Int J Comput Vis (2008) 76: 217–229
DOI 10.1007/s11263-007-0053-0

A Tensor Approximation Approach to Dimensionality Reduction

Hongcheng Wang · Narendra Ahuja

Received: 6 October 2005 / Accepted: 9 March 2007 / Published online: 30 June 2007
© Springer Science+Business Media, LLC 2007

Abstract Dimensionality reduction has recently been extensively studied for computer vision applications. We present a novel multilinear algebra based approach to reduced dimensionality representation of multidimensional data, such as image ensembles, video sequences and volume data. Before reducing the dimensionality we do not convert it into a vector as is done by traditional dimensionality reduction techniques like PCA. Our approach works directly on the multidimensional form of the data (matrix in 2D and tensor in higher dimensions) to yield what we call a Datum-as-Is representation. This helps exploit spatio-temporal redundancies with less information loss than image-as-vector methods. An efficient rank-R tensor approximation algorithm is presented to approximate higher-order tensors. We show that rank-R tensor approximation using Datum-as-Is representation generalizes many existing approaches that use image-as-matrix representation, such as generalized low rank approximation of matrices (GLRAM) (Ye, Y. in Mach. Learn. 61:167–191, 2005), rank-one decomposition of matrices (RODM) (Shashua, A., Levin, A. in CVPR'01: Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition, p. 42, 2001) and rank-one decomposition of tensors (RODT) (Wang, H., Ahuja, N. in ICPR '04: ICPR '04: Proceedings of the 17th international conference on pattern recognition (ICPR'04), vol. 1, pp. 44–47, 2004). Our approach yields the most compact data representation among all known image-as-matrix methods. In addition, we propose another rank-R tensor approximation algorithm based on slice projection of third-order tensors, which needs fewer iterations for convergence for the important special case of 2D image ensembles, e.g., video. We evaluated the performance of our approach vs. other approaches on a number of datasets with the following two main results. First, for a fixed compression ratio, the proposed algorithm yields the best representation of image ensembles visually as well as in the least squares sense. Second, proposed representation gives the best performance for object classification.

Keywords Rank-R tensor approximation · Multilinear analysis · Dimensionality reduction · Object recognition

1 Introduction

Many computer vision applications require processing large amounts of multidimensional data, such as face/object databases for recognition or retrieval, video sequences for security and surveillance and 3D/4D CT/fMRI images for medical analysis. The dimensions can be a mix of space and time, e.g., a video sequence where two of the dimensions are spatial and the third temporal. Datum is recognized as a basic element of multidimensional data, for example, each datum in a video sequence is a 2D image and each datum in a 4D fMRI sequence is a 3D volume. As data size and the amount of redundancy increase fast with dimensionality, it is desirable to obtain compact and concise representations of

A shorter version of this paper was published at IEEE CVPR 2005 (Wang and Ahuja 2005).

H. Wang (✉)
United Technologies Research Center (UTRC), Each Hartford, CT, USA
e-mail: wanghc@vision.ai.uiuc.edu

N. Ahuja
University of Illinois at Urbana-Champaign (UIUC), Urbana, IL, USA
e-mail: ahuja@vision.ai.uiuc.edu

the data, e.g., by identifying suitable basis functions for subspace representation. The process of finding a set of compact bases and projections of the data on these bases as representation is referred to as dimensionality reduction.

Many computer vision applications require processing large amounts of multidimensional data, such as face/object databases for recognition or retrieval, video sequences for security and surveillance and 3D/4D CT/fMRI images for medical analysis. The dimensions can be a mix of space and time, e.g., a video sequence where two of the dimensions are spatial and the third temporal. Datum is recognized as a basic element of multidimensional data; for example, each datum in a video sequence is a 2D image and each datum in a 4D fMRI sequence is a 3D volume. As data size and the amount of redundancy increase fast with dimensionality, it is desirable to obtain compact and concise representations of the data, e.g., by identifying suitable basis functions for subspace representation. The process of finding a set of compact bases and projections of the data on these bases as representation is referred to as dimensionality reduction.

Traditional methods for reducing the dimensionality of image ensembles usually transform each datum (an image) into a vector by concatenating rows (we call it *Image-as-Vector*). One example of these methods is principal component analysis (PCA), which has been widely used in face representation (Sirovich and Kirby 1987), face recognition (Turk and Pentland 1991), and many other applications. PCA is used to find a set of mutually orthogonal basis functions which capture the largest variation in the training data. The features are obtained by projecting each zero mean image onto a p dimensional subspace, which can be obtained by the singular value decomposition (SVD). Independent component analysis (ICA) is another method used for image coding. The ICA (Comon 1994) model is a generative model, which describes how the images are generated by a process of mixing a set of independent components. ICA is very closely related to blind source separation (BSS) (Jutten and Herault 1991). There are many other methods, such as, minimum entropy coding (Barlow 1989) and sparse coding using simple-cell receptive field properties (Olshausen and Field 1996). Recently a hybrid linear model (Hong et al. 2005) has been proposed for image representation. Tenenbaum and Freeman (2000) presented a bilinear model for separating the underlying two factors (so-called content and style) of perceptual observations. Multilinear algebra has recently received much attention in computer vision and signal processing. High order singular value decomposition (HOSVD) was discussed in (Lathauwer et al. 2000), and has been used in computer vision applications such as face recognition (Vasilescu and Terzopoulos 2002) and facial expression decomposition (Wang and Ahuja 2003). It was shown in (Vasilescu and Terzopoulos 2003) that the root mean squared error (RMSE) of the reconstruction for the same compression ratio is higher for HOSVD using Image-as-Vector representation than PCA. All these PCA- or tensor-based techniques adopt the Image-as-Vector representation by concatenating image rows into a single vector. One inherent problem of this representation is that the spatial redundancy within each image matrix is not fully utilized, and some information on local spatial relationships is lost.

Some researchers in computer vision and machine learning have recently begun to treat an image as a matrix (we call these methods *Image-as-Matrix* methods) (Shashua and Levin 2001; Yang et al. 2004; Ye 2005). Shashua and Levin (2001) proposed representing a collection of images using a set of rank-one matrices. Yang et al. (2004) recently proposed a two-dimensional PCA (2DPCA) by constructing an image covariance matrix using the original image matrices. As noted in (Yang et al. 2004), 2DPCA-based image representation is not as memory efficient as PCA since 2DPCA requires more coefficients than PCA. Ye (2005) proposed a method called generalized low rank approximation of matrices (GLRAM). In contrast to PCA, GLRAM projects the original data onto a (p_1, p_2)-dimensional space such that the projection has the largest variance among all (p_1, p_2)-dimensional spaces. We will show that all these methods assume independency among all rows/columns of each image, as a result of which they can not fully capture pixel-pairwise redundancies. More importantly, it is hard to generalize these methods to higher dimensional datasets such as a collection of 3D volumes.

In this paper, we propose a tensor decomposition framework for dimensionality reduction using a *Datum-as-Is* representation. The datum can be a 2D image, or a 3D or higher-dimensional volume. The advantages of our tensor decomposition framework using Datum-as-Is representation lie in the following three aspects.

- *Capturing redundancies across different modes of data*: Elements in image rows or columns are often similar. Using Datum-as-Is representation can not only capture element-pairwise variations, but also row/column variations.
- *Generalization*: It is easy to generalize our framework to higher-dimensional data, such as a collection of 3D volumes. It will be shown that three existing image-as-matrix algorithms are special cases of our tensor framework.

A recent independent work using a similar representation (Xu et al. 2005) showed that this can also avoid the curse of dimensionality dilemma in image analysis. We also proposed an out-of-core tensor approximation algorithm using Datum-as-Is representation to handle large-scale multidimensional visual data in image-based rendering (Wang et al. 2005).

The rest of the paper is organized as follows. We first give a brief overview of multilinear algebra and the formulation of rank-R approximation of tensors in Sect. 2. Then

we describe two algorithms for this problem: generalized rank-R approximation of tensors in Sect. 3 and rank-R approximation of third-order tensors in Sect. 4. The algorithms are used for dimensionality reduction and feature extraction in Sect. 5. In Sect. 6, we report experimental results on the quality and computational complexity of the representation, and its efficacy in object recognition. Conclusions are given in Sect. 7.

2 Overview of Multilinear Algebra

In this section, we introduce the relevant preliminary material concerning multilinear algebra. The notations we use are described in Table 1.

