Pattern Anal Applic (2009) 12:391–406
DOI 10.1007/s10044-008-0135-9

THEORETICAL ADVANCES

Computationally efficient eigenspace decomposition of correlated images characterized by three parameters

K. Saitwal · A. A. Maciejewski · R. G. Roberts

Received: 17 October 2007 / Accepted: 2 April 2008 / Published online: 12 August 2008

Abstract Eigendecomposition is a common technique that is performed on sets of correlated images in a number of pattern recognition applications including object detection and pose estimation. However, many fast eigendecomposition algorithms rely on correlated images that are, at least implicitly, characterized by only one parameter, frequently time, for their computational efficacy. In some applications, e.g., three-dimensional pose estimation, images are correlated along multiple parameters and no natural one-dimensional ordering exists. In this work, a fast eigendecomposition algorithm that exploits the "temporal" correlation within image data sets characterized by one parameter is extended to improve the computational efficiency of computing the eigendecomposition for image data sets characterized by three parameters. The algorithm is implemented and evaluated using three-dimensional pose estimation as an example application. Its accuracy and computational efficiency are compared to that of the original algorithm applied to one-dimensional pose estimation.

K. Saitwal
Behavioral Recognition Systems, Inc., 2100 West Loop South, 9th Floor, Houston, TX 77027, USA

A. A. Maciejewski (✉)
Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373, USA
e-mail: aam@colostate.edu

R. G. Roberts
Department of Electrical and Computer Engineering, Florida A & M—Florida State University, Tallahassee, FL 32310-6046, USA

Keywords Eigenspace · Singular value decomposition · Computational complexity · Image sequences · Three-dimensional correlations · Computer vision

1 Originality and contribution

Purely appearance-based techniques such as singular value decomposition (SVD) have been extensively used in many computer vision applications, viz., face characterization, object recognition, pose estimation, visual tracking, and inspection. Unfortunately, the offline calculation of the SVD of correlated images of three-dimensional (3D) objects can be prohibitively expensive and this fundamental problem is addressed here. Numerous computationally efficient SVD algorithms have been proposed, which exploit the correlation between the images along one dimension, typically using the natural order implied by time. However, there is no natural order for images correlated along three dimensions, as is the case with three-dimensional pose detection. In addition, 3D applications exacerbate the computational and storage issues. This paper characterizes such image data sets by three parameters and exploits their 3D frequency properties to propose an ordering of the frequency harmonics in terms of their energy recovery ability for a given 3D image data set. This frequency analysis is used to extend one of the fastest known "one-dimensional" SVD algorithms to improve the performance of computing the eigendecomposition of general 3D image data sets. The novelty of this algorithm lies in the computationally efficient use of 3D correlations of general 3D image data sets for computing their partial SVDs. The empirical results show that the proposed algorithm gives nearly an order of magnitude increase in performance for a given user-specified measure of accuracy.

2 Introduction

Eigendecomposition-based techniques play an important role in numerous image processing and computer vision applications. The advantage of these techniques, also referred to as subspace methods, is that they are purely appearance based and require few online computations. Variously referred to as eigenspace methods, singular value decomposition (SVD) methods, principal component analysis methods, and Karhunun–Loeve transformation methods [1, 2], they have been used extensively in a variety of applications such as face characterization [3, 4] and recognition [5–9], lip-reading [10, 11], object recognition [12–15], pose detection [16, 17], visual tracking [18, 19] and inspection [20–23]. All of these applications take advantage of the fact that a set of highly correlated images can be approximately represented by a small set of eigenimages [24–32]. Once the set of principal eigenimages is determined, online computation using these eigenimages can be performed very efficiently. However, the offline calculation required to determine both the appropriate number of eigenimages as well as the eigenimages themselves can be prohibitively expensive.

Many computationally efficient eigendecomposition algorithms have been proposed in recent years including the SVD power method [24, 25], Lanczos algorithm [28], conjugate-gradient algorithms [26, 27], recursive/adaptive eigenspace update techniques [29, 30], and frequency-domain techniques [31]. Recently, Chang et al. [32] proposed a fundamentally different algorithm by reducing the "temporal" resolution of image data sets, while Saitwal et al. [33] improved Chang's algorithm further by effectively using spatial similarities of original images at low resolutions. Note that all these algorithms considered exploiting the correlation between the images of a video sequence using the implicit or natural order imposed by time. [These image data sets will be referred to as one-dimensional (1D) image data sets here.] However, certain pattern recognition applications require that different views of an object taken from different three-dimensional (3D) spatial camera locations be considered in the image data set whose eigendecomposition is desired. Because they are characterized by three parameters, such image data sets will be referred to as 3D image data sets. The existing eigendecomposition algorithms cannot take advantage of the correlation in such 3D image data sets directly, as they do not consider the correlation along more than one dimension. The goal of this paper is to extend Chang's eigendecomposition algorithm [32], to efficiently compute the partial SVD of such 3D image data sets.

The remainder of this paper is organized as follows. Section 2 provides a review of the fundamentals of applying eigendecomposition to related images. This section also defines comparison criteria to quantify the difference between two eigendecompositions. Section 3 gives an overview of Chang's algorithm [32] and points out that it can only work with 1D image data sets. Section 4 explains the generation and frequency analysis of fully general 3D image data sets. This section also explains how to efficiently compute and use the real 3D discrete Fourier transform (DFT) of these sets. Section 5 addresses the issue of ordering the frequencies for these 3D image data sets. The frequency analysis along with the proposed ordering of 3D frequencies for the given 3D image data sets, outlined in Sect. 6, is used to extend Chang's algorithm to quickly compute the desired portion of the eigendecomposition of 3D image data sets based on a user-specified measure of accuracy. In Sect. 7, the performance of the proposed algorithm is evaluated on different 3D image data sets. Finally, some concluding remarks are given in Sect. 8.

3 Preliminaries

3.1 Singular value decomposition of correlated images

In this work, a grey-scale image is an $h \times v$ array of square pixels with intensity values normalized between 0 and 1. Thus, an image will be represented by a matrix $\mathcal{X} \in [0,1]^{h \times v}$. Because sets of related images are considered here, the *image vector* $\mathbf{x}$ of length $m = h \times v$ can be obtained by "row-scanning" an image into a column vector, i.e., $\mathbf{x} = \mathrm{vec}(\mathcal{X}^T)$. The *image data matrix* of a set of images $\mathcal{X}_1, \ldots, \mathcal{X}_n$ is an $m \times n$ matrix, denoted X, and defined as $X = [\mathbf{x}_1, ..., \mathbf{x}_n]$, where typically $m \gg n$. The case with fixed n is considered in this study, as opposed to cases where X is constantly updated with new images.

The SVD of X is given by

$$X = U \Sigma V^T, \tag{1}$$

where $U \in \Re^{m \times m}$ and $V \in \Re^{n \times n}$ are orthogonal matrices, and $\Sigma = [\Sigma_d \ \mathbf{0}]^T \in \Re^{m \times n}$ where $\Sigma_d = \mathrm{diag}\,(\sigma_1, ..., \sigma_n)$ is the matrix of singular values with $\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_n \geq 0$ and $\mathbf{0}$ is an n by $m-n$ zero matrix. The SVD of X plays a central role in several important imaging applications such as image compression and pattern recognition. The columns of U, denoted $\hat{\mathbf{u}}_i, i = 1, \ldots, m$, are referred to as the left singular vectors or eigenimages of X, while the columns of V, denoted $\hat{\mathbf{v}}_i, i = 1, \ldots, n$, are referred to as the right singular vectors of X. The corresponding singular values measure how "aligned" the columns of X are with the associated eigenimage.

In practice, the singular values and the corresponding singular vectors are not known or computed exactly, and instead their estimates are used. Hence it is important to define appropriate comparison criteria that can measure the

errors between the true and approximated eigenspaces. The next subsection defines three such error measures that are relevant to a user's motivation for performing an eigendecomposition.

3.2 Difference measures for SVD

The simplest error measure considered in this work is the difference between the true and the approximated singular values. Note that the *i*th approximated singular vector may not be aligned with the *i*th true singular vector even though the subspaces containing the first k vectors may span the same vector space. Hence two more error measures are defined in this section that will compare the subspaces consisting of the singular vectors rather than the individual vectors.

3.2.1 Energy recovery ratio

True and approximated eigenimages of X can be compared in terms of their capability of recovering the amount of the total energy in X. This "energy recovery ratio" for an orthonormal set of approximate eigenimages can be given by

$$\rho(X, \tilde{\hat{\mathbf{u}}}_1, \tilde{\hat{\mathbf{u}}}_2, \ldots, \tilde{\hat{\mathbf{u}}}_k) = \frac{\sum_{i=1}^{k} \|\tilde{\hat{\mathbf{u}}}_i^T X\|^2}{\|X\|_F^2} \leq 1, \tag{2}$$

where $\|\cdot\|_F$ represents the Frobenius norm and $\tilde{\hat{\mathbf{u}}}_i$ is the *i*th approximated eigenimage. The true eigenimages $\{\hat{\mathbf{u}}_1, \hat{\mathbf{u}}_2, \ldots, \hat{\mathbf{u}}_k\}$ yield the maximum energy recovery ratio for a given k.

3.2.2 Subspace criterion

True eigenimages give an optimum energy recovery ratio in (2). Hence, it is possible that more approximated eigenimages are required than the true ones to achieve the same energy recovery ratio. Thus another measure used in this study is the degree to which approximate eigenimages span the subspace of the first k^* true eigenimages, which will be referred to as the subspace criterion, s, given by

$$s = \sqrt{\frac{1}{k^*} \sum_{i=1}^{k} \sum_{j=1}^{k^*} (\tilde{\hat{\mathbf{u}}}_i \cdot \hat{\mathbf{u}}_j)^2}. \tag{3}$$

Consider $U_{k^*} = [\hat{\mathbf{u}}_1, \hat{\mathbf{u}}_2, \ldots, \hat{\mathbf{u}}_{k^*}]$ and $\tilde{U}_k = [\tilde{\hat{\mathbf{u}}}_1, \tilde{\hat{\mathbf{u}}}_2, \ldots, \tilde{\hat{\mathbf{u}}}_k]$. If the column space of U_{k^*} is included in that of $\tilde{U}_k$, then $\|\tilde{U}_k^T \hat{\mathbf{u}}_j\| = 1$ for $j = 1, 2, ..., k$. Hence, if the entire subspace of U_{k^*} is spanned by $\tilde{U}_k$, then $s = 1$, otherwise $s < 1$.

