in the case $M \gg N, K$.

C. Computational Effort Required for the SKL Algorithm

In the worst case, when at every step the number of columns in U_i is exactly K, the complexity of each step is approximately $4MP(K+P) + 2M(K+P)K$. Neglecting the contribution of the initialization step, the total complexity depends on the block size P, and is $C(P) = 2MN(K^2 + 3PK + 2P^2/P)$ (see [18] for details). For fixed M, N, K, the complexity $C(P)$ is minimized by a block size $P = (K/\sqrt{2})$, which nullify the derivative $C'(P)$. For this optimal value the total number of operations required for finding the partial KL basis using the SKL algorithm becomes $(4\sqrt{2}+6)MNK \cong 12MNK$ (in comparison to $O(MN^2)$ in traditional algorithms.) The total space required is $M(K+P)$ and is significantly smaller than the MN space usually required.

IV. SKL ALGORITHM—EXPERIMENTAL RESULTS

A. Running Time and Convergence Tests

We implemented both the proposed SKL algorithm and the traditional algorithms, and found that the expected computational gains are indeed achieved. For example, for $N = 200, K = 10, P = 7$ the running time of the SKL was 0.211 of the correlation matrix based KL algorithm time. The running time is indeed optimized by a block size $P \approx K/\sqrt{2}$ and we used this value in all experiments (see [18]).

We used synthetic data for verifying the convergence of the SKL algorithm and for testing its robustness under changing parameters. The input vectors were taken from a fixed dimensional subspace and were accompanied by substantial noise. A lower (K) dimensional KL subspace was constructed using the SKL algorithm. In all runs the SKL algorithm converges to the optimal subspace with very high level of numerical precision. We found that the rate of convergence depends mostly on the energy gap between the Kth and the $(K+1)$th basis vectors. When this gap is null, the K-dimensional optimal subspace is not unique and the algorithm provides one of the optimal subspaces. The algorithm converges to the optimal subspace even when very high noise (much more energetic than the weakest $(K$th) subspace component) is present. The presence of the noise reduces however the relative energy gap and slows convergence (see [18] for full details).

B. Tests on Real Imagery

Testing the algorithm on real images, for which the true basis is not available, we can only compare the results of the SKL algorithm with those provided by the standard KL algorithm. To avoid the problems associated with identical or very close eigenvalues (see above) we chose to do that by comparing the approximation properties of the two spaces.

The tests were carried on ensembles of sub-images taken from a larger image (see [18]) and on ensembles of images from the following two faces and objects databases.

1) Olivetti and Oracle Research Laboratory (ORL) database of faces (400 images). There are ten different images of each of 40 distinct subjects with varying lighting, facial expressions, and facial details (glasses/no glasses).

2) Columbia Object Image Library (COIL-20) database of objects (1440 images). There are 20 objects each with 72 images taken at different positions.

Table I summarizes typical results. The learning sets were the whole set of face images from the ORL database and two subsets of the COIL-20 objects database—corresponding to 9 and 19 positions for each object (180 and 380 images). The test sets consisted of one instance for each of the different subjects from the databases. That is, 40 images of faces from the ORL database and 20 images of objects from COIL-20. In all runs the block size of the SKL algorithm was approximately the basis dimension divided by $\sqrt{2}$. Apparently, the SKL algorithm and the KL algorithm practically output the same subspace. Observe also that the KL basis calculated by the SKL and the KL basis calculated by the KL algorithm (Fig. 1) are visually very similar up to gray level inversion, at least for the more energetic basis vectors. Visually observable differences appear only for the much less energetic vectors such as the 40th vector.

V. Discussion and Conclusions

The results of the experiments described in Section IV confirms that the SKL algorithm performs better than the standard KL algorithm while preserving a compatible level of approximation accuracy. We recall that the SKL algorithm requires $O(MK)$ space and runs in $O(MNK)$ time, in comparison to the standard KL algorithm that requires $O(MN)$ space and runs in $O(MN^2)$ time. The results presented in the tables of Section IV indicate that the SKL algorithm captures more than 99%, and in most cases close to 99.99%, of the energy content of the standard KL approximation.

The proposed SKL algorithm does not transform the data to zero mean input. From an approximation standpoint this is not a problem since increasing the dimension of the SKL basis by one to $K+1$ will yield better accuracy then the equivalent K dimensional zero mean approximation (which, for non zero mean data, requires the calculated average as a "hidden" $(K+1)$th component). A similar SKL algorithm, which transforms the data to zero mean input is obtained by subtracting the empirical mean, accumulated up to this point, from the input before it enters the update procedure.