A high-order tensor is denoted as: $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$. The n-mode product of a tensor $\mathcal{A}$ by a matrix $U \in^{J_n \times I_n}$, denoted by $\mathcal{A} \times_n U$, is defined by a tensor with entries: $(\mathcal{A} \times_n U)_{i_1 \ldots i_{n-1} j_n i_{n+1} \ldots i_N} = \sum_{i_n} a_{i_1 \ldots i_N} u_{j_n i_n}$. The scalar product of two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is defined as: $\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} a_{i_1 i_2 \cdots i_N} b_{i_1 i_2 \cdots i_N}$. The Frobenius norm of a tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$ is then defined as $\|\mathcal{A}\| = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$. The n-rank of $\mathcal{A}$, denoted by $R_n = \text{rank}_n(\mathcal{A})$, is the dimension of the vector space spanned by the n-mode vectors. For example, an Nth-order tensor has rank 1 when it equals the outer product of N vectors $U^1, U^2, \ldots, U^N$, i.e., $a_{i_1 i_2 \ldots i_N} = u^{(1)}_{i_1} u^{(2)}_{i_2} \ldots u^{(N)}_{i_N}$, for all values of the indices, written as: $\mathcal{A} = U^{(1)} \circ U^{(2)} \circ \cdots \circ U^{(N)}$. Unfolding a tensor $\mathcal{A}$ along the nth mode is denoted as $unf(\mathcal{A}, n)$ or $A_{(n)}$. The unfolding of a third order tensor is illustrated in Fig. 1.

Two important properties can be observed from the unfolding of a third-order tensor (e.g., an image ensemble by representing each image as a matrix) in Fig. 1.

Observation 1 *The unfolding matrices of an image ensemble tensor are usually long, thin matrices. For example,* $A_{(1)} \in \mathbb{R}^{I_1 \times I_2 I_3}$, *with* $I_1 \ll I_2 I_3$.

Table 1 Notation

Notation	Descriptions
$\mathcal{A}, \mathcal{B}, \ldots$	tensor (calligraphic letters)
$A, B, \ldots$	matrix (capitals)
a, b	vector (low-case letters)
$A_{(i)}$	unfolding tensor along the ith mode
A_i	ith image
$U^{(i)}$	subspace matrix along the ith mode
N	the order of a tensor
p	the number of principal components
r	the number of rank-one tensor bases
d	the dimension of GLRAM
R	the dimension of rank-R tensor approximation

Observation 2 *The columns in* $A_{(1)}$ *correspond to the columns of all images, the columns in* $A_{(2)}$ *correspond to the rows of all images, and the rows in* $A_{(3)}$ *correspond to images-as-vector.*

Observation 1 is very useful for efficient implementation of rank-R tensor approximation algorithm, while Observation 2 could help us better understand why our method can capture more redundancies than image-as-vector methods. We will elaborate on these in Sect. 3.

Like SVD for matrices, HOSVD has been recently developed for tensors (Lathauwer et al. 2000) as in the theorem below.

Theorem 1 *Any tensor* $\mathcal{A} \in \mathbb{C}^{I_1 \times I_2 \times \cdots \times I_N}$ *can be expressed as the product*:

$$\mathcal{A} = \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times \cdots \times_N U^{(N)}, \tag{1}$$

with the following properties:

(1) $U^{(n)} = (U^{(n)}_1 U^{(n)}_2 \ldots U^{(n)}_{I_n})$ *is a unitary* $(I_n \times I_n)$ *matrix.*

(2) *The subtensors* $\mathcal{B}_{i_n=\alpha}$ *of* $\mathcal{B} \in \mathbb{C}^I_1 \times I_2 \times \cdots \times I_N$ *have the following properties*:

- *All-orthogonality*: *two subtensors* $\mathcal{B}_{i_n=\alpha}$ *and* $\mathcal{B}_{i_n=\beta}$ *are orthogonal for all possible values of* n, α *and* β *subject to* $\alpha \neq \beta$: $\langle \mathcal{B}_{i_n=\alpha}, \mathcal{B}_{i_n=\beta} \rangle = 0$, *when* $\alpha \neq \beta$,
- *Ordering*: $\|\mathcal{B}_{i_n=1}\| \geq \|\mathcal{B}_{i_n=2}\| \geq \cdots \geq \|\mathcal{B}_{i_n=I_n}\| \geq 0$ *for all possible values of* n.

Therefore, computing the HOSVD of an Nth-order tensor leads to the computation of N different matrix SVDs of matrices with size $I_n \times I_1 I_2 \ldots I_{n-1} I_{n+1} \ldots I_N$, $1 \leq n \leq N$.

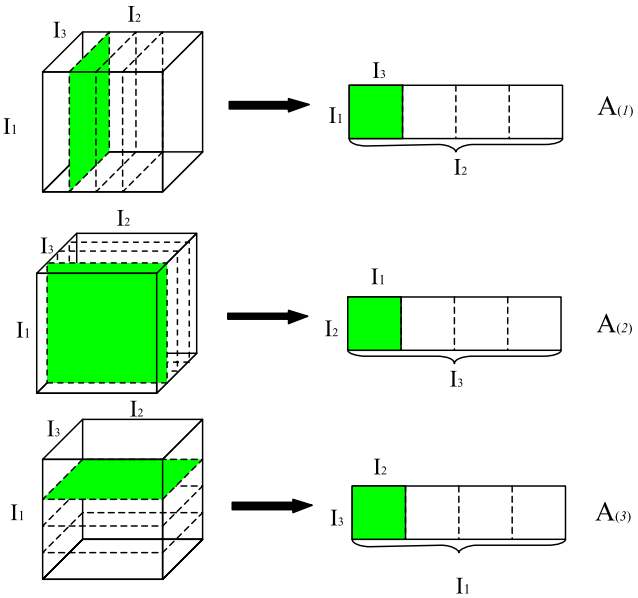

Fig. 1 Unfolding of a third-order tensor

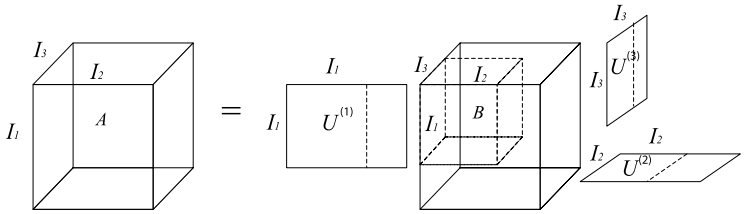

Fig. 2 HOSVD of a third-order tensor

The HOSVD of a third-order tensor is illustrated in Fig. 2.

Kofidis and Regalia (2002), Lathauwer et al. (2000), Lathauwer et al. (2000) are good sources of details of multilinear algebra.

3 Generalized Rank-R Approximation of Tensors

3.1 Tensor Approximation Algorithm

Given a real Nth-order tensor $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, find a tensor $\tilde{\mathcal{A}} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, having $\text{Rank}_1(\tilde{\mathcal{A}}) = R_1, \text{Rank}_2(\tilde{\mathcal{A}}) = R_2, \ldots, \text{Rank}_N(\tilde{\mathcal{A}}) = R_N$, that minimizes the least-squares cost function

$$\tilde{\mathcal{A}} = \arg\min_{\hat{\mathcal{A}}} \|\mathcal{A} - \hat{\mathcal{A}}\|^2. \tag{2}$$

The desired tensor is represented as

$$\tilde{\mathcal{A}} = \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times \cdots \times_N U^{(N)} \tag{3}$$

where $U^{(1)} \in \mathbb{R}^{I_1 \times R_1}$, $U^{(2)} \in \mathbb{R}^{I_2 \times R_2}$, $\ldots$, $U^{(N)} \in \mathbb{R}^{I_N \times R_N}$ and $\mathcal{B} \in \mathbb{R}^{R_1 \times R_2 \times \cdots \times R_N}$. $U^{(i)}$ has orthonormal columns for $1 \le i \le N$.

For image ensembles, we use third-order tensors, $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times I_3}$ ($I_1 \times I_2$ is the dimensionality of each image, and I_3 is the number of images). We also assume $R_1 = R_2 = R_3 = R$ here for simplicity. Hence we name our approach rank-R approximation of tensors.