The above two error measures provide slightly different information regarding the "quality" of the estimated eigenimages. The energy recovery ratio, ρ, implicitly includes the effect of the singular values and thus weights the estimated eigenimages differently based on their importance. In contrast, the subspace criterion, s, is purely a subspace measure.

4 Overview of Chang's algorithm

One of the fastest known "one-dimensional" algorithms for computing the first k approximated eigenimages of correlated images to a user-specified accuracy is proposed by Chang et al. [32]. This section gives an overview of that algorithm, along with its computational efficiency. For this purpose, consider X where each $\mathbf{x}_{i+1}$ is obtained from $\mathbf{x}_i$ by a planar rotation of $\theta = 2\,\pi/n$. The correlation matrix X^TX is given by

$$X^TX = \begin{bmatrix} \mathbf{x}_1^T\mathbf{x}_1 & \mathbf{x}_1^T\mathbf{x}_2 & \cdots & \mathbf{x}_1^T\mathbf{x}_n \\ \mathbf{x}_2^T\mathbf{x}_1 & \mathbf{x}_2^T\mathbf{x}_2 & \cdots & \mathbf{x}_2^T\mathbf{x}_n \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{x}_n^T\mathbf{x}_1 & \mathbf{x}_n^T\mathbf{x}_2 & \cdots & \mathbf{x}_n^T\mathbf{x}_n \end{bmatrix}. \tag{4}$$

It is shown in [32] that X^TX is a *circulant matrix* with *circularly symmetric* rows. Hence its eigendecomposition [34] is given by

$$X^TX = HDH^T \tag{5}$$

where D is an $n \times n$ matrix given by

$$D = \mathrm{diag}(\lambda_1, \lambda_2, \lambda_2, \lambda_3, \lambda_3, \ldots) \tag{6}$$

and H is an $n \times n$ matrix consisting of the successively higher frequencies, starting from zero frequency, as its columns, which is given by

$$H = \sqrt{\frac{2}{n}} \begin{bmatrix} \frac{1}{\sqrt{2}} & c_0 & -s_0 & c_0 & -s_0 & \cdots \\ \frac{1}{\sqrt{2}} & c_1 & -s_1 & c_2 & -s_2 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \cdots \\ \frac{1}{\sqrt{2}} & c_{n-1} & -s_{n-1} & c_{2(n-1)} & -s_{2(n-1)} & \cdots \end{bmatrix} \tag{7}$$

where $c_k = \cos\left(\frac{2\pi k}{n}\right)$ and $s_k = \sin\left(\frac{2\pi k}{n}\right)$. This development means that an unordered SVD for a planar rotated image sequence can be given in closed form. In particular, $V = H$, i.e., the right singular vectors are pure sinusoids of frequencies that are multiples of $2\pi/n$ radians. To compute U, observe that $U\,\Sigma = XH$, which can be computed efficiently using fast Fourier transform (FFT) techniques [32]. Although the above eigendecomposition analysis does not hold true in general, the following two properties were observed in [32] for arbitrary video sequences:

1. The right singular vectors are well-approximated by sinusoids of frequencies that are multiples of $2\,\pi/n$

radians, and the magnitude-squared of the spectra, i.e., the "power spectra" of the right singular vectors, consist of a narrow band around the corresponding dominant harmonics.

2. The dominant frequencies of the power spectra of the (ordered) singular vectors increase approximately linearly with their index.

These two properties indicate that the right singular vectors are approximately spanned by the first few harmonics. Consequently, by projecting the row space of X to a smaller subspace spanned by a few of the harmonics, the computational expense associated with the SVD computation can be significantly reduced.

Chang's algorithm makes use of the above two properties to determine the first k left singular vectors of X. Let p be such that the power spectra of the first k singular vectors are restricted to the band $[0, 2\pi p/n]$. Owing to the properties of the singular vectors discussed earlier, p is typically not much larger than k. Let H_p denote the matrix comprised of the first p columns of H. Then the first k singular values $\tilde{\sigma}_1, \ldots, \tilde{\sigma}_k$ and the corresponding left singular vectors $\tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k$ of XH_p serve as excellent estimates to those of X.

It was shown in [32] that when p is chosen so as to satisfy $\rho(X^T, \mathbf{h}_1,...,\mathbf{h}_p) \geq \mu$, the quantity $\rho(X, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k)$ turns out to exceed μ for some $k \leq p$, with $\hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_k$ being very good estimates for $\hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_k$, and $\tilde{\sigma}_1, \ldots, \tilde{\sigma}_k$ being very good estimates for $\sigma_1,...,\sigma_k$. The energy recovery ratio $\rho(X, \hat{\mathbf{u}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k)$ can be efficiently approximated by $\sum_{i=1}^{k} \tilde{\sigma}_i^2 / \|X\|_F^2$.

If $p \ll n$ (which is typically true), then the total computation required for Chang's algorithm is approximately $O(mn \log_2 n)$. This compares very favorably with the direct SVD approach ($O(mn^2)$ flops), and in most cases with the updating SVD methods ($O(mnk^2)$ flops). However, this algorithm exploits the correlations along only one dimension and hence cannot be directly applied to general 3D image data sets, which are introduced in the next section.

5 Generation and analysis of 3D image data sets

5.1 Experimental setup

Figure 1 illustrates the experimental setup for generating fully general 3D image data sets, in which the correlated images are characterized by three parameters instead of one. In this setup, camera locations are defined on a spherical patch above the object with two consecutive camera locations separated by an equiangular distance in that patch. The range of these camera locations is characterized by two parameters, i.e., α_l and β_m, while the third parameter γ_n characterizes image plane rotation to capture

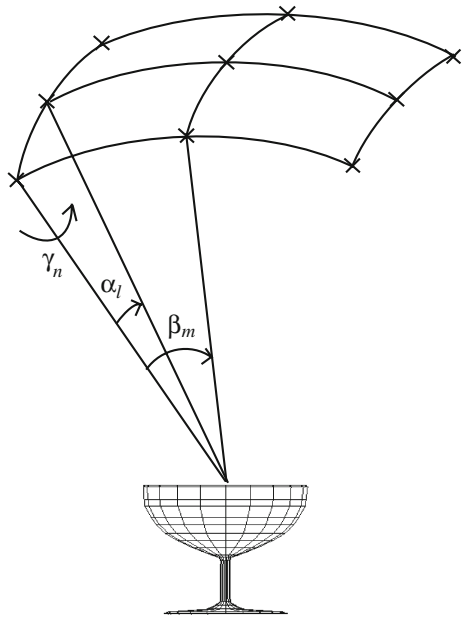

Fig. 1 This figure shows the experimental setup for generating 3D image data sets, in which the images are characterized by three parameters, i.e., α_l, β_m, and γ_n. The *crosses* (x) denote the simulated camera locations that are placed on the spherical patch above the object. The range of the parameters α_l and β_m can be varied with respect to nadir (−40° to 40° was typically used here), whereas the range of γ_n is typically unrestricted

different views of the object in equal increments. In practice, the required images can be captured using a video camera attached to a robot end effector. The robot movement can be controlled to position the camera in one of the specified locations in the spherical patch. The robot end effector can then be rotated to achieve the image plane rotation of the camera for capturing different orientations of the object from the same location.

The images of an object captured using the experimental setup in Fig. 1 are row-scanned and put into one four-dimensional (4D) image array $X_{m \times L \times M \times N}$. The entries along the first dimension of $X(:, l, m, n)$[1] correspond to the row-scanned image of an object taken from camera location (l,m) at the image plane rotation n, where $1 \leq l \leq L$, $1 \leq m \leq M$, and $1 \leq n \leq N$.[2] The entries of X can be rearranged to obtain the following 3D image data matrix:

$$\breve{X} = [\mathbf{x}_{111}, \ldots, \mathbf{x}_{L11}, \mathbf{x}_{121}, \ldots \mathbf{x}_{L21}, \mathbf{x}_{131}, \ldots, \mathbf{x}_{LM1}, \mathbf{x}_{112}, \mathbf{x}_{212}, \ldots, \mathbf{x}_{LM2}, \mathbf{x}_{113}, \ldots, \mathbf{x}_{LMN}] \quad (8)$$

where an image vector $\mathbf{x}_{lmn}$ corresponds to the row-scanned image of an object taken from camera location (l,m) at

[1] A colon (:) in an array argument is used here to specify that all entries in the corresponding dimension of that array are considered.

[2] Please note that we use indices such as l, m, and n (by convention), despite the fact that they are used elsewhere to denote other quantities, i.e., m and n are also used to denote the number of pixels and images, respectively. Context should prevent any confusion.

image plane rotation n. The next subsection analyzes the frequency representation of these 3D image data matrices.

5.2 Frequency analysis of 3D image data sets

Consider the three-dimensional signal $g(x, y, z)$ containing L, M, and N samples in the x, y, and z dimensions, respectively. Its corresponding frequency representation using the 3D DFT can be given by

$$G(\alpha,\beta,\gamma) = \frac{1}{\sqrt{LMN}} \sum_{x=0}^{L-1} \sum_{y=0}^{M-1} \sum_{z=0}^{N-1} g(x,y,z)\omega_L^{\alpha x}\omega_M^{\beta y}\omega_N^{\gamma z} \tag{9}$$

where $0 \le \alpha \le L-1$, $0 \le \beta \le M-1$, and $0 \le \gamma \le N-1$, while $\omega_L = e^{-j2\pi/L}$, $\omega_M = e^{-j2\pi/M}$, and $\omega_N = e^{-j2\pi/N}$. Thus, similar to 1D image data sets, an orthonormal basis for the image data matrix $\breve{X}$ can be generated using the basis for the 3D DFT. In particular, the following 3D array represents one 3D frequency:

$$F_{\alpha\beta\gamma}(x,y,z) = \frac{1}{\sqrt{LMN}}\omega_L^{\alpha x}\omega_M^{\beta y}\omega_N^{\gamma z} \tag{10}$$

where α, β, and γ denote the desired frequency components in three dimensions with $0 \le x \le L-1$, $0 \le y \le M-1$, and $0 \le z \le N-1$. All $F_{\alpha\beta\gamma}$ arrays can be lexicographically ordered (in the same manner as the ordering of $\mathbf{x}_{lmn}$ vectors in $\breve{X}$) into their respective column vectors (denoted $\mathbf{f}_{\alpha\beta\gamma}$) so that the corresponding $LMN \times LMN$ "3D" Fourier matrix is given by

$$\breve{F} = [\mathbf{f}_{000}|\cdots|\mathbf{f}_{\alpha\beta\gamma}|\cdots|\mathbf{f}_{(L-1)(M-1)(N-1)}]. \tag{11}$$