The fact that the basis created by the SKL algorithm, is such a good approximation for the true KL basis, in spite of the curtailing of the temporary basis at each computational stage is not self evident. In [18], we explained this fortunate behavior using a probabilistic model for the data, and have shown that, due to coherency, even true subspace components which are much weaker than the noise, can beat it and claim their part in the resulting space.

In conclusion we can say that the SKL algorithm ca be utilized for computing an optimal low dimensional basis for problems associated with very large data set and for problems with intrinsic sequential nature. Its reduced computation and memory requirements and its update capability make it superior to the standard KL algorithm in many applications. We note that farther improvement to the SKL algorithm performance can be achieved if it can be terminated at some stage by utilizing a stopping rule. Such a stopping rule, based on some convergence criteria, should be able to decide when the low dimensional KL basis has reached a satisfactory level of energy approximation content.

References

[1] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. New York: Academic, 1990.

[2] D. Brillinger, *Time Series—Data Analysis and Theory*. New York: Holden-Day, 1981.

[3] H. L. Van Trees, *Detection, Estimation, and Modulation Theory*. New York: Wiley, 1965.

[4] L. Sirovich and M. Kirby, "Low dimensional procedure for the characterization of human faces," *J. Opt. Soc. Amer. A*, vol. 4, no. 3, pp. 519–524, 1987.

[5] M. Turk and A. P. Pentland, "Face recognition using eigenfaces," *CVPR'91*, pp. 586–591, 1991.

[6] H. Murase and S. K. Nayar, "Visual learning and recognition of 3-d objects from appearance," *Int. J. Comput. Vis.*, vol. 14, pp. 5–24, Jan. 1995.

[7] M. Lindenbaum, "Direct positioning from grey leve images," CS dept., Technion, Tech. Rep. CIS 9519, 1995.

[8] G. D. Hager and P. N. Belhumeur, "Real time tracking of image regions with changes in geometry and illumination," in *CVPR'96*, 1996, pp. 403–410.

[9] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 19, pp. 696–710, July 1997.

[10] M. J. Black and A. D. Jepson, "Eigentracking: Robust matching and tracking of articulated objects using a view-based representation," *Int. J. Comput. Vis.*, vol. 26, pp. 63–84, Jan. 1998.

[11] J. B. Roseborough and H. Murase, "Partial eigenvalue decomposition for large image sets using run-length encoding," *Pattern Recognit.*, vol. 28, no. 3, pp. 421–430, 1995.

[12] H. Murase and M. Lindenbaum, "Spatial temporal adaptive method for partial eigenstructure decomposition of large image matrices," *IEEE Trans. Image Processing*, vol. 4, no. 5, pp. 620–629, 1995.

[13] H. Zha, "A two way chasing scheme for reducing a symmetric arrowhead matrix to tridiagonal form," *J. Numer. Lin. Alg. Applicat.*, vol. 1, no. 1, pp. 49–57, 1992.

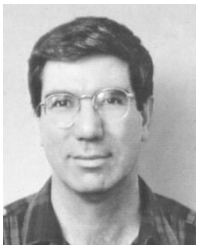
[14] S. Van Huffel and H. Park, "Parallel tri- and bi-diagonalization of border bidiagonal matrices," *Parallel Comput.*, vol. 20, pp. 1107–1128, 1994.

[15] T. Kanade and T. Morita, "A sequential factorization method for recovering shape and motion from image streams," in *ARPA'94*, vol. II, 1994, pp. 1177–1187.

[16] M. W. Berry, S. T. Dumais, and G. W. Obrien, "Using linear algebra for intelligent information retrieval," *SIAM Rev.*, vol. 37, no. 4, pp. 573–595, 1995.

[17] G. Golub and C. Van Loan, *Matrix Computations*. Baltimore, MD: Johns Hopkins Univ. Press, 1992.

[18] A. Levy, "Optimal Low Diemensional Approximation—Theoretical and Computational Aspects," Ph.D. dissertation, Technion, Haifa, Israel, 1999.



Avraham Levy was born in Israel in 1955. He received the B.Sc. and Ph.D. in mathematics from the Technion, Haifa, Israel, in 1977 and 1999, respectively.

Since October 1998, he has been with HP Laboratories—Israel, Haifa. His main research interest is mathematical methods in image processing.

Michael Lindenbaum was born in Israel in 1956. He received the B.Sc., M.Sc., and D.Sc. degrees in electrical engineering from the Technion, Haifa, Israel, in 1978, 1987, and 1990, respectively.

From 1978 to 1985, he served in the Israel Defense Forces. He did his postdoctoral work at the NTT Basic Research Labs, Tokyo, Japan, and since October 1991, he has been with the Department of Computer Science, Technion. His main research interest is computer vision, and especially statistical analysis of object recognition and grouping processes.