From (2), i.e.,

$$f = \|\mathcal{A} - \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times \cdots \times_N U^{(N)}\|^2. \tag{4}$$

For given matrices $U^{(1)}, U^{(2)}, \ldots, U^{(N)}$, $\mathcal{B}$ can be obtained by solving a classical linear least-squares problem: $\mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times \cdots \times_N U^{(N)} = \mathcal{A}$. Since $U^{(1)}, U^{(2)}, \ldots, U^{(N)}$ have orthonormal columns, we have $\mathcal{B} = \mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T} \times \cdots \times_N U^{(N)^T}$. As stated in (Lathauwer et al. 2000), the minimization in (2) is equivalent to the maximization, over the matrices $U^{(1)}, U^{(2)}, \ldots, U^{(N)}$ having orthonormal columns, of the function

$$g(U^{(1)}, \ldots, U^{(N)}) = \|\mathcal{A} \times_1 U^{(1)^T} \times \cdots \times_N U^{(N)^T}\|^2. \tag{5}$$

We apply the alternative least squares (ALS) (Kroonenberg 1983; Lathauwer et al. 2000) to find the (local) optimal solution of (2). In each step, we optimize only one of the matrices, while keeping others fixed. For example, with $U^{(1)}, \ldots, U^{(n-1)}, U^{(n+1)}, \ldots, U^{(N)}$ fixed, we project tensor $\mathcal{A}$ onto the $(R_1, \ldots, R_{n-1}, R_{n+1}, \ldots, R_N)$-dimensional space, i.e., $U^{(n)}_{j+1} = \mathcal{A} \times_1 U^{(1)^T}_{j+1} \times \cdots \times_{n-1} U^{(n-1)^T}_{j+1} \times_{n+1} U^{(n+1)^T}_{j} \times \cdots \times_N U^{(N)^T}_{j}$, and then the columns of $U^{(n)}$ can be found as an orthonormal basis for the dominant subspace of the projection.

Algorithm 1: Rank-$(R_1, R_2, \ldots, R_N)$ tensor approximation

Data: Given $\mathcal{A}$ and $R_1, R_2, \ldots, R_N$

Result: Find U^n, $(1 \le n \le N)$ and $\mathcal{B}$.

Initialize $U^{(n)}_0 \in \mathbb{R}^{I_n \times R_n}$, $1 \le n \le N$;

while $\sim$ *stop* **do**

$\tilde{U}^{(1)}_{j+1} = uf(\mathcal{A} \times_2 U^{(2)^T}_j \times_3 U^{(3)^T}_j \times \cdots \times_N U^{(N)^T}_j, 1)$;

$U^{(1)}_{j+1} = svds(\tilde{U}^{(1)}_{j+1}, R_1)$;

$\tilde{U}^{(2)}_{j+1} = uf(\mathcal{A} \times_1 U^{(1)^T}_{j+1} \times_3 U^{(3)^T}_j \times \cdots \times_N U^{(N)^T}_j, 2)$;

$U^{(2)}_{j+1} = svds(\tilde{U}^{(2)}_{j+1}, R_2)$;

$\cdots$

$\tilde{U}^{(N)}_{j+1} = uf(\mathcal{A} \times_1 U^{(1)^T}_{j+1} \times_2 U^{(2)^T}_{j+1} \times \cdots \times_(N-1) U^{(N-1)^T}_{j+1}, N)$;

$U^{(N)}_{j+1} = svds(\tilde{U}^{(N)}_{j+1}, R_N)$;

$\mathcal{B} = \mathcal{A} \times_1 U^{(1)^T}_{j+1} \times_2 U^{(2)^T}_{j+1} \times \cdots \times_N U^{(N)^T}_{j+1}$;

if $(\|\mathcal{B}_{j+1}\|^2 - \|\mathcal{B}_j\|^2 < \varepsilon)$ *stop*

end

The pseudo-code of the algorithm is described in Algorithm 1. The "$svds(A, R)$" denotes the left eigenvectors of matrix A corresponding to the R largest singular values.

3.2 Implementation issues

(1) *Initialization*: One issue in the implementation is the initialization of the algorithm. In the original algorithm proposed by Lathauwer et al. (2000), the values of $U^{(n)} \in \mathbb{R}^{I_n \times R_n}$ were initialized with the truncation of the HOSVD. The columns of the column-wise orthogonal matrices span the space of the dominant left singular vectors of the matrix unfoldings $A_{(n)}$ $(1 \le n \le N)$. While the computation of HOSVD is very expensive ($\mathcal{O}(I_1 I_2 \cdots I_N)$), we use $U^{(n)}_0 = [I_{R_n}\ 0]^T$ or $U^{(n)}_0 =$ uniformly distributed random numbers (though columns are not necessarily orthonormal). Like

many iterative search methods, empirically we have not seen any major difference in the results obtained using these initializations (see Sect. 6.2). However, our initializations are much simpler to compute than HOSVD.

(2) *Efficient implementation* Most of the time spent by the algorithm is in the computation of eigenvectors using SVD. The time complexity of SVD on a $r \times c$ matrix is $\mathcal{O}(rc\min(r,c))$. Therefore, the total time cost is $\mathcal{O}(\sum_n I_n R^{N-1}\min\{I_n, R^{N-1}\})$, $(1 \le n \le N)$, which means it is efficient for low-rank approximation, i.e., when R is small. As R increases, the time and memory requirements increase fast, making the algorithm inefficient for large R. Because unfolded matrices are long and thin, as described in Observation 1, we utilize a simple strategy that can improve the efficiency. Suppose the SVD of A is USV^T. To avoid the large value of c, we exploit the fact that the eigensystem of AA^T is US^2U^T. Then we only need a subset of the column vectors from the left singular matrix, U, of A. Therefore, once we have obtained U from the eigensystem of AA^T, it is not necessary to compute V anymore.

3.3 Relationship with Existing Image-as-Matrix Approaches

For image ensembles (third-order tensors), the relationship of our tensor approximation algorithm with Datum-as-Is representation and GLRAM by Ye (2005) is described in the Theorem 2.

Theorem 2 *The tensor approximation algorithm using image-as-matrix representation in Algorithm* 1 *reduces to the GLRAM algorithm* (Ye 2005) *when the tensor is projected onto only the first two subspaces,* $U^{(1)}$ *and* $U^{(2)}$.

Proof Given a set of images, $A_1, A_2, \ldots, A_{I_3}$, with size of $I_1 \times I_2$ each, the GLRAM algorithm (Ye 2005) is aimed at finding transformation matrices $U^{(1)}$ and $U^{(2)}$ with orthonormal columns, and M_i which solve the minimization problem:

$$\min_{\substack{U^{(1)}\in\mathbb{R}^{I_1\times l_1}: U^{(1)^T}U^{(1)}=I_{l_1}\\ U^{(2)}\in\mathbb{R}^{I_2\times l_2}: U^{(2)^T}U^{(2)}=I_{l_2}\\ M_i\in\mathbb{R}^{l_1\times l_2}}} \sum_{i=1}^{I_3} \|A_i - U^{(1)}M_iU^{(2)^T}\|_F^2.$$

It is shown in (Ye 2005) that for a given $U^{(2)}$, $U^{(1)}$ consists of the l_1 eigenvectors of the matrix

$$M_L = \sum_{i=1}^{I_3} A_i U^{(2)}U^{(2)^T}A_i^T \tag{6}$$

corresponding to the largest l_1 eigenvalues, and for a given $U^{(1)}$, $U^{(2)}$ consists of the l_2 eigenvectors of the matrix

$$M_R = \sum_{i=1}^{I_3} A_i^T U^{(1)}U^{(1)^T}A_i \tag{7}$$

corresponding to the largest l_2 eigenvalues.

A third-order tensor, $\mathcal{A} \in \mathbb{R}^{I_1\times I_2\times I_3}$, is formulated for our tensor approximation algorithm using image-as-matrix representation. Its rank-R approximation is represented as $\mathcal{A} = \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)}$. According to the property of tensor product, i.e., $\mathcal{A} \times_m B \times_n C = \mathcal{A} \times_n C \times_m B$, where $B \in \mathbb{R}^{J_m\times I_m}$ and $C \in \mathbb{R}^{J_n\times I_n}$, we have

$$\mathcal{A} = (\mathcal{B} \times_3 U^{(3)}) \times_1 U^{(1)} \times_2 U^{(2)} = \mathcal{C} \times_1 U^{(1)} \times_2 U^{(2)} \tag{8}$$

where $\mathcal{C} = \mathcal{B} \times_3 U^{(3)}$. Our task is to find $U^{(1)}$ and $U^{(2)}$ such that the projection of the tensor $\mathcal{A}$ to the two subspaces has the minimum error of approximation in the least squares sense. This corresponds to finding the maximum Frobenius norm of the tensor $\mathcal{C}$, i.e., $\|\mathcal{C}\|^2 = \|\mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T}\|^2$. Again we apply the ALS for the optimization, i.e., we first fix $U^{(2)^T}$, and then solve for $U^{(1)}$ by solving the quadratic expression $f = \|\tilde{\mathcal{C}}^{(2)} \times_1 U^{(1)^T}\|^2$, where $\tilde{\mathcal{C}}^{(2)} = \mathcal{A} \times_2 U^{(2)^T}$. Using the matrix formulation, we obtain the unfolding matrix of $\tilde{\mathcal{C}}^{(2)}$ along the second mode as $\tilde{C}^{(2)} = U^{(2)^T} \cdot A_{(2)} = [U^{(2)^T} \cdot A_1^T U^{(2)^T} \cdot A_2^T \cdots U^{(2)^T} \cdot A_{I_3}^T]$ since $A_{(2)}$ can be represented as $[A_1^T A_2^T \cdots A_{I_3}^T] \in \mathbb{R}^{I_2\times I_1 I_3}$. $\tilde{C}^{(2)^T}$ is the column-permuted version of the unfolding matrix of $\tilde{\mathcal{C}}^{(2)}$ along the first mode. Since column permutation of a matrix does not change its left singular vectors, $U^{(1)}$ can then be found from $\tilde{C}^{(2)}$, i.e., $\tilde{C}^{(2)^T} = [A_1 \cdot U^{(2)} A_2 \cdot U^{(2)} \cdots A_{I_3} \cdot U^{(2)}]$. According to Algorithm 1, we have

$$\tilde{C}^{(2)^T} \cdot \tilde{C}^{(2)} = \sum_{k=1}^{I_3} A_k U^{(2)}U^{(2)^T}A_k^T. \tag{9}$$