Note that the columns of $\breve{F}$ give the 3D DFT basis for complex matrices. However, for (real-valued) images in 3D image data sets, the matrix $\breve{X}$ in (8) will contain all real values and hence, similar to the basis given by columns of H in (7) for 1D image data sets, $\breve{X}$ will have a real basis. To find this real basis, Euler's formula ($e^{-jx} = \cos x - j\sin x$) can be used to rewrite (10) as follows:

$$\begin{aligned} F_{\alpha\beta\gamma}(x,y,z) = \frac{1}{\sqrt{LMN}}\big[&(c_{\alpha x}c_{\beta y}c_{\gamma z} - c_{\alpha x}s_{\beta y}s_{\gamma z} \\ &- s_{\alpha x}c_{\beta y}s_{\gamma z} - s_{\alpha x}s_{\beta y}c_{\gamma z}) \\ &- j\big(c_{\alpha x}c_{\beta y}s_{\gamma z} + c_{\alpha x}s_{\beta y}c_{\gamma z} \\ &+ s_{\alpha x}c_{\beta y}c_{\gamma z} - s_{\alpha x}s_{\beta y}s_{\gamma z}\big)\big] \end{aligned} \tag{12}$$

where $c_{\alpha x} = \cos\left(\frac{2\pi\alpha x}{L}\right)$, $c_{\beta y} = \cos\left(\frac{2\pi\beta y}{M}\right)$, and $c_{\gamma z} = \cos\left(\frac{2\pi\gamma z}{N}\right)$, while $s_{\alpha x}$, $s_{\beta y}$, and $s_{\gamma z}$ are the corresponding sine components. Let r denote the number of non-zero α, β, and γ frequencies, i.e., $r = 0$, 1, 2, or 3. Then there will be 2^r different sine-cosine combinations for $F_{\alpha\beta\gamma}$. If these sine-cosine combinations are lexicographically ordered and are scaled by $\sqrt{2^r}$ to give orthonormal columns of $H_{\alpha\beta\gamma}$, (where H has either 1, 2, 4 or 8 columns) then the *real* 3D Fourier matrix of size $LMN \times LMN$ can be given by

$$\breve{H} = \left[\mathbf{f}_{000}|H_{001}|\cdots|H_{\alpha\beta\gamma}\cdots\right] \tag{13}$$

where the first column, $\mathbf{f}_{000}$, of $\breve{H}$ refers to the zero frequency component corresponding to $r = 0$. Note that if any of the three dimensions, e.g., L, is even, then only the cosine (real) component of the corresponding maximum "real" frequency, i.e., $\frac{L}{2}+1$ is considered, otherwise both cosine (real) and sine (imaginary) components of $\frac{L+1}{2}$ are considered while generating the corresponding orthonormal columns in $\breve{H}$.

The resulting matrix $\breve{H}$, which is generated for a given $\breve{X}$, can be used to extend Chang's algorithm to compute the approximate SVD of $\breve{X}$. In particular, the row space of $\breve{X}$ can be projected to the first few columns of $\breve{H}$ and the SVD of $\breve{X}\breve{H}_p$ can be used to approximate the SVD of $\breve{X}$, where $\breve{H}_p$ denotes the matrix containing the first p columns of $\breve{H}$. The computation of $\breve{X}\breve{H}$ can be performed efficiently using DFT techniques. However, the implementation and use of real 3D bases in $\breve{H}$ is not as trivial as in the 1D case. This is discussed in the next subsection.

5.3 Efficient computation of $\breve{X}\breve{H}$ using FFT techniques

Consider the 3D DFT of $g(x,y,z)$ in (9), which can be rewritten as

$$G(\alpha,\beta,\gamma) = \sum_{l=0}^{1}\sum_{m=0}^{1}\sum_{n=0}^{1}(-j)^{l+m+n}g_{lmn} \tag{14}$$

where $j = \sqrt{-1}$ and

$$\begin{aligned} g_{000} &= \frac{1}{\sqrt{LMN}}\sum_{x=0}^{L-1}\sum_{y=0}^{M-1}\sum_{z=0}^{N-1} g(x,y,z)c_{\alpha x}c_{\beta y}c_{\gamma z}, \\ g_{001} &= \frac{1}{\sqrt{LMN}}\sum_{x=0}^{L-1}\sum_{y=0}^{M-1}\sum_{z=0}^{N-1} g(x,y,z)c_{\alpha x}c_{\beta y}s_{\gamma z}, \\ &\vdots \\ g_{111} &= \frac{1}{\sqrt{LMN}}\sum_{x=0}^{L-1}\sum_{y=0}^{M-1}\sum_{z=0}^{N-1} g(x,y,z)s_{\alpha x}s_{\beta y}s_{\gamma z}. \end{aligned} \tag{15}$$

The eight terms g_{lmn} can be calculated using a series of "nested" FFT's as follows:

$$\begin{aligned} XF_1 &= \Re(\mathrm{fft}_l(\Re(\mathrm{fft}_m(\Re(\mathrm{fft}_n(X)))))), \\ XF_2 &= \Re(\mathrm{fft}_l(\Re(\mathrm{fft}_m(\Im(\mathrm{fft}_n(X)))))), \\ &\vdots \\ XF_8 &= \Im(\mathrm{fft}_l(\Im(\mathrm{fft}_m(\Im(\mathrm{fft}_n(X)))))) \end{aligned} \tag{16}$$

where $X_{m \times L \times M \times N}$ is the original 4D image array, from which the image data matrix $\breve{X}$ is generated, while fft_l, fft_m,

and fft_n denote the FFT of X computed along the α_l, β_m, and γ_n dimension, respectively.

Note that the arrays XF_1 through XF_8 will each have $mLMN$ elements (i.e., the same size as that of X), which suggests that these "multiplication" arrays have a significant amount of duplicated information resulting in unnecessary processing. In particular, consider the FFT computation of X along its γ_n dimension. The corresponding Fourier basis vectors will come in complex conjugate pairs. However, because X contains all real entries, the real and imaginary components of these complex basis vectors will form the corresponding "real" basis, which will result in two sets of duplicate basis vectors. Note that only one of these two sets is necessary and sufficient to proceed with the FFT computation along the β_m dimension. Thus the XF_1 array can be computed efficiently using the following steps:

1. Compute $F_n = \text{fft}_n(X)$, which will be of the same size as X.
2. Compute $F_m = \text{fft}_m(\Re(F_n(:,:,:,1:\frac{N+1}{2})))$, which will have $mLM\left(\frac{N+1}{2}\right)$ elements.[3]
3. Compute $F_l = \text{fft}_l(\Re(F_m(:,:,1:\frac{M+1}{2},:)))$, which will have $mL\left(\frac{M+1}{2}\right)\left(\frac{N+1}{2}\right)$ elements.
4. Finally, let $XF_1 = \Re(F_{l1}(:,1:\frac{L+1}{2},:,:))$, which will have $m\left(\frac{L+1}{2}\right)\left(\frac{M+1}{2}\right)\left(\frac{N+1}{2}\right)$ elements.

The remaining arrays, i.e., XF_2 through XF_8, can be computed similarly and all these arrays can be rearranged into their respective 2D matrices, denoted XH_1 to XH_8. Without loss of generality, for each frequency combination, the corresponding columns in the XH_i matrices can be given equal importance. However, the relative importance of one frequency combination with any other is not as trivial as in the 1D case. Therefore, this issue needs to be addressed before combining the columns of the XH_i matrices to form the final XH matrix [which is essentially the multiplication of $\breve{X}$ in (8) and $\breve{H}$ in (14)] of size $m \times LMN$. This ordering of the columns of $\breve{X}\breve{H}$ in terms of their "importance" is addressed in the next section before extending Chang's algorithm to general 3D image data sets.

6 Proposed ordering of 3D frequency components

6.1 Introduction

This section explains the heuristics behind ordering different 3D frequency components in terms of their energy recovery ability for a given 3D image data set. For this study, several image data sets were generated by ray-tracing different objects as per the specifications of the experimental setup given in Fig. 1. Figure 2 shows eight such artificial objects that were considered in this study. For each object, a total of LMN images were ray-traced and the corresponding 3D image data matrix, $\breve{X}$, of size $m \times LMN$ was generated, where $m = 128 \times 128 = 2^{14}$. Recall that LM is equal to the total number of camera locations above the object, while N denotes the number of images that are captured (after rotating the image plane of the camera) at each camera location.

To propose a "good" ordering of 3D frequencies based on the given specifications of 3D image data sets, Object 1 in Fig. 2 is used as a representative example here. In particular, two ray-traced 3D image data sets of this object are evaluated in detail. These two image data sets have the same number of images with $L = M = N = 9$ and the parameters α_l and β_m in both of these image data sets are allowed to span 80 degrees each. More specifically, the camera locations along the parameters α_l and β_m are placed from $-40°$ to $40°$ with $10°$ separation between the two consecutive camera locations along both α_l and β_m. The only difference in these two image data sets is the range of the parameter γ_n. In one image data set, γ_n spans only 80 degrees (image plane rotation from $0°$ to $80°$ in $10°$ increments), while in the other image data set, γ_n spans 320 degrees (image plane rotation from $0°$ to $320°$ in $40°$ increments). Because the first image data set assigns equal ranges to all three parameters, it will be referred to as the "equiangular range" (ER) image data set. On the other hand, the second image data set assigns unequal ranges to the three parameters and hence it will be referred to as the "non-equiangular range" (NER) image data set.

To analyze the above two representative image data sets, consider a three-dimensional ($L \times M \times N$) array, which has N two-dimensional slices consisting of LM entries each. Let the ith slice correspond to the ith value of γ_n. In particular, the entries in the first slice denote images corresponding to the minimum value of γ_n, while the entries in the last slice denote images corresponding to the maximum value of γ_n.

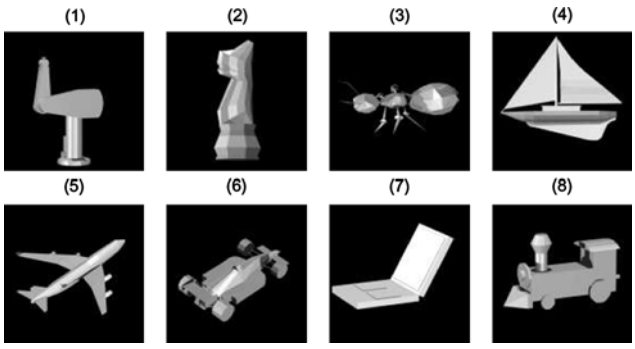

Fig. 2 This figure shows eight artificial (ray-traced) objects that are used in this study. Each image of the object is of size 128×128, resulting in an image data matrix $\breve{X}$ of size $2^{14} \times LMN$ for each object

[3] The notation $(1{:}i)$ here refers to the first i entries in the corresponding dimension of an array.

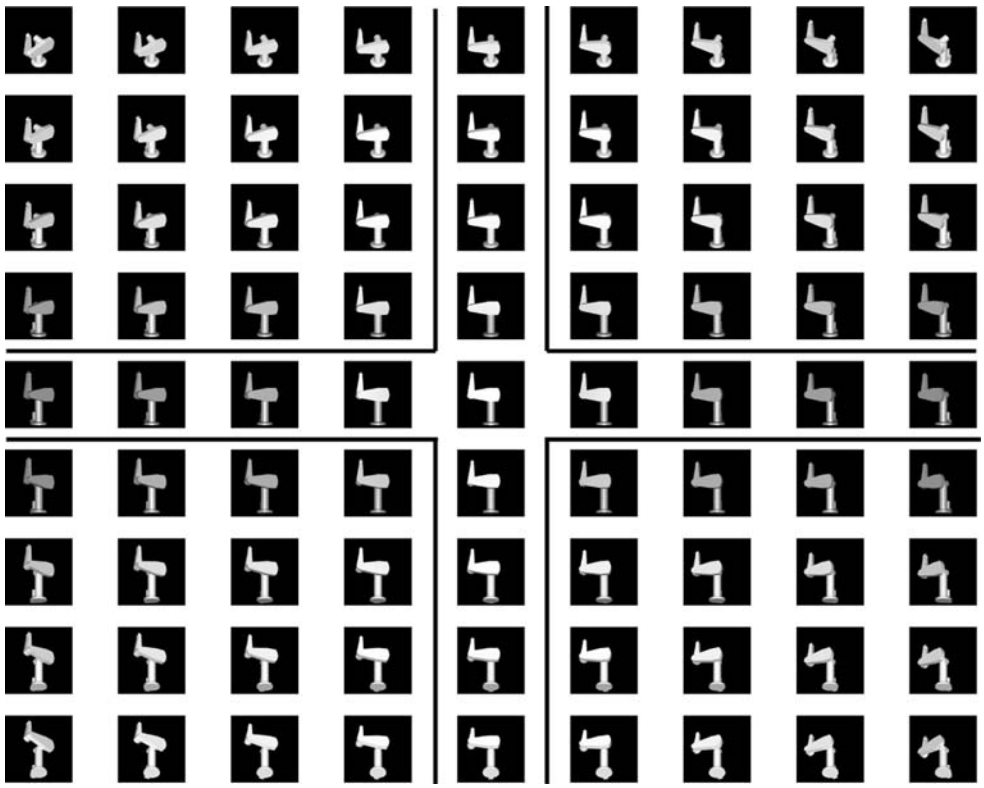

Fig. 3 This figure shows the *images in the first slice* (corresponding to $\gamma_n = 0°$) of both equiangular range and non-equiangular range image data sets. Within these images, the parameter α_l varies from left to right from $-40°$ to $40°$ in $10°$ increments, while the parameter β_m varies from top to bottom from $-40°$ to $40°$ in $10°$ increments. In particular, the *images in the middle column* (confined by the *vertical solid lines*) of this figure correspond to $\alpha_l = 0°$, while the images in the middle row (confined by the *horizontal solid lines*) of this figure correspond to $\beta_m = 0°$

With this terminology, because the minimum value of γ_n for both the representative image data sets is 0°, both sets have the same first slice, as shown in Fig. 3.

6.2 Equiangular range (ER) image data set

This subsection evaluates the variation of images along the three parameters for the ER image data set. In particular, images in the middle row and the middle column of Fig. 3 are used here as representative examples for evaluating the variation of images in the parameters α_l and β_m, respectively. The image in the center (corresponding to $\alpha_l = \beta_m = \gamma_n = 0°$) of Fig. 3 is planar rotated from 0° to 80° to obtain nine images in 10° increments (refer to Fig. 4) and the resulting image set is used as a representative example for evaluating the variation of images in the parameter γ_n. Figure 5 shows the entries of the first nine right singular vectors of this image data set corresponding to the variation of images in the three parameters.[4] Similar to 1D image data sets, Fig. 5 shows that the right singular vector entries corresponding to variation of images along each parameter are well-approximated by sinusoids of increasing frequencies starting from zero. This suggests that the frequency combinations in $\breve{H}$ for such image data sets can be ordered by the increasing sum of their frequencies, i.e., $(\alpha + \beta + \gamma)$, from 0 to $\left(\left\lceil\frac{L+1}{2}\right\rceil + \left\lceil\frac{M+1}{2}\right\rceil + \left\lceil\frac{N+1}{2}\right\rceil\right)$, where $\lceil . \rceil$ represents the ceiling operation. Note that this is an intuitive extension of the 1D image data set case due to the equiangular range of all three parameters in this image data set.

Even if one attempts to order the columns in $\breve{H}$ with increasing sum of frequencies in three dimensions, there will be many frequency combinations that will have the same sum. For example, $\alpha + \beta + \gamma = 2$ will result in six different frequency combinations, i.e., (1, 1, 0), (1, 0, 1), (0, 1, 1), (2, 0, 0), (0, 2, 0), and (0, 0, 2). (In general, $\alpha + \beta + \gamma = n$ will result in $\binom{n+2}{2}$ frequency combinations.) Therefore, it is desirable to have an effective ordering within the group of frequency combinations with the same sum of frequencies. To achieve this, consider the right singular vector entries in Fig. 5 again. The plots show that variation of images along γ_n requires relatively higher

[4] Note that the image data set under consideration has $LMN = 729$ images in total. Hence, the corresponding right singular vectors will each have 729 elements. However, only nine of those 729 entries can be used to monitor the variation of images along any one parameter while keeping the other two parameters constant.

Fig. 4 This figure shows the variation of images along the parameter γ_n in the equiangular range (*ER*) image data set with $\alpha_l = \beta_m = 0°$. Within these images, the parameter γ_n varies from left to right from 0° to 80° in 10° increments. Note that the leftmost image in this figure matches with the center image of Fig. 3

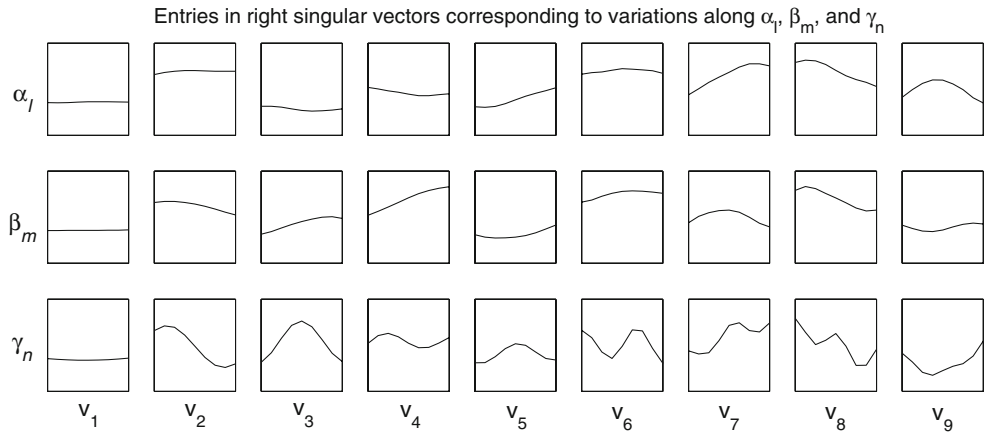

Fig. 5 The *plots* in this figure show entries of the first nine right singular vectors of the ER image data set for Object 1 in Fig. 2. In particular, the *first row* shows entries of the first nine right singular vectors corresponding to variation of nine images along the α_l parameter with $\beta_m = \gamma_n = 0°$. (The corresponding images are shown in the *middle row* of Fig. 3.) The *second row* shows entries of the same right singular vectors corresponding to variation of nine images along the β_m parameter with $\alpha_l = \gamma_n = 0°$ (refer to the middle column of Fig. 3). Finally, the *third row* shows entries of the same nine right singular vectors corresponding to variation of nine images along the γ_n parameter with $\alpha_l = \beta_m = 0°$. (The *image in the center* of Fig. 3 is planar rotated from 0° to 80° in 10° increments to obtain the corresponding images.) Note that only nine (out of a total of *LMN* = 729) right singular vector entries are plotted in each plot here

frequencies than the variation along the other two parameters. A potential explanation for this is that the images varying along the parameter γ_n are planar rotations of each other. Hence the right singular vectors of the corresponding 1D image data matrices would be pure sinusoids with increasing frequencies (due to the properties of circulant matrices [32]). On the other hand, the images varying along the parameters α_l and β_m are not planar rotations and hence the right singular vectors of the corresponding 1D image data matrices will typically not be pure sinusoids.[5] Therefore higher γ frequencies are required as compared to α and β frequencies (within the group of frequency combinations having the same sum of frequency components) to achieve the same level of "importance."

The above observations motivate ordering the frequency combinations for the ER image data set as follows:

1. Group frequency combinations in increasing order of the frequency sum $M_1 = \alpha + \beta + \gamma$.
2. Within a group of frequencies with the same frequency sum M_1, order these combinations in increasing value of $M_2 = \alpha + \beta$.
3. Within a subgroup of frequencies with equal values of M_2, order the combinations based on $M_3 = \alpha - \beta$ using the following ordering on the set of integers:

   $$0, 1, -1, 2, -2, 3, -3, \ldots$$

This gives more importance to the combinations with lower individual α and β frequencies over combinations with higher α and β frequencies. Alternatively, one can obtain the same ordering with $M_3' = \max(\alpha,\beta)$ with a preference given to a smaller β over a smaller α in the case when two combinations have the same M_3' value.