Equation (9) is exactly equivalent to Equation (6). $U^{(1)}$ can be found as the dominant subspaces of $\tilde{C}^{(2)^T} \cdot \tilde{C}^{(2)}$.

Similarly we can obtain $U^{(2)}$ by solving the quadratic expression $f = \|\tilde{\mathcal{C}}^{(1)} \times_2 U^{2^T}\|^2$, where $\tilde{\mathcal{C}}^{(1)} = \mathcal{A} \times_1 U^{(1)^T}$. Using the matrix formulation, we obtain the unfolding matrix of $\tilde{\mathcal{C}}^{(1)}$ along the first mode as $\tilde{C}^{(1)} = U^{(1)^T} \cdot A_{(1)} = [U^{(1)^T} \cdot \mathcal{A}(:,1,:)U^{(1)^T} \cdot \mathcal{A}(:,2,:) \cdots U^{(1)^T} \cdot \mathcal{A}(:,I_2,:)]$ since $A_{(1)}$ can be represented as $[\mathcal{A}(:,1,:)\mathcal{A}(:,2,:) \cdots \mathcal{A}(:,I_2,:)] \in \mathbb{R}^{I_1\times I_2 I_3}$. $\tilde{C}^{(1)^T}$ is the column-permuted version of the unfolding matrix of $\tilde{\mathcal{C}}^{(1)}$ along the second mode. $U^{(2)}$ can then be found from $\tilde{C}^{(1)}$, i.e., $\tilde{C}^{(1)^T} = [A_1^T \cdot U^{(1)} A_2^T \cdot U^{(1)} \cdots A_{I_3}^T \cdot U^{(1)}]$. According to Algorithm 1, we have

$$\tilde{C}^{(1)^T} \cdot \tilde{C}^{(1)} = \sum_{k=1}^{I_3} A_k^T U^{(1)}U^{(1)^T}A_k. \tag{10}$$

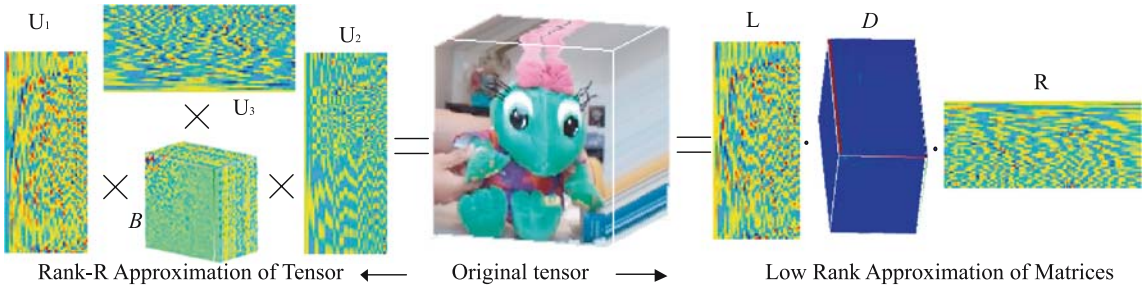

Fig. 3 Illustration of rank-R approximations of tensors vs. generalized low rank approximation of matrices (GLRAM). Dimension $R = 40$ and $d = 40$

Equation (10) is exactly equivalent to (7). $U^{(3)}$ can be found as the dominant subspaces of $\tilde{C}^{(1)^T} \cdot \tilde{C}^{(1)}$. □

The following corollary can be easily proved from Theorem 2.

Corollary 1 *Rank-one decomposition of matrices by* Shashua and Levin (2001) *is a special case of GLRAM with the same initialization in the sense that the best rank-one approximation of the matrices can be iteratively obtained by applying GLRAM. Rank-one tensor decomposition* (Wang and Ahuja 2004) *of* $\mathcal{A} \in \mathbb{R}^{I_1 \times I_2 \times \cdots \times I_N}$, *i.e.*, $\mathcal{A} = \sum_{i=1}^{r} \lambda_i u_i^{(1)} \circ u_i^{(2)} \circ \cdots \circ u_i^{(N)}$ (*where* λ_i *is a constant corresponding to the core tensor and* $u^{(j)}$, $(1 \le j \le N)$, *is a vector*), *is a special case of rank-R tensor approximation algorithm in the sense that we iteratively find the best rank-one tensor approximation of the original tensor.*

Proof The rank-one approximation of matrices in (Shashua and Levin 2001) is aimed at iteratively finding the best rank-one approximation, $A_k = \sum_{k=1}^{r} \lambda_{ij} u_j^{(1)} u_j^{(2)^T}$, of a collection of images (matrices), $A_1, A_2, \ldots, A_{I_3}$. The term $u^{(1)}$ is the eigenvector associated with the largest eigenvalue of $\hat{A}^T \hat{A}$, where $\hat{A} = [A_1 u^{(2)}, \ldots, A_{I_3} u^{(2)}]$, while $\hat{A}^T \hat{A}$ is equal to $\sum_{i=1}^{I_3} A_i u^{(2)} u^{(2)^T} A_i^T$. Similarly the term $u^{(2)}$ is the eigenvector associated with the largest eigenvalue of $\hat{A}\hat{A}^T$, where $\hat{A} = [A_1^T u^{(1)}, \ldots, A_{I_3}^T u^{(1)}]$, while $\hat{A}\hat{A}^T$ is equal to $\sum_{i=1}^{I_3} A_i^T u^{(1)} u^{(1)^T} A_i$. These exactly define the rank-one approximation using GLRAM.

It is easy to show that rank-R tensor decomposition reduces to rank-one decomposition of matrices (Shashua and Levin 2001) when the tensor is only projected onto the first two subspaces, $u^{(1)}$ and $u^{(2)}$ according to Theorem 2. □

Claim 1 *Rank-R tensor approximation algorithm using Datum-as-Is representation implicitly encodes the covariance between any pair of elements* (*pixels or voxels*) *in the image or volume space.*

Let us consider the aforementioned image ensemble, which has I_3 images with a resolution $I_1 \times I_2$. If we represent each image as a vector as in PCA, the entire image ensemble is represented as a matrix $\mathbf{B} \in \mathbb{R}^{(I_1 I_2) \times I_3}$. On the other hand, if we represent each image as a matrix, we need to factorize three unfolded matrices: $\mathbf{A}_{(1)} \in \mathbb{R}^{I_1 \times (I_2 I_3)}$, $\mathbf{A}_{(2)} \in \mathbb{R}^{I_2 \times (I_3 I_1)}$, and $\mathbf{A}_{(3)} \in \mathbb{R}^{I_3 \times (I_1 I_2)}$. Note that $\mathbf{A}_{(3)}$ is actually the transpose of $\mathbf{B}$. Suppose $\mathbf{U}_i$ is an eigenvector of the covariance matrix of $\mathbf{B}$. Then $\mathbf{A}_{(3)}\mathbf{U}_i$ is actually an eigenvector of the covariance matrix of $\mathbf{A}_{(3)}$. Because of this relationship between $\mathbf{A}_{(3)}$ and $\mathbf{B}$, our approach indirectly encodes the covariance between an arbitrary pair of pixels in the image plane. More importantly, we can show from Observation 2 that our scheme also encodes row and column covariances by factorizing $\mathbf{A}_{(1)}$ and $\mathbf{A}_{(2)}$.

Therefore, our algorithm is more general than that proposed by Ye (2005), since Ye considers the projection along only the temporal axis while our algorithm achieves reduction along both spatial and temporal axes. Figure 3 contrasts the different projection schemes used by these two approaches.

The relationship of the methods discussed above can be represented as in Fig. 4.

4 Tenor Approximation of Third-Order Tensors Using Slice Projection

Algorithm 1 is designed for approximating tensors of any order. In this section, we consider the important special case of image ensembles, which are third-order tensors. We present a specific algorithm for any rank approximation of third-order tensors, called slice projection.

Our approach is inspired by the work of Ye et al. (2004), Ye (2005). The basic idea of slice projection for rank-R approximation of tensors is similar to Algorithm 1 in that a tensor is transformed into matrices for the convenience of manipulation. In Algorithm 1, a tensor is unfolded along different coordinate axes to formulate matrices, while here a third-order tensor is represented as slices along the three

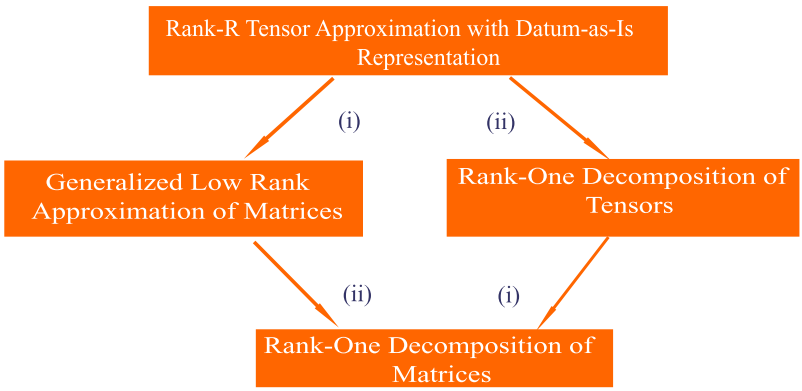

Fig. 4 Relationship among four methods: Tensor approximation of tensors with Datum-as-Is representation, generalized low rank approximation of matrices (GLRAM), rank-one decomposition of tensors (RODT) and rank-one decomposition of matrices (RODM). These methods are related through two operations: (i) partial projection and (ii) iterative rank one approximation

coordinate axes, A_i, A_j and A_k, where $1 \le i \le I_1$, $1 \le j \le I_2$ and $1 \le k \le I_3$. Each slice is represented by a matrix orthogonal to that direction. By projecting the slices along each direction to two corresponding coordinate axes under the rank constraints, we can find the best approximation of the original slices. We need to maximize the norm of the best approximation of original tensor, which corresponds to maximizing the summation of the norms of the slice projections in three directions. Then our problem is formulated as follows: given a tensor $\mathcal{A}$ (hence A_i, A_j and A_k), find $U^{(1)}, U^{(2)}$ and $U^{(3)}$ which solve

$$\max_{U^{(1)},U^{(2)},U^{(3)}} \sum_{i=1}^{I_1} \|U^{(2)^T} A_i U^{(3)}\|^2 + \sum_{j=1}^{I_2} \|U^{(1)^T} A_j U^{(3)}\|^2 + \sum_{k=1}^{I_3} \|U^{(1)^T} A_k U^{(2)}\|^2 \tag{11}$$

where $U^{(1)} \in \Re^{I_1 \times R_1}$, $U^{(2)} \in \Re^{I_2 \times R_2}$ and $U^{(3)} \in \Re^{I_3 \times R_3}$ have orthonormal columns; $1 \le i \le I_1$, $1 \le j \le I_2$, and $1 \le k \le I_3$. The following theorem describes how to find a locally optimal solution using an iterative procedure.

Theorem 3 (Slice-Projection Theorem) *Let $U^{(1)}, U^{(2)}$ and $U^{(3)}$ be the optimal solution to the maximization problem in (11). Then*

- *Given $U^{(1)}$ and $U^{(2)}$, $U^{(3)}$ consists of the R_3 eigenvectors of the matrix $M_{31} = \sum_{j=1}^{I_2} A_j^T U^{(1)} U^{(1)^T} A_j$ (and $M_{32} = \sum_{i=1}^{I_1} A_i^T U^{(2)} U^{(2)^T} A_i$) corresponding to the largest R_3 eigenvalues.*
- *Given $U^{(1)}$ and $U^{(3)}$, $U^{(2)}$ consists of the R_2 eigenvectors of the matrix $M_{23} = \sum_{i=1}^{I_1} A_i U^{(3)} U^{(3)^T} A_i^T$ (and $M_{21} = \sum_{k=1}^{I_3} A_k^T U^{(1)} U^{(1)^T} A_k$) corresponding to the largest R_2 eigenvalues.*
- *Given $U^{(2)}$ and $U^{(3)}$, $U^{(1)}$ consists of the R_1 eigenvectors of the matrix $M_{12} = \sum_{k=1}^{I_3} A_k U^{(2)} U^{(2)^T} A_k^T$ (and $M_{13} = \sum_{i=1}^{I_2} A_j U^{(3)} U^{(3)^T} A_j^T$) corresponding to the largest R_1 eigenvalues.*

Given $U^{(1)}$, $U^{(2)}$ and $U^{(3)}$, the projection of $\mathcal{A}$ onto the coordinate axes is represented as $\mathcal{B} = \mathcal{A} \times U^{(1)^T} \times U^{(2)^T} \times U^{(3)^T}$.

Proof Given U_1 and U_2, U_3 maximizes

$$\max_{U^{(3)}} \sum_{j=1}^{I_2} \|U^{(1)^T} A_j U^{(3)}\|^2 \text{ and } \sum_{i=1}^{I_1} \|U^{(2)^T} A_i U^{(3)}\|^2. \tag{12}$$

The first term in (12) can be rewritten as

$$\sum_{j=1}^{I_2} \operatorname{trace}(U^{(3)^T} A_j^T U^{(1)} U^{(1)^T} A_j U^{(3)})$$

$$= \operatorname{trace}\left(U^{(3)^T} \left(\sum_{j=1}^{I_2} A_j^T U^{(1)} U^{(1)^T} A_j \right) U^{(3)} \right)$$

$$= \operatorname{trace}(U^{(3)^T} M_{31} U^{(3)})$$

which is maximal for a given $U^{(1)}$ only if $U^{(3)} \in \Re^{I_3 \times R_3}$ consists of the R_3 eigenvectors of the matrix M_{31} corresponding to the largest R_3 eigenvalues. From the second term in (12), we obtain $U^{(3)}$ which maximizes $\operatorname{trace}(U^{(3)^T} M_{32} U^{(3)})$. In either case, $U^{(3)}$ is locally optimal for the maximization of (12). Similarly, we can show other parts of the theorem. □

This theorem provides us with an iterative procedure to find $U^{(1)}, U^{(2)}$, and $U^{(3)}$. By updating $U^{(1)}, U^{(2)}$, and $U^{(3)}$ iteratively, the procedure will converge to a (local) maximum of (11). This is described in Algorithm 2. The advantage of Algorithm 2 is that it is time and memory efficient for any rank approximation of third-order tensors since the cost for finding the eigenvectors is only $\mathcal{O}(I_1^3 + I_2^3 + I_3^3)$, but the tradeoff is that this algorithm only works for third-order tensors.

Table 2 Comparison of data representation using different methods

Method	Image as	Formulation	Number of scalars
PCA (p)	vector	$\tilde{A}_i = \Phi \cdot P_i + \overline{A}$	$p(I_1 \cdot I_2 + I_3)$
Rank-one (r)	matrix	$\tilde{\mathcal{A}} = \sum_{i=1}^{r} \lambda_i \cdot U_i^{(1)} \circ U_i^{(2)} \circ U_i^{(3)}$	$r(I_1 + I_2 + I_3 + 1)$
GLRAM (d)	matrix	$\tilde{A}_i = L \cdot D_i \cdot R^T$	$d(I_1 + I_2 + I_3 d)$
Rank-R (R)	matrix	$\tilde{\mathcal{A}} = \mathcal{B} \times_1 U^{(1)} \times_2 U^{(2)} \times_3 U^{(3)}$	$R(R^2 + I_1 + I_2 + I_3)$

Algorithm 2: Rank-R approximation of third-order tensors

Data: Given a third-order tensor, $\mathcal{A}$, and R

Result: Find U^k, $(1 \le k \le 3)$ and $\mathcal{B}$.