Note that the above ordering is uniquely defined since there is a one-to-one correspondence between (M_1, M_2, M_3) and (α, β, γ).

Recall that there will be a maximum of eight sine-cosine combinations for the given frequency combination of (α, β, γ). All of these sine-cosine combinations are considered to be equally important and hence their ordering can be arbitrarily chosen. The analysis of the proposed frequency ordering considers the following ordering of the sine-cosine combinations for the given frequency combination: (cos, cos, cos), (cos, cos, sin), (cos, sin, cos), (cos, sin, sin), (sin, cos, cos), (sin, cos, sin), (sin, sin, cos), and (sin, sin, sin).

Using the above heuristics, the first few frequency combinations in the proposed ordering for the ER image

[5] A simple way of viewing this is that the images varying along γ_n contain the "same" information in all of them, while the images varying along the other two parameters contain slightly different information in consecutive images.

Table 1 First few frequency combinations for ER image data set

α	0	0	1	0	0	1	0	1	2	0	0	1	0	1	2	0	2	1	3	0
β	0	0	0	1	0	0	1	1	0	2	0	0	1	1	0	2	1	2	0	3
γ	0	1	0	0	2	1	1	0	0	0	3	2	2	1	1	1	0	0	0	0

data set are given in Table 1, while the pictorial view is shown in Fig. 6. Note that because $L = M = N = 9$, there are five "real" frequencies (0 through 4) along all three dimensions for this image data set. This results in a total of 125 frequency combinations with the maximum sum of three frequencies being 12. Figure 7 shows that the frequency combinations in this proposed ordering give a very good approximation as compared to that using the optimum ordering in terms of the energy recovery ratio (with maximum relative error of 1.57%).

It was observed [35] that the above heuristics worked well for ordering the frequency combinations for a wide variety of ER image data sets. Therefore, the proposed measures (M_1, M_2, M_3) can be used for generating $\breve{H}$ for a given ER image data set if the difference between the number of images sampled along all three dimensions is within "reasonable" limits, for e.g., within an order of magnitude.

6.3 Non-equiangular range (NER) image data set

This subsection evaluates the variation of images along the three parameters for the NER image data set. Because the data representing the variation along the parameters α_l and β_m remain the same as in the ER image data set, the only data that must be generated again are the images varying along the parameter γ_n. In particular, the image in the center (corresponding to $\alpha_l = \beta_m = \gamma_n = 0°$) of Fig. 3 is now planar rotated from 0° to 320° to obtain nine images in 40° increments (refer to Fig. 8). The resulting image set is used as a representative example for evaluating the variation of images in the parameter γ_n.

The variation of images in α_l and β_m for this NER image data set is almost identical to that in the ER image data set [35], however, images in the parameter γ_n vary quite differently. In particular, Fig. 9 shows the corresponding entries of the first nine right singular vectors of this image data set. These entries indicate that the sum of frequencies does not seem to be a good starting measure to order the frequency components for this image data set, as even the first few right singular vectors seem to contain much higher γ frequencies. In particular, apart from the zero frequency, all four γ frequencies[6] are as important as the first α and β frequencies. This is due to the fact that the range of the parameter γ_n (320°) is four times larger than that of the parameters α_l and β_m. However, within the same γ frequency, it appears that α and β frequencies should be ordered in their increasing sum.

Several other experiments were conducted for a variety of NER image data sets and some general trends were observed [35] about the relative importance of α, β, and γ frequencies for different ranges of α_l, β_m, and γ_n. Let η, τ, and δ represent the ratios between the three parameters, i.e., $\eta : \tau : \delta = \max(\alpha_l) - \min(\alpha_l) : \max(\beta_m) - \min(\beta_m) : \max(\gamma_n) - \min(\gamma_n)$[7] and let the corresponding ratio of the relative importance of α, β, and γ frequencies be given by $\eta_\alpha : \tau_\beta : \delta_\gamma$. It was observed that

$$\eta_\alpha : \tau_\beta : \delta_\gamma = \begin{cases} \eta : \tau : \delta & \text{if } \delta \geq \max(\eta, \tau) \\ \eta : \tau : 2\delta & \text{otherwise} \end{cases} \tag{17}$$

served as a good heuristic.

The above observations motivate an NER ordering where the measures are now given by:

- $M_1 = \alpha + \beta$
- $M_2 = \alpha - \beta$
- $M_3 = \gamma$.

These measures place a higher priority on α and β, as compared to the ER ordering case, so that higher frequencies in γ are considered sooner. However, like the ER case, there is a preference for combinations of lower α and β frequencies as opposed to combinations with a high frequency. With the known η_α, τ_β, and δ_γ [refer to (17)], one can order the family of 3-tuples of non-negative integers in the following way:

- First, consider the set of 3-tuples of integers, (α, β, γ), such that $0 \leq \alpha \leq \eta_\alpha$, $0 \leq \beta \leq \tau_\beta$, and $0 \leq \gamma \leq \delta_\gamma$. There are $(\eta_\alpha + 1)$ $(\tau_\beta + 1)$ $(\delta_\gamma + 1)$ such combinations. Order these 3-tuples using the NER ordering measures described above and denote the ordered set by S_1.
- Next, starting with the set of 3-tuples of integers, (α, β, γ), such that $0 \leq \alpha \leq 2\eta_\alpha$, $0 \leq \beta \leq 2\tau_\beta$, and $0 \leq \gamma \leq 2\delta_\gamma$, remove all those 3-tuples that are in S_1. Order the remaining 3-tuples using NER ordering, concatenate this ordered set with S_1, and denote the resulting ordered set as S_2.
- Continue the above process inductively by the following rule. Remove the 3-tuples in S_n from the set of 3-tuples (α, β, γ), such that $0 \leq \alpha \leq (n+1)\eta_\alpha$, $0 \leq \beta \leq (n+1)\tau_\beta$, and $0 \leq \gamma \leq (n+1)\delta_\gamma$. Order the remaining $[(n+1)\eta_\alpha + 1]$ $[(n+1)\tau_\beta + 1]$ $[(n+1)\delta_\gamma + 1] - (n\eta_\alpha + 1)$ $(n\tau_\beta + 1)$ $(n\delta_\gamma + 1)$ 3-tuples using NER ordering and concatenate this ordered list to S_n to obtain S_{n+1}.

[6] Note that because $L = M = N = 9$, there are five "real" frequencies (0 through 4) along all three parameters for this NER image data set.

[7] η,τ, and δ must be integers with no common factors.

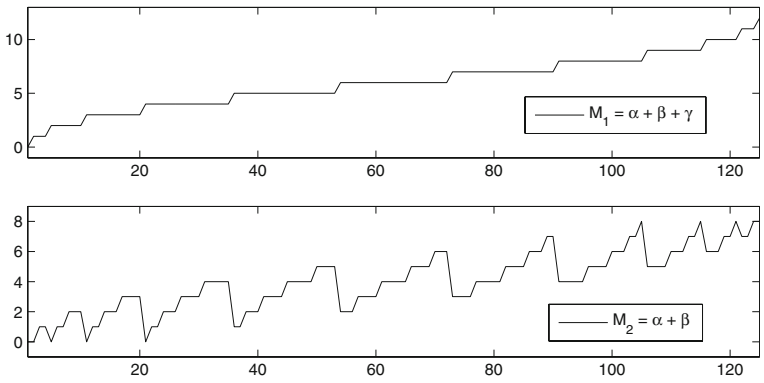

Fig. 6 The *plots* in this figure give a pictorial representation of the proposed ordering of all frequency combinations for the ER image data set for Object 1 in Fig. 2. These *plots* are shown as a function of the frequency combination index and follow the measures M_1 and M_2 outlined for ordering these combinations for the ER image data set

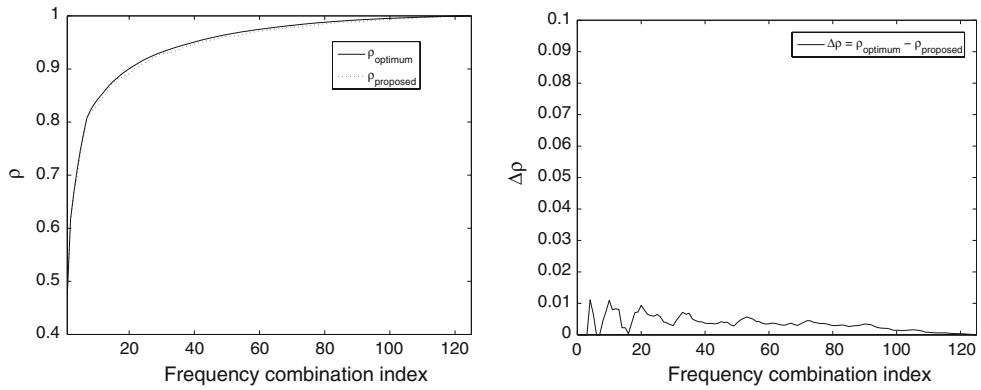

Fig. 7 The *plots* in this figure show the comparison of the proposed ordering with the optimum ordering of the frequency combinations in terms of their energy recovery ability in $\breve{X}$ for the ER image data set for Object 1 in Fig. 2. The *plot on the left* shows the energy recovery ratio as a function of frequency combination index, while the *plot on the right* shows the corresponding difference in the energy recovery ratios

Fig. 8 This figure shows the variation of images along the parameter γ_n in the non-equiangular range (*NER*) image data set with $\alpha_l = \beta_m = 0°$. Within these images, the parameter γ_n varies from left to right from 0° to 320° in 40° increments. Note that the *leftmost image* in this figure matches with the center image of Fig. 3

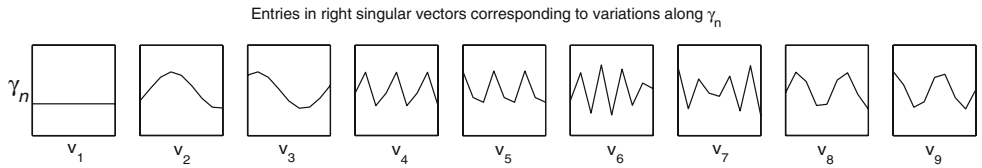

Fig. 9 This figure shows entries of the first nine right singular vectors of the NER image data set for Object 1 in Fig. 2 corresponding to variation of nine images along the γ_n parameter with $\alpha_l = \beta_m = 0°$. (The *image in the center* of Fig. 3 is planar rotated from 0° to 320° in 40° increments to obtain the corresponding images.) Note that only nine (out of a total of *LMN* = 729) *right singular vector entries* are plotted in each plot here

The above procedure is illustrated in Fig. 10 for $(\eta_\alpha : \tau_\beta : \delta_\gamma = 1{:}1{:}2)$.

Using the above heuristics, the first few frequency combinations in the proposed ordering for the NER image data set are given in Table 2, while its pictorial view is shown in Fig. 11. The maximum relative error in the energy recovery ratio for the representative NER image data set computed using this proposed ordering as opposed

Fig. 10 This figure shows two ordered sets of 3-tuples (S_1 and S_2-S_1) using NER ordering measures for the case where (η_α:τ_β:δ_γ) is equal to (1:1:2)

$S_1 =$

α	0	0	0	1	0	1	0	1	0	1	1	1
β	0	0	0	0	1	0	1	0	1	1	1	1
γ	0	1	2	0	0	1	1	2	2	0	1	2

$S_2 - S_1 =$

α	0	0	1	0	1	0	1	1	2	⋯	0	2	⋯	1	2	⋯	2
β	0	0	0	1	0	1	1	1	0	⋯	2	1	⋯	2	2	⋯	2
γ	3	4	3	3	4	4	3	4	0	⋯	4	0	⋯	4	0	⋯	4

to that computed using the optimum ordering is 1.76%, which indicates the promise of the proposed heuristic.

Although the above two ordering heuristics (ER and NER) determine the frequency ordering for $\breve{X}$ solely based on the range of all three parameters, empirical results [35] showed that they successfully covered the frequency ordering for almost all 3D image data sets. Thus these ordering heuristics for a given $\breve{X}$ along with the efficient computation of $\breve{X}\breve{H}$ using FFT techniques (refer to Sect. 4.3) can be effectively used to extend Chang's eigendecomposition algorithm to compute the SVD of $\breve{X}$, which is the topic of the next section.

7 A fast eigendecomposition algorithm for 3D image data sets

The entire algorithm for the fast computation of a partial SVD of $\breve{X}_{m\times n}$, where $n = LMN$, can be summarized as follows:

1. Form the matrix $Y = \breve{X}\breve{H}$ using fast Fourier transform techniques described in Sect. 4 such that the columns of Y are placed in the order discussed in Sect. 5 based on the range of the α_l, β_m, γ_n parameters.
2. Determine the smallest number p such that $\rho(\breve{X}^T, \mathbf{h}_1, \ldots, \mathbf{h}_p) > \mu$, where μ is the user-specified reconstruction ratio. The key observation here is that the matrix $\breve{X}\breve{H}_p$ can be constructed directly from the first p columns of Y.
3. Compute the SVD of $\breve{X}\breve{H}_p = \sum_{i=1}^{p} \tilde{\sigma}_i \tilde{\hat{\mathbf{u}}}_i \tilde{\hat{\mathbf{v}}}_i^T$.
4. Return $\tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k$ such that $\rho(\breve{X}, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k) \geq \mu$.

The computational expense of the proposed algorithm is now analyzed. The cost incurred in Step 1, i.e., performing the FFT of the 3D image data matrix $\breve{X}$, requires $O(mn\log_2 n)$ flops. Step 2, that of estimating p, requires $O(mp)$ flops. In Step 3, the cost of computing the SVD of the matrix comprising the first p columns of $\breve{X}\breve{H}$ is $O(mp^2)$. Step 4, determining the needed dimension k, requires $O(mnk)$ flops. If $p \ll n$, then the total computation required is approximately $O(mn\log_2 n)$, otherwise it is approximately $O(mp^2)$. This compares favorably with the direct SVD approach, which requires $O(mn^2)$ flops.

For 1D image data sets, it was observed that the individual pixel values do not change rapidly across the sequence of images resulting in $p \ll n$. However, for 3D image data sets, the value of p increases because the images are not as highly correlated and the right singular vectors are not as closely approximated by pure sinusoids. To illustrate this phenomenon, consider the NER image data set with $L = M = N = 9$. Let the columns of $\breve{H}$ be in descending order of their ability to recover energy in $\breve{X}$ using $\rho(\breve{X}^T, \breve{\mathbf{h}}_1, \breve{\mathbf{h}}_2, \ldots)$. Then the dotted line in Fig. 12 shows $\rho(\breve{X}, \breve{\mathbf{h}}_1, \ldots, \breve{\mathbf{h}}_p)$ as a function of p, while the solid lines show $\rho(\breve{X}, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k)$ for $k = 1, 2, \ldots, p$ and $p = 50$, 100, 150, 200. It is evident that while the $\tilde{\hat{\mathbf{u}}}_i$ give good estimates of the $\hat{\mathbf{u}}_i$, $\rho(\breve{X}^T, \breve{\mathbf{h}}_1, \breve{\mathbf{h}}_2, \ldots)$ does not give as tight a lower bound on $\rho(\breve{X}, \hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_k)$ as in the 1D case.

Table 2 First few frequency combinations for the NER image data set

α	0	0	...	0	1	0	1	0	...	1	0	1	1	...	1	2	0	2	0	...	2	0
β	0	0	...	0	0	1	0	1	...	0	1	1	1	...	1	0	2	0	2	...	0	2
γ	0	1	...	4	0	0	1	1	...	4	4	0	1	...	4	0	0	1	1	...	4	4

It is possible to provide a tighter bound on $\rho(\breve{X}, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_p)$ by using an orthonormal basis for the range of $\breve{X}\breve{H}_p$, however, the following example illustrates why this may not be desirable. Consider the case shown in Fig. 13 where the user-specified reconstruction ratio $\mu = 0.95$. The required dimension of the true eigenspace for the homogeneously sampled image data set is $k^* = 80$, however, the first $p = 166$ optimally ordered columns of $\breve{X}\breve{H}$ are required to attain $\rho(\breve{X}, \breve{\mathbf{h}}_1, \ldots, \breve{\mathbf{h}}_p) \geq \mu$. If all these $p = 166$ columns of $\breve{X}\breve{H}$ are used, then the approximated eigenimages result in $\rho(\breve{X}, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_{166}) = 0.9693$ (refer to Fig. 13(a)), whereas one only needs $k = 83$ eigenimages to obtain $\rho(\breve{X}, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k) = 0.95$. This indicates that a large number of frequency combinations are required to span the first k eigenimages. Now if the columns of $\breve{X}\breve{H}$ are orthonormalized using the QR decomposition $(\breve{X}\breve{H} = QR)$, then $\rho(\breve{X}, \mathbf{q}_1, \ldots, \mathbf{q}_{99}) \geq \mu$, where $\mathbf{q}_i$ denotes the ith column of the resulting Q matrix. Thus the value of p is reduced from 166 to 99, which potentially improves the offline computational efficiency of calculating the SVD of $\breve{X}$. The approximated eigenimages for $\breve{X}$ would now be given by the columns of $\breve{U} = QU_r$ where U_r denotes the matrix containing the left singular vectors of R. Figure 13(b) shows that these approximated eigenimages result in $\rho(\breve{X}, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_{99}) = 0.9503$, which

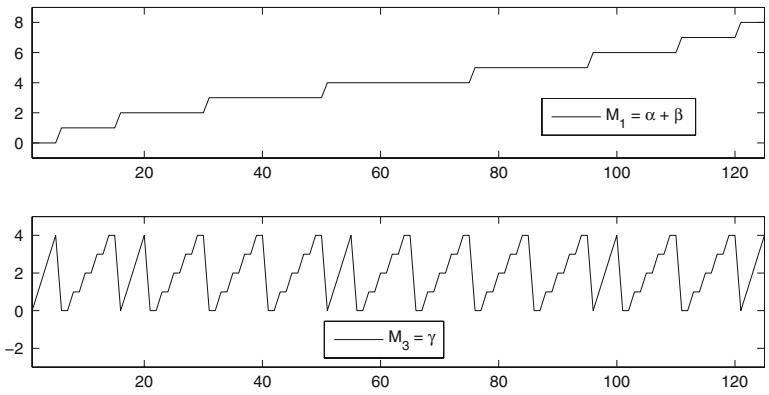

Fig. 11 The *plots* in this figure give a pictorial representation of the proposed ordering of all frequency combinations for the NER image data set for Object 1 in Fig. 2. These *plots* are shown as a function of the frequency combination index and follow the measures M_1 and M_3 outlined for ordering these combinations for the NER image data set. Note that the measure M_2 automatically follows M_1 and M_3

satisfies the user-specified accuracy very closely. However, the resulting dimension of the approximate eigenspace is now $k = 99$, which results in a significant increase in online computations, which are proportional to the dimension of the subspace used in the approximation. Note that the difference between k and k^* increases drastically as the number of images in $\breve{X}$ increases. Also there is no simple method to determine a priori how many columns of $\breve{X}\breve{H}$ should be considered for computing its QR decomposition.

8 Experimental results

The proposed extension of Chang's algorithm for computation of the partial SVD of 3D image data sets was evaluated using the eight artificial (ray-traced) objects shown in Fig. 2 and the sixteen real objects shown in Fig. 14. In particular, the algorithm was used to calculate the partial SVD of $\breve{X}$ with $\mu = 0.95$ for three image data sets for each artificial object and for two image data sets for each real object. Table 3 explains the specifications of the corresponding image data sets.[8]

Tables 4, 5, and 6 summarize the performance of the algorithm, showing p, k, k^*, and the computation times for image data sets of artificial objects. Compared to the direct SVD, the speed-up factors with the proposed algorithm are in the range of 3.00–17.00 with averages of 6.28, 5.93, and 7.31 for these three image data sets. Because the original image frames for all the objects have the same resolution, the speed-up factor here depends on the values of p and k. This is evident from all the table entries, as the minimum speed-up factors are obtained for objects 5 and 7 with the maximum obtained for object 2. Note that despite the higher number of images in data set three, which requires an increase in the time to perform step 4, the overall speed-up factors are comparable. Note that the value of p is much larger than k in most of the cases indicating that a large

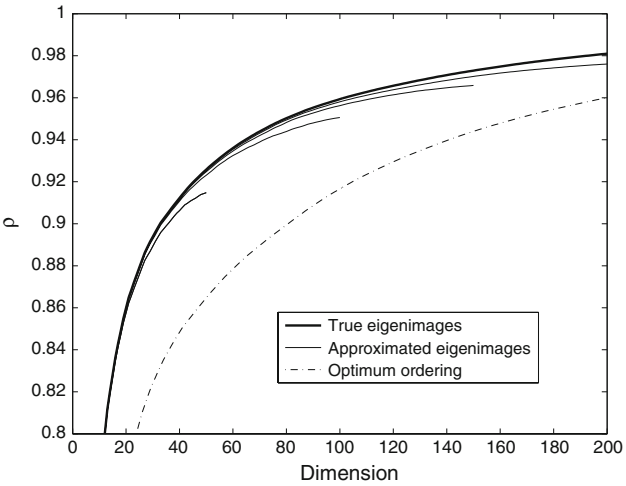

Fig. 12 This figure shows the typical relationship between the true left singular vectors, the computed estimates (as a function of k, $1 \leq k \leq p$, for several fixed values of p), and ordered frequency combinations. The *plots* shown here are for the 3D image data matrix generated for the NER image data set for Object 1 in Fig. 2 with $L = M = N = 9$ and 90, 90, 360° ranges for α_l, β_m, and γ_n parameters, respectively

number of frequency combinations are required to span the first k eigenimages. It is also interesting to note that the value of k is also generally large (with respect to the number of images in the image data set) indicating that the 3D image data sets have far less correlation between images than in the 1D case.