Initialize $U_0^{(k)} \in \Re^{I_k \times R_k}$, $1 \le k \le 3$;

while $\sim$ *stop* **do**

$M_{21} = \sum_{k=1}^{I_3} A_k^T U_j^{(1)} U_j^{(1)^T} A_k$;
$U_{j+1}^{(2)} = svds(M_{21}, R_2)$;

$M_{32} = \sum_{i=1}^{I_1} A_i^T U_{j+1}^{(2)} U_{j+1}^{(2)^T} A_i$;
$U_{j+1}^{(3)} = svds(M_{32}, R_3)$;

$M_{13} = \sum_{i=1}^{I_2} A_j U_{j+1}^{(3)} U_{j+1}^{(3)^T} A_j^T$;
$U_{j+1}^{(1)} = svds(M_{13}, R_1)$;

$\mathcal{B} = \mathcal{A} \times_1 U_{j+1}^{(1)^T} \times_2 U_{j+1}^{(2)^T} \times_3 U_{j+1}^{(3)^T}$;
if $(\|\mathcal{B}_{j+1}\|^2 - \|\mathcal{B}_j\|^2 < \varepsilon)$ *stop*

end

5 Dimensionality Reduction and Feature Extraction

5.1 Dimensionality Reduction

Representation of the data using PCA consists of p eigenvectors, $\Phi \in \mathbb{R}^{I_1 I_2 \times p}$ and reduced representations $P \in \mathbb{R}^{I_3 \times p}$. Tensor rank-one decomposition (Wang and Ahuja 2004) consists of the projection λ corresponding to each rank-one tensor $U^{(1)} \circ U^{(2)} \circ \cdots \circ U^{(N)}$. GLRAM (Ye 2005) projects each image using two matrices $L \in \mathbb{R}^{I_1 \times d}$ and $R \in \mathbb{R}^{I_2 \times d}$, and the projection is $D \in \mathbb{R}^{d \times d}$. For rank-R approximation, a tensor is approximated by a core tensor $\mathcal{B} \in \mathbb{R}^{R \times R \times R}$ and three subspaces $U^{(1)} \in \mathbb{R}^{I_1 \times R}$, $U^{(2)} \in \mathbb{R}^{I_2 \times R}$ and $U^{(3) \in \mathbb{R}^{I_3 \times R}}$. A comparison of these methods is given in Table 2, where $\overline{A}$ is the mean of the data.

In image ensemble applications, the number of images is usually much greater than the dimension d or R for dimensionality reduction, i.e., $I_3 \gg d$ and $I_3 \gg R$. Therefore, $(I_1 + I_2 + I_3 d)d > (R^2 + I_1 + I_2 + I_3)R$ if we assume $R = d$, i.e., the representation of original data using rank-R approximation is more compact than that using GLRAM. When using image-as-vector representation, the rank-R approximation of a tensor reduces to SVD of a matrix.

5.2 Feature Extraction

Rank-R approximation of tensors can be used to extract features of the image ensemble. By projecting the original tensor data onto $R_1, R_2, \ldots, R_N$ axis system, we obtain a new tensor. By projecting it to any combination of two axes system, we define a feature of the slice along the plane defined by the two axes. The projections of a third-order tensor on any two axes are defined as $\mathcal{B}_{xy} = \mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T}$, $\mathcal{B}_{yz} = \mathcal{A} \times_2 U^{(2)^T} \times_3 U^{(3)^T}$ and $\mathcal{B}_{xz} = \mathcal{A} \times_1 U^{(1)^T} \times_3 U^{(3)^T}$. Given a test image, which can be represented as a tensor $\mathcal{A}$ of size $I_1 \times I_2 \times 1$, the matrix B

$$B = \mathcal{A} \times_1 U^{(1)^T} \times_2 U^{(2)^T} \tag{13}$$

can be used as a feature of the image. The feature extracted using Rank-R approximation of tensors using image-as-matrix representation is a matrix while that using SVD/PCA is a vector.

6 Experimental Results

In this section, we experimentally evaluate the performance of our proposed algorithm with respect to the quality of representation, computational complexity, and efficacy in object classification. The datasets we use include:

- Video dataset: It is a 129-frame video sequence of a moving toy. The size of each frame is 129×129. So we build a third-order tensor of $129 \times 129 \times 129$.
- Olivetti Research Laboratory (ORL) dataset:[1] It contains 10 different images of each of 40 distinct persons. The images were take at different times with different illumination and varying facial expressions. The size of each image is 92×112. Therefore the dimensionality of the dataset is $92 \times 112 \times 400$.
- Yale face dataset:[2] It is a widely used face database for researchers in face recognition. It contains 165 images of 15 persons. Each person has 11 images with different illumination, varying facial expressions and facial details. The images are centered, aligned using eyes positions,

[1] http://www.uk.research.att.com/facedatabase.html.

[2] http://cvc.yale.edu/projects/yalefaces/yalefaces.html.

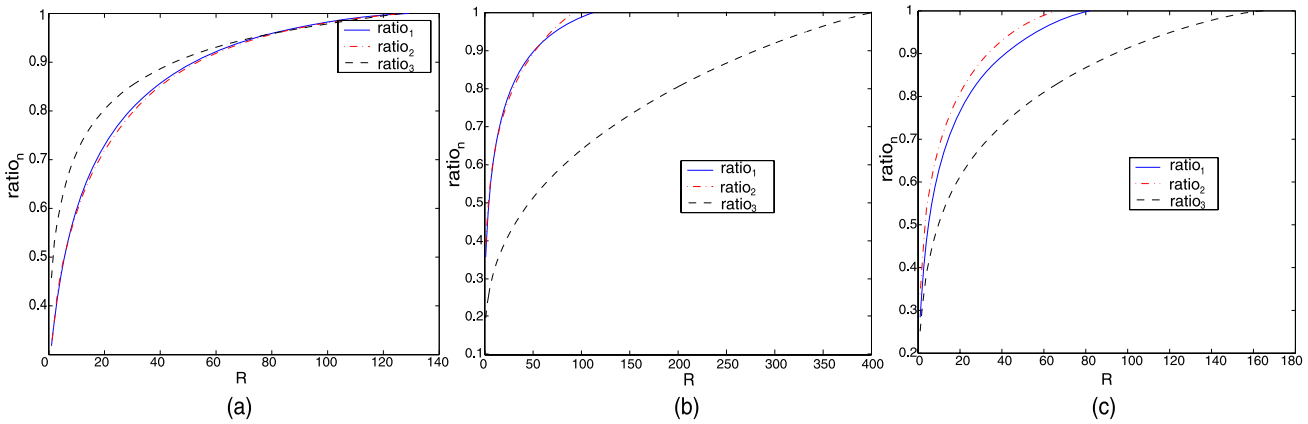

Fig. 5 The plot of ratio$_n$ vs. R for **a** video sequence, **b** ORL, **c** YALE

Table 3 Sensitivity to initializations for the video sequence

Initialization	Iterations	Convergence RMSE	Computation time (s)
HOSVD	22	3563.000781	87.4150
$[\, I_{R_n}\ 0\,]^T$	15	3563.236328	25.8980
rand1	16	3563.235387	30.2940
rand2	15	3563.235388	29.7430
rand3	14	3563.235387	26.5880
rand4	16	3563.235388	30.3430
rand5	15	3563.235387	29.9540

and cropped to the size of 82×66. The eye positions are provided by Ben Axelrod (http://www.facedetection.com/downloads/YaleData.txt). The dataset has the size of $82 \times 66 \times 165$.

Our experiments were performed using Matlab on a Pentium IV 1.5 GHz machine with 512 MB RAM.

6.1 Determination of R

As in SVD, the singular values can be used to evaluate how much variation the approximation of data can capture. We employ a similar scheme by using the Frobenius-norms $\|\mathcal{B}_n\|$ as "singular values" of a tensor along the nth mode. R_n can be determined by the ratio

$$\text{ratio}_n = \frac{\sum_{j=1}^{R_n} \mathcal{B}_{i_n=j}}{\sum_{j=1}^{I_n} \mathcal{B}_{i_n=j}}.$$

According to the ordering property in Theorem 1, i.e., $\|\mathcal{B}_{i_n=1}\| \geq \|\mathcal{B}_{i_n=2}\| \geq \cdots \geq \|\mathcal{B}_{i_n=I_n}\| \geq 0$ for all possible values of n, ratio$_n$ captures the largest variation along the nth mode.

Figure 5 is the ratio plot for three datasets. We can see that ratio$_1$ is very close to ratio$_2$. This illustrates that the rows and columns of each dataset have similar variations.

6.2 Sensitivity to Initialization

In this section, we evaluate the sensitivity of the rank-R tensor approximation algorithm to three different initialization schemes: (1) HOSVD, (2) $U_0^{(n)} = [\, I_{R_n}\ 0\,]^T$, and (3) $U_0^{(n)} =$ uniformly distributed random numbers. We compared the number of iterations for convergence and computational time using different initialization schemes using different datasets.

We used $R = 10$ for all three datasets with the specified threshold ($\varepsilon = 10^{-6}$). The results are given in Tables 3, 4, and 5 for three datasets. Generally speaking, using HOSVD usually obtains a little bit better locally optimal solution than the other two schemes, but it is much more expensive computationally. However, we can see from the tables that the difference is not significant among the convergence RMSEs obtained using different initialization schemes. From our experience with extensive experiments using different image/video datasets, simple initialization using scheme (2) or (3) can practically obtain good results for image processing and computer vision applications.

6.3 Compact Data Representation

We applied PCA, GLRAM, rank-one decomposition, and rank-R approximation of tensors methods on different

Table 4 Sensitivity to initializations for the ORL dataset

Initialization	Iterations	Convergence RMSE	Computation time (s)
HOSVD	6	2590.507936	248.8580
$[\, I_{R_n}\ 0\,]^T$	9	2590.508057	30.0430
rand1	11	2590.507936	39.3470
rand2	9	2590.507936	32.3960
rand3	9	2590.507936	32.1460
rand4	10	2590.507936	36.0310
rand5	11	2590.507936	39.2770

Table 5 Sensitivity to initializations for the YALE dataset

Initialization	Iterations	Convergence RMSE	Computation time (s)
HOSVD	8	1531.010022	23.6130
$[\, I_{R_n}\ 0\,]^T$	13	1531.009766	11.1960
rand1	11	1531.010022	10.3550
rand2	10	1531.010022	9.3730
rand3	11	1531.010022	10.2040
rand4	11	1531.010022	10.2940
rand5	10	1531.010022	9.4140

datasets. For third-order data, HOSVD using image-as-vector representation is equivalent to PCA. It is also shown in (Wang et al. 2005; Vasilescu and Terzopoulos 2003) that for higher-order data, the reconstruction error is larger for higher-order decomposition of tensors using image-as-vector representation than PCA. We focus on third-order tensors in this paper, though the algorithm can be generalized to higher-order data.

The parameters we used to achieve a constant compression ratio for different algorithms are given in Table 2. The compression ratio is defined as $\alpha = \frac{I_1 \times I_2 \times I_3}{s}$, where s is the number of scalars required for representation of the data (see Table 2). Though we can utilize the orthonormality property of columns of basis matrices to further reduce the storage for an $n \times r$ orthonormal matrix, the bookkeeping to achieve is tedious, and for $n \gg r$ the difference is small. We therefore does not consider this aspect here. We compute the RMSE error of the reconstructed data with respect to the original data,

$$\text{RMSE} = \sqrt{\frac{1}{I_3}\|\mathcal{A} - \tilde{\mathcal{A}}\|^2}$$

as a measure of relative performance of the algorithms.

Figure 6(a)–(c) shows the reconstructions obtained using the same compression ratio but different representation methods. For the toy sequence (Fig. 6(a)), the temporal redundancy is very strong since the scene does not change much from frame to frame, though spatial redundancies also exist. Interestingly, the reconstruction using GLRAM is visually even worse than that using PCA. The reconstruction using GLRAM is very blurry, and has some features (e.g., eyes) missing (Fig. 6(a2)). This is because GLRAM captures more redundancies in the spatial than in the temporal dimension, while PCA captures mostly the temporal redundancies in this case. The reconstructions are much better for tensor rank-one decomposition and rank-R approximation methods since both methods capture the redundancies in all dimensions. Moreover, since the basis columns of rank-R approximation are orthonormal (therefore more compact), rank-R approximation of tensors yields much better reconstruction than tensor rank-one decomposition. For the face datasets (Fig. 6(b), (c)), PCA is the worst among all methods since the spatial redundancies are not well captured due to its image-as-vector formulation. The critical features like eyes, mouth and nose begin to appear in Fig. 6(b2), (b3), and become pretty clear in Fig. 6(b4). For the compression ratio $\alpha = 77 : 1$ (corresponding to using five principal components for PCA), the reconstruction using our algorithm (Fig. 6(c4)) is visually quite close to the original image (Fig. 6(e)). Figure 6(d)–(f) shows the reconstruction error for each dataset. These plots are consistent with the relative visual quality of the reconstructions. Our algorithm produces the best visual reconstructions as well as the smallest RMSE of all methods. As the compression ratio decreases, all four methods give similar performance. This is reasonable since the use of increasing number of components leads to steadily decreasing amount of information loss.

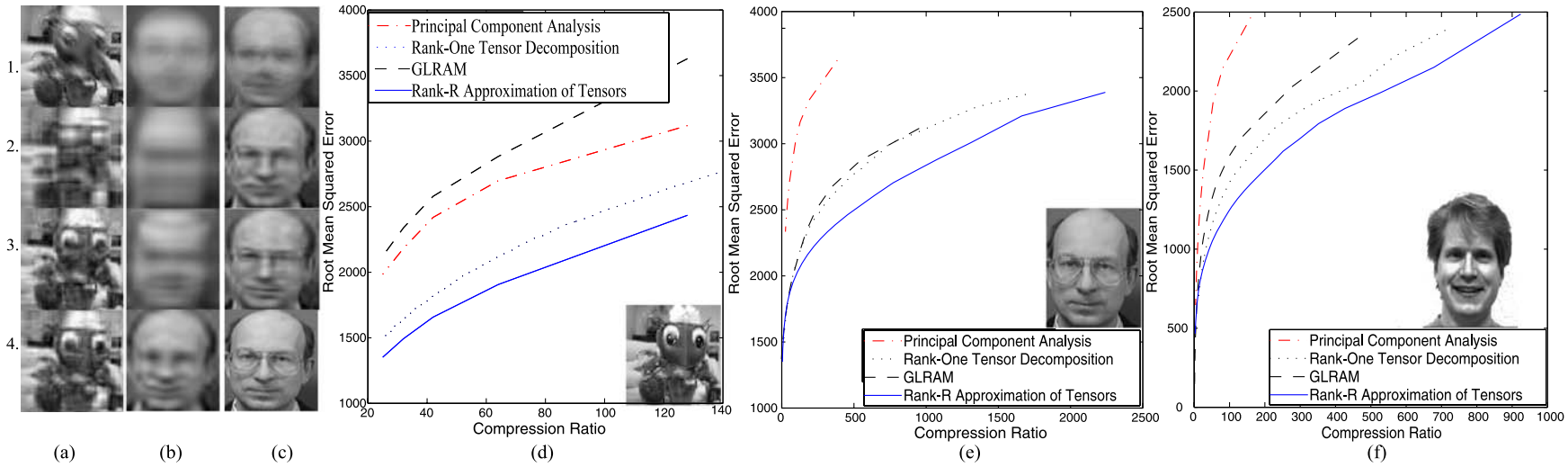

Fig. 6 Comparison of the quality of reconstructions achieved using rank-R approach and other methods. The parameters used for each method were chosen so as to achieve the same compression ratio. **a** Reconstruction results for the toy sequence ($p = 1$, $d = 10$, $r = 43$, $R = 20$, $\alpha = 128 : 1$). Original image is in (**d**). **b** Reconstruction results for ORL ($p = 1$, $d = 5$, $r = 18$, $R = 14$, $\alpha = 385 : 1$). **c** Reconstruction results for ORL ($p = 5$, $d = 11$, $r = 88$, $R = 32$, $\alpha = 77 : 1$). Original image is in (**e**). Reconstruction error for **d** toy sequence, **e** ORL, **f** Yale

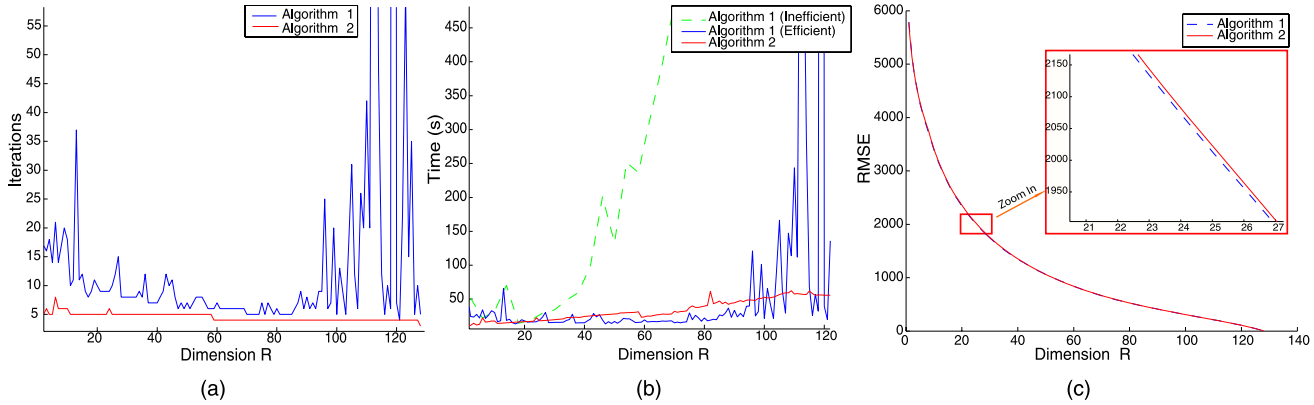

Fig. 7 Comparison of Algorithm 1 and Algorithm 2. **a** Number of iterations for convergence. **b** Execution time. **c** Reconstruction error