Tables 7 and 8 summarize the performance of the algorithm, showing p, k, k^*, and the computation times for image data sets for real objects. Compared to the direct SVD, the speed-up factors with the proposed algorithm for these real objects are in the range of 3.82–22.47 with averages of 7.80 and 21.64 for the two image data sets. Note that the proposed algorithm performs better on real objects than on artificial objects. Additionally, if the user-specified accuracy μ is reduced to 0.90, then the range of speed-up factors of the proposed algorithm as compared to

[8] Due to the physical limitations of the robot, the range for the α_l and β_m parameters for the real objects was restricted to 60°.

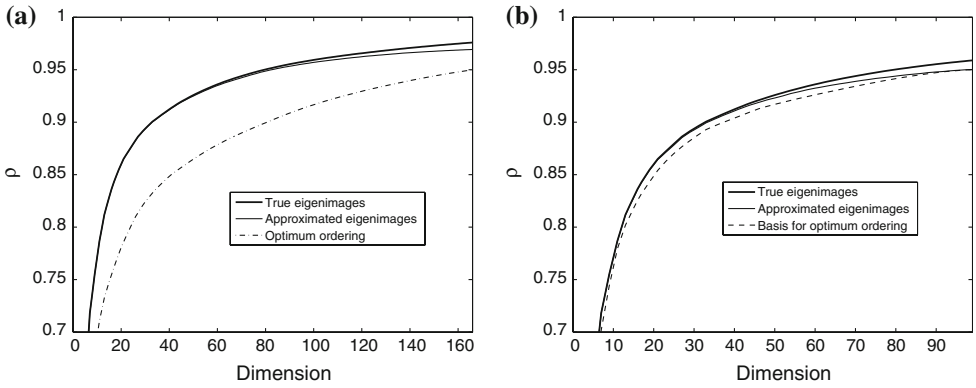

Fig. 13 This figure shows the energy recovery ratio plots for $\breve{X}$ using the columns of the $\breve{X}\breve{H}$ matrix for computing the approximated eigenimages in (**a**) versus an orthonormal basis for the $\breve{X}\breve{H}$ matrix in (**b**)

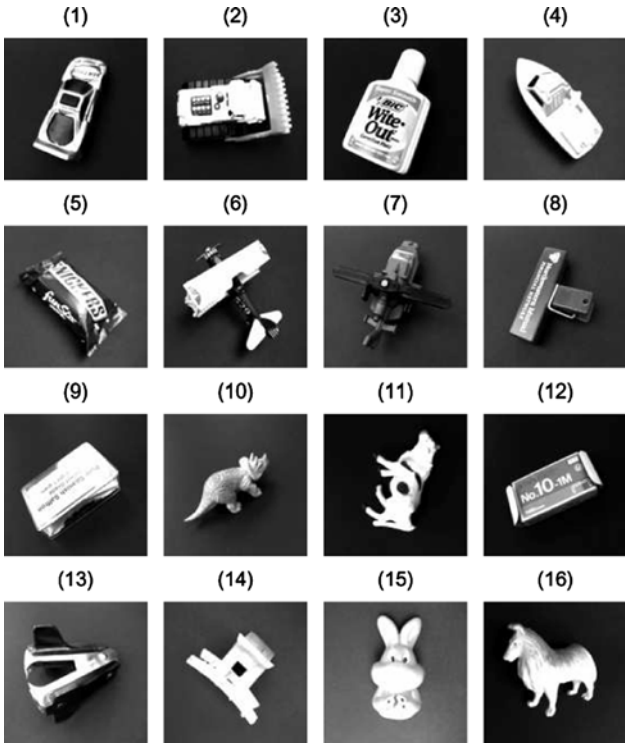

Fig. 14 This figure shows 16 real objects that are used in this study. Each image of the object is of size 128 × 128, resulting in an image data matrix $\breve{X}$ of size $2^{14} \times LMN$ for each object

Table 3 Image data sets generated for each object in Figs. 2 and 14

Image data set	Number of images	Range	Angular separation
Artificial objects			
1	10 × 10 × 10	81, 81, 81	9, 9, 9
2	10 × 10 × 10	81, 81, 324	9, 9, 36
3	9 × 9 × 36	80, 80, 360	10, 10, 10
Real objects			
1	9 × 9 × 9	60, 60, 60	7.5, 7.5, 7.5
2	9 × 9 × 33	60, 60, 240	7.5, 7.5, 7.5

the direct SVD improves to 5.23–23.12 with averages of 9.74, 9.19, and 14.35 for image data sets for artificial objects, and averages of 10.26 and 32.48 for image data sets for real objects.

The quality of the resulting eigendecomposition was also evaluated using the error measures described earlier for both real and artificial objects. In particular, Tables 4, 5, 6, 7, and 8 show that the subspace criterion measure between the true and approximated eigenimages for all image data sets is in the range of 0.9213–0.9930. Also, the difference between $\rho(\breve{X}, \hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_{k^*})$ and $\rho(\breve{X}, \tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k)$ for each set was less than 0.17%, with an average of 0.09%, which reveals that $\{\tilde{\hat{\mathbf{u}}}_1, \ldots, \tilde{\hat{\mathbf{u}}}_k\}$ provides a very good approximate basis for the first k^* eigenimages $\{\hat{\mathbf{u}}_1, \ldots, \hat{\mathbf{u}}_{k^*}\}$.

9 Conclusion

This paper considered the efficient computation of the eigendecomposition of general three-dimensional image data sets that are parameterized by three different parameters. It was shown that the three-dimensional frequency properties of these image data sets can be used to extend one of the fastest known one-dimensional eigendecomposition algorithm, proposed by Chang, to such image data sets. In particular, two important extensions were discussed, i.e., (1) efficient computation of the multiplication of the image data matrix with three-dimensional frequency combinations using FFT techniques, and (2) optimum ordering of the frequency combinations based on the ranges of the three-dimensional parameters that parameterize the given image data sets. These extensions were successfully implemented to realize a computationally efficient eigendecomposition algorithm, which performed very well on several three-dimensional image data sets with different parameterizations. This eigendecomposition

Table 4 Performance of the proposed algorithm on image data set 1 for artificial objects (all times are in seconds)

Object	p	k	k^*	Time required for different steps					Speed-up	Subspace
				1	2	3	4	Total	Factor	Criterion (s)
1	238	105	99	6.53	0.024	3.23	3.73	**13.52**	7.81	0.9676
2	61	36	34	6.54	0.006	0.46	1.26	**8.27**	12.77	0.9679
3	485	183	161	6.53	0.049	10.54	6.50	**23.62**	4.47	0.9555
4	299	82	73	6.53	0.030	4.56	2.91	**14.02**	7.53	0.9508
5	625	248	222	6.53	0.063	17.43	8.77	**32.81**	3.22	0.9526
6	536	227	201	6.53	0.054	12.77	8.03	**27.39**	3.86	0.9508
7	638	209	192	6.53	0.065	17.60	7.37	**31.58**	3.34	0.9637
8	292	110	101	6.54	0.029	4.15	3.88	**14.63**	7.22	0.9556
							Average		**6.28**	**0.9581**
							Minimum		**3.22**	**0.9508**
							Maximum		**12.77**	**0.9679**

Table 5 Performance of the proposed algorithm on image data set 2 for artificial objects (all times are in seconds)

Object	p	k	k^*	Time required for different steps					Speed-up	Subspace
				1	2	3	4	Total	Factor	Criterion (s)
1	259	126	121	6.53	0.026	3.49	4.46	**14.51**	7.28	0.9750
2	73	42	40	6.54	0.007	0.64	1.49	**8.68**	12.17	0.9815
3	494	240	215	6.53	0.050	10.84	8.50	**25.92**	4.07	0.9556
4	277	101	93	6.54	0.028	4.07	3.57	**14.21**	7.43	0.9536
5	641	309	277	6.54	0.065	17.56	10.95	**35.18**	3.00	0.9538
6	541	261	235	6.53	0.054	12.98	9.27	**28.92**	3.65	0.9532
7	646	250	228	6.54	0.065	17.84	8.94	**33.43**	3.16	0.9593
8	307	129	122	6.54	0.031	4.77	4.64	**16.01**	6.60	0.9686
							Average		**5.93**	**0.9626**
							Minimum		**3.00**	**0.9532**
							Maximum		**12.17**	**0.9815**

Table 6 Performance of the proposed algorithm on image data set 3 for artificial objects (all times are in seconds)

Object	p	k	k^*	Time required for different steps					Speed-up	Subspace
				1	2	3	4	Total	Factor	Criterion (s)
1	626	194	184	38.20	0.06	16.77	64.20	**119.23**	7.23	0.9712
2	151	73	70	40.97	0.01	2.05	7.65	**50.70**	17.00	0.9871
3	898	351	320	39.40	0.33	85.66	37.00	**162.40**	5.31	0.9529
4	577	147	139	40.16	0.14	14.33	25.50	**80.15**	10.75	0.9727
5	1,236	457	425	40.67	0.41	134.22	48.59	**223.91**	3.85	0.9631
6	1,044	418	382	39.91	0.15	113.41	66.14	**219.62**	3.92	0.9566
7	1,333	392	373	39.55	0.46	146.36	42.32	**248.71**	3.47	0.9786
8	736	185	175	40.37	0.08	23.02	36.97	**124.56**	6.92	0.9642
							Average		**7.31**	**0.9683**
							Minimum		**3.47**	**0.9529**
							Maximum		**17.00**	**0.9871**