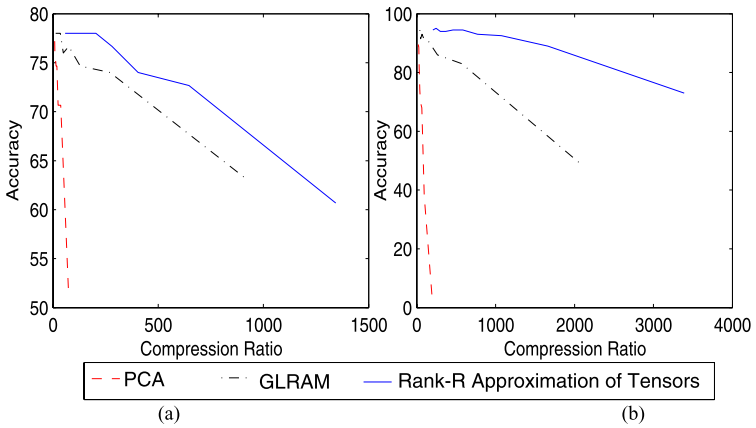

Fig. 8 Comparing the quality of representation of the different methods on face recognition for the Yale and ORL face datasets. **a** Accuracy vs. compression ratio on Yale. **b** Accuracy vs. compression ratio on ORL

6.4 Computational Efficiency

We compared Algorithm 1 and Algorithm 2 in terms of number of iterations for convergence (Fig. 7(a)), execution time (Fig. 7(b)) and RMSE error (Fig. 7(c)) for the video sequence dataset. The threshold we used is $\varepsilon = 10^{-4}$. Figure 7(a) shows that Algorithm 1 usually takes more iterations to converge to a locally optimal solution than Algorithm 2, which often costs more time as shown in Fig. 7(b). For Algorithm 1, we also compared the execution time for both efficient and inefficient cases. Algorithm 1 with inefficient implementation is much slower than efficient implementation as R increases due to large memory requirements. The RMSE representation errors are comparable for both Algorithms as shown in Fig. 7(c).

6.5 Appearance-Based Recognition

To evaluate the quality of the representation obtained by our algorithm, we applied it to appearance-based recognition in face images in ORL and Yale face databases. For the Yale database (containing 11 different images of each of 15 distinct persons), we used 11-fold cross-validation; i.e., we randomly selected 10 images per person, and used the remaining 1 for testing. For the ORL face database (containing 10 different images of each of 40 distinct persons), we applied 10-fold cross-validation. Similarly, we randomly selected nine images from each class as a training set, and used the remaining image for testing. For each database, we repeated the process 10 times. Features can be obtained using (13). The face is correctly classified if its feature has the minimum Frobenius distance from the features of the same person. The reported final accuracy is the average of the 10 runs.

We compared our algorithm with GLRAM and PCA. As shown in Fig. 8, comparing the performance of different methods for a fixed compression ratio, our method yields the highest accuracy in every case (Fig. 8). This shows that our algorithm has the best reduced dimensionality representation.

7 Conclusion

We have introduced a new approach to dimensionality reduction based on multilinear algebra using Datum-as-Is representation. The method is designed to capture the spatial and temporal redundancies along each dimension of the

tensor. Experiments show superior performance of rank-R approximation of tensors in terms of quality of data representation and object classification accuracy for a fixed compression ratio. In the future, we will consider integrating other spatial bases, for example the Fourier basis, to encode spatial redundancy in general images.

Acknowledgements The support of the Office of Naval Research under grant N00014-03-1-0107 is gratefully acknowledged. We would like to thank the anonymous reviewers for their constructive comments. Thanks go as well to Qing Qu and Yizhou Yu for the helpful discussion that strengthened this paper.

References

Barlow, H. (1989). Unsupervised learning. *Neural Computation*, *1*, 295–311.

Comon, P. (1994). Independent component analysis, a new concept? *Signal Processing*, *36*(3), 287–314.

Hong, W., Wright, J., Huang, K., & Ma, Y. (2005). A multi-scale hybrid linear model for lossy image representation. In *ICCV '05: Proceedings of the tenth IEEE international conference on computer vision* (Vol. 1, pp. 764–771).

Jutten, C., & Herault, J. (1991). Blind separation of sources, part 1: An adaptive algorithm based on neuromimetic architecture. *Signal Processing*, *24*(1), 1–10.

Kofidis, E., & Regalia, P. A. (2002). On the best rank-1 approximation of higher-order supersymmetric tensors. *SIAM Journal on Matrix Analysis and Applications*, *23*(3), 863–884.

Kroonenberg, P. (1983). *Three-mode principal component analysis*. Leiden: DSWO.

Lathauwer, L., Moor, B. D., & Vandewalle, J. (2000). On the best rank-1 and rank-(R_1, R_2, ..., R_N) approximation of high-order tensors. *SIAM Journal on Matrix Analysis and Applications*, *21*(4), 1324–1342.

Lathauwer, L., Moor, B. D., & Vandewalle, J. (2000). A multilinear singular value decomposition. *SIAM Journal on Matrix Analysis and Applications*, *21*(4), 1253–1278.

Olshausen, B., & Field, D. (1996). Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, *381*(13), 607–609.

Shashua, A., & Levin, A. (2001). Linear image coding for regression and classification using the tensor-rank principle. In *CVPR'01: Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition* (p. 42).

Sirovich, L., & Kirby, M. (1987). Low-dimensional procedure for the characterization of human faces. *Journal of Optical Society of America*, *4*(3), 519–524.

Tenenbaum, J. B., & Freeman, W. T. (2000). Separating style and content with bilinear models. *Neural Computation*, *12*(6), 1247–1283.

Turk, M., & Pentland, A. (1991). Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, *3*(1), 71–86.

Vasilescu, M. A. O., & Terzopoulos, D. (2002). Multilinear analysis of image ensembles: tensorfaces. In *ECCV '02: Proceedings of the 7th European conference on computer vision—part I* (pp. 447–460).

Vasilescu, M. A. O., & Terzopoulos, D. (2003). Multilinear subspace analysis of image ensembles. In *CVPR '03: Proceedings of the 2005 IEEE computer society conference on computer vision and pattern recognition* (pp. 93–99).

Wang, H., & Ahuja, N. (2003). Facial expression decomposition. In *ICCV '03: Proceedings of the ninth IEEE international conference on computer vision* (p. 958).

Wang, H., & Ahuja, N. (2004). Compact representation of multidimensional data using tensor rank-one decomposition. In *ICPR '04: Proceedings of the 17th international conference on pattern recognition* (Vol. 1, pp. 44–47).

Wang, H., & Ahuja, N. (2005). Rank-R approximation of tensors: using image-as-matrix representation. In *CVPR '05: Proceedings of the 2005 IEEE computer society conference on computer vision and pattern recognition* (Vol. 2, pp. 346–353).

Wang, H., Wu, Q., Shi, L., Yu, Y., & Ahuja, N. (2005). Out-of-core tensor approximation of multi-dimensional matrices of visual data. *ACM Transactions on Graphics*, *24*(3), 527–535.

Xu, D., Yan, S., Zhang, L., Zhang, H.-J., Liu, Z., & Shum, H.-Y. (2005). Concurrent subspaces analysis. In *IEEE computer society conference on computer vision and pattern recognition (CVPR'05)* (Vol. 2, pp. 203–208).

Yang, J., Zhang, D., Frangi, A. F., & Yang, J. Y. (2004). Two-dimensional PCA: a new approach to appearance-based face representation and recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *26*(1), 131–137.

Ye, J. (2005). Generalized low rank approximations of matrices. *Machine Learning*, *61*, 167–191.

Ye, J., Janardan, R., & Li, Q. (2004). GPCA: an efficient dimension reduction scheme for image compression and retrieval. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 354–363), New York, NY, USA.