Table 7 Performance of the proposed algorithm on image data set 1 for real objects (all times are in seconds)

Object	p	k	k^*	Time required for different steps					Speed-up	Subspace
				1	2	3	4	Total	Factor	Criterion (s)
1	351	154	132	4.35	0.03	6.25	4.07	**14.72**	3.82	0.9320
2	259	108	91	4.35	0.02	3.51	2.86	**10.76**	5.23	0.9332
3	245	89	76	4.33	0.02	3.44	2.30	**10.11**	5.56	0.9220
4	56	12	11	4.35	0.00	0.38	0.31	**5.06**	11.12	0.9468
5	115	44	36	4.35	0.01	1.52	1.13	**7.03**	8.03	0.9262
6	288	93	79	4.35	0.02	4.45	2.41	**11.25**	5.00	0.9276
7	208	84	75	4.36	0.02	2.70	2.18	**9.27**	6.07	0.9261
8	134	28	26	4.35	0.01	1.55	0.73	**6.66**	8.44	0.9261
9	22	5	4	4.35	0.00	0.07	0.13	**4.56**	12.33	0.9930
10	94	13	13	4.36	0.01	1.03	0.35	**5.75**	9.78	0.9756
11	221	61	56	4.35	0.02	3.11	1.60	**9.10**	6.18	0.9449
12	207	58	50	4.36	0.05	2.71	1.50	**8.64**	6.51	0.9260
13	194	61	55	4.36	0.01	2.32	1.61	**8.33**	6.75	0.9300
14	67	14	14	4.36	0.00	0.54	0.37	**5.29**	10.63	0.9375
15	21	5	5	4.37	0.00	0.06	0.13	**4.57**	12.31	0.9812
16	184	41	38	4.35	0.01	2.43	1.10	**7.91**	7.11	0.9530
							Average		**7.80**	**0.9426**
							Minimum		**3.82**	**0.9220**
							Maximum		**12.33**	**0.9930**

Table 8 Performance of the proposed algorithm on image data set 2 for real objects (all times are in seconds)

Object	p	k	k^*	Time required for different steps					Speed-up	Subspace
				1	2	3	4	Total	Factor	Criterion (s)
1	1,215	346	301	16.43	0.31	76.50	33.39	**126.72**	5.85	0.9338
2	874	205	182	16.42	0.22	35.67	19.79	**71.95**	10.29	0.9367
3	823	198	170	16.49	0.21	31.77	19.13	**67.59**	10.96	0.9257
4	159	25	24	16.38	0.04	2.25	2.43	**21.11**	35.10	0.9537
5	391	96	84	16.45	0.10	7.12	9.30	**32.99**	22.46	0.9300
6	975	169	147	16.55	0.25	45.31	16.31	**78.43**	9.45	0.9286
7	680	135	123	16.39	0.17	20.35	13.02	**49.94**	14.84	0.9423
8	422	64	57	16.39	0.10	8.05	6.18	**30.74**	24.11	0.9213
9	53	7	7	16.40	0.01	0.35	0.67	**17.43**	42.51	0.9778
10	296	26	25	16.39	0.07	4.24	2.51	**23.24**	31.88	0.9533
11	717	142	128	16.45	0.18	22.76	13.69	**53.09**	13.96	0.9449
12	706	140	120	16.40	0.18	21.94	13.54	**52.07**	14.03	0.9304
13	629	114	105	16.40	0.16	17.56	11.02	**45.15**	16.41	0.9358
14	184	28	25	16.48	0.04	2.41	2.70	**21.65**	34.23	0.9355
15	55	9	9	16.41	0.01	0.37	0.87	**17.68**	41.91	0.9909
16	596	89	84	16.43	0.15	15.32	8.59	**40.51**	18.29	0.9536
							Average		**21.64**	**0.9433**
							Minimum		**5.85**	**0.9213**
							Maximum		**42.51**	**0.9909**

algorithm can be used for computationally efficient three-dimensional pose estimation of objects.

Acknowledgments This work was supported in part by the National Imagery and Mapping Agency under contract no. NMA201-00-1-1003, through collaborative participation in the Robotics Consortium sponsored by the US Army Research Laboratory under the Collaborative Technology Alliance Program, Cooperative Agreement DAAD19-01-2-0012, and the Missile Defense Agency under the contract no. HQ0006-05-C-0035. Approved for Public Release 07-MDA-2783 (26 SEPT 07). The US Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the US Government. A preliminary version of portions of this work was presented at the IEEE Southwest Symposium on Image Analysis and Interpretation held at Denver, CO, USA, March 26–28, 2006.

References

1. Fukunaga K (1990) Introduction to statistical pattern recognition. Academic Press, London
2. Martinez AM, Kak AC (2001) PCA versus LDA. IEEE Trans PAMI 23(2):228–233
3. Sirovich L, Kirby M (1987) Low-dimensional procedure for the characterization of human faces. J Opt Soc Am 4(3):519–524
4. Kirby M, Sirovich L (1990) Application of the Karhunen–Loeve procedure for the characterization of human faces. IEEE Trans PAMI 12(1):103–108
5. Turk M, Pentland A (1991) Eigenfaces for recognition. J Cogn Neurosci 3(1):71–86
6. Belhumeur PN, Hespanha JP, Kriegman DJ (1997) Eigenfaces vs. fisherfaces: recognition using class specific linear projection. IEEE Trans PAMI 19(7):711–720
7. Brunelli R, Poggio T (1993) Face recognition: features versus templates. IEEE Trans PAMI 15(10):1042–1052
8. Pentland A, Moghaddam B, Starner T (1994) View-based and modular eigenspaces for face recognition. In: Proc of the IEEE comp soc conf on computer vision and pattern recognition. Seattle, WA, USA, Jun 21–23, pp. 84–91
9. Yang MH, Kriegman DJ, Ahuja N (2002) Detecting faces in images: a survey. IEEE Trans PAMI 24(1):34–58
10. Murase H, Sakai R (1996) Moving object recognition in eigenspace representation: gait analysis and lip reading. Pattern Recognit Lett 17(2):155–162
11. Chiou G, Hwang JN (1997) Lipreading from color video. IEEE Trans Image Process 6(8):1192–1195
12. Murase H, Nayar SK (1994) Illumination planning for object recognition using parametric eigenspaces. IEEE Trans PAMI 16(12):1219–1227
13. Huang CY, Camps OI, Kanungo T (1997) Object recognition using appearance-based parts and relations. In: Proc of the IEEE comp soc conf on computer vision and pattern recognition. San Juan, PR, USA, 17–19 June 1997, pp 877–883
14. Campbell RJ, Flynn PJ (1999) Eigenshapes for 3D object recognition in range data. In: Proc of the IEEE comp soc conf on computer vision and pattern recognition. Fort Collins, CO, USA, 23–25 June 1999, pp 505–510
15. Jogan M, Leonardis A (2000) Robust localization using eigenspace of spinning-images. In: Proc of the IEEE workshop on omnidirectional vision. Hilton Head Island, South Carolina, USA, pp 37–44
16. Yoshimura S, Kanade T (1994) Fast template matching based on the normalized correlation by using multiresolution eigenimages. In: 1994 IEEE workshop on motion of non-rigid and articulated objects. Austin, Texas, 11–12 November 1994, pp 83–88
17. Winkeler J, Manjunath BS, Chandrasekaran S (1999) Subset selection for active object recognition. In: Proc of the IEEE comp soc conf on computer vision and pattern recognition. Fort Collins, Colorado, USA, 23–25, pp 511–516
18. Nayar SK, Murase H, Nene SA (1994) Learning, positioning, and tracking visual appearance. In: Proc of the IEEE int conf on robot automat. San Diego, CA, USA, 8–13 May 1994, pp 3237–3246
19. Black MJ, Jepson AD (1998) Eigentracking: robust matching and tracking of articulated objects using a view-based representation. Int J Comput Vis 26(1):63–84
20. Murase H, Nayar SK (1995) Visual learning and recognition of 3-D objects from appearance. Int J Comput Vis 14(1):5–24
21. Murase H, Nayar SK (1997) Detection of 3D objects in cluttered scenes using hierarchical eigenspace. Pattern Recognit Lett 18(4): 375–384
22. Nayar SK, Nene SA, Murase H (1996) Subspace method for robot vision. IEEE Trans Robot Automat 12(5):750–758
23. Moghaddam B, Pentland A (1997) Probabilistic visual learning for object representation. IEEE Trans PAMI 19(7):696–710
24. Stewart GW (1973) Introduction to matrix computation. Academic, New York
25. Shlien S (1982) A method for computing the partial singular value decomposition. IEEE Trans PAMI 4(6):671–676
26. Haimi-Cohen R, Cohen A (1987) Gradient-type algorithms for partial singular value decomposition. IEEE Trans PAMI 9(1):137–142
27. Yang X, Sarkar TK, Arvas E (1089) A survey of conjugate gradient algorithms for solution of extreme eigen-problems for a symmetric matrix. IEEE Trans ASSP 37(10):1550–1556
28. Vogel CR, Wade JG (1994) Iterative SVD-based methods for ill-posed problems. SIAM J Sci Comput 15(3):736–754
29. Murakami H, Kumar V (1982) Efficient calculation of primary images from a set of images. IEEE Trans PAMI 4(5):511–515
30. Chandrasekaran S, Manjunath B, Wang Y, Winkeler J, Zhang H (1997) An eigenspace update algorithm for image analysis. CVGIP: graphic models and image processing 59(5):321–332
31. Murase H, Lindenbaum M (1995) Partial eigenvalue decomposition of large images using the spatial temporal adaptive method. IEEE Trans Image Process 4(5):620–629
32. Chang CY, Maciejewski AA, Balakrishnan V (2000) Fast eigenspace decomposition of correlated images. IEEE Trans Image Process 9(11):1937–1949
33. Saitwal K, Maciejewski AA, Roberts RG, Draper BA (2006) Using the low-resolution properties of correlated images to improve the computational efficiency of eigenspace decomposition. IEEE Trans Image Process 15(8):2376–2387
34. Davis PJ (1979) Circulant matrices. Wiley, New York
35. Saitwal K (2006) Fast eigenspace decomposition of correlated images using their spatial and temporal properties. PhD Dissertation, Colorado State University, USA