Hierarchical Tensor Approximation of Multi-Dimensional Visual Data

Qing Wu, Tian Xia, Chun Chen, Hsueh-Yi Sean Lin, Hongcheng Wang, Yizhou Yu

***Abstract*— Visual data comprise of multi-scale and inhomogeneous signals. In this paper, we exploit these characteristics and develop a compact data representation technique based on a hierarchical tensor-based transformation. In this technique, an original multi-dimensional dataset is transformed into a hierarchy of signals to expose its multi-scale structures. The signal at each level of the hierarchy is further divided into a number of smaller tensors to expose its spatially inhomogeneous structures. These smaller tensors are further transformed and pruned using a tensor approximation technique. Our hierarchical tensor approximation supports progressive transmission and partial decompression. Experimental results indicate that our technique can achieve higher compression ratios and quality than previous methods, including wavelet transforms, wavelet packet transforms, and single-level tensor approximation. We have successfully applied our technique to multiple tasks involving multi-dimensional visual data, including medical and scientific data visualization, data-driven rendering and texture synthesis.**

***Index Terms*— Multilinear Models, Multidimensional Image Compression, Hierarchical Transformation, Tensor Ensemble Approximation, Progressive Transmission, Texture Synthesis**

I. Introduction

With advances in imaging technologies—-such as CCD, laser, magnetic resonance, and diffusion tensor—-and physically based solid and fluid simulation technologies, new visual data of multiple dimensions have been produced at an unprecedented rate and scale. These new technologies bring new challenges to existing visual data modeling and processing techniques. One of the fundamental and challenging problems is how to efficiently represent, analyze and visualize such a vast and ever-growing amount of visual data.

Visual data exhibit two important intertwined characteristics. First, they comprise of signals at many different scales or frequencies. For example, we can decompose a composite signal into a series of cosine waves using the Fourier transform. Such a decomposition in the frequency domain represents the original signal as a superposition of simpler elementary components at distinct frequencies. Second, these signals have spatially inhomogeneous magnitudes. If we adaptively subdivide the spatial domain, it is possible to approximate the original signal as a piecewise smooth function which has a more or less uniform magnitude over each local region. Existing techniques, such as wavelet transforms, have successfully exploited both of the aforementioned characteristics to achieve a transformation that is local in both frequency and space domains. As a result of such decomposition and transformation, the inherent structures of the original signal become better exposed to compression. One important aspect of such inherent structures is that even though the original signal appears inhomogeneous, its elementary components at different frequencies or local regions exhibit correlation [1]. Further operations performed on these components can remove redundancy and achieve compact representation. There are at least two possible operations we can perform. First, correlated components can be grouped together and represented collectively in a lower dimensional space to remove redundancy. Second, a component can be simply pruned if its magnitude is negligible. Such processing gives rise to significantly reduced size and dimensionality of a dataset and overall, reduced computational and representational complexity.

Multilinear models based on tensor approximation have received much attention recently. They are capable of generating a more compact representation of multi-dimensional data than traditional dimensionality reduction methods. In this paper, we exploit the aforementioned characteristics of visual data and develop an analysis and representation technique based on

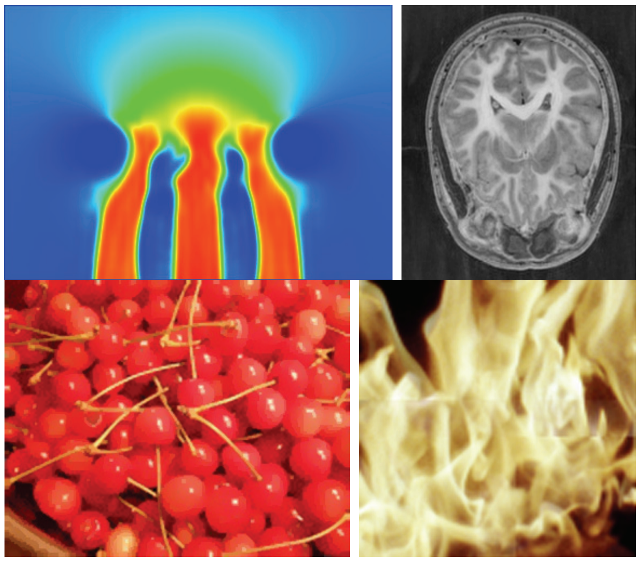


Fig. 1. Examples of visual data our hierarchical tensor approximation can be applied to.

Q. Wu, T. Xia, and Y. Yu are currently with Department of Computer Science, University of Illinois at Urbana-Champaign, 201 North Goodwin Ave, Urbana, IL 61801.

C. Chen is with College of Computer Science, Zhejiang University, Hangzhou, 310027, China.

H.-Y. Lin participated in this work while he was visiting University of Illinois at Urbana-Champaign from Department of Computer Science, National Chiao-Tung University, Hsinchu 300, Taiwan.

H. Wang is currently with United Technologies Research Center, 411 Silver Ln MS 129-15, East Hartford, CT 06066. He participated in this work while he was a PhD candidate at University of Illinois at Urbana-Champaign.

a hierarchical tensor-based transformation. In this technique, an original multi-dimensional dataset is transformed into a hierarchy of signals to expose its multi-scale structures. The signal at each level of the hierarchy is further divided into a number of tensors with smaller spatial support to expose its spatially inhomogeneous structures. These smaller tensors are further transformed and pruned using a tensor approximation technique to achieve a highly compact representation.

Our hierarchical tensor approximation has two significant advantages. First, it can achieve far higher quality than wavelet transforms at large compression ratios. In comparison to a traditional multiresolution analysis which simply projects signals at various different resolutions onto a prescribed basis which was obtained without any specific knowledge of the data, our hierarchical approximation actually adopts bases specifically tailored for the characteristics of the data currently being approximated. Our high-level approach is thus consistent with a rich line of research on basis pursuit that seeks to find a dictionary which is adapted to the particular signal at hand [2]. Second, this hierarchical tensor-based representation facilitates progressive or partial data transmission and visualization. By transmitting the compressed data level by level, the receiver can quickly view the low resolution versions first and decide whether it is worthwhile to wait for higher resolution details. In addition, if visualization only concerns a portion of the original dataset, the subdivided tensors at each level support partial transmission and decompression and hence faster response time to user requests.

We have successfully applied our new hierarchical tensor approximation in multiple tasks, including medical and scientific data visualization, data-driven rendering and texture synthesis.

II. Background and Related Work

A real Nth-order tensor $\mathcal{A} \in \Re^{n_1 \times n_2 \times \ldots \times n_N}$, can be considered as an element of a composite vector space, $R^{n_1} \otimes R^{n_2} \otimes \cdots \otimes R^{n_N}$, where we call each R^{n_i} an elementary vector space, and $\otimes$ denotes the Kronecker product of vector spaces. The dimensionality of the i-th elementary vector space is n_i. Let us first review basic tensor approximation techniques, including rank-r approximation and rank-$(r_1, r_2, ..., r_N)$ approximation.

A rank-r approximation of $\mathcal{A}$ is formulated as

$$\hat{\mathcal{A}} = \sum_{j=1}^{r} b_j \times_1 \mathbf{u}_j^{(1)} \times_2 \mathbf{u}_j^{(2)} \times \cdots \times_N \mathbf{u}_j^{(N)}, \qquad (1)$$

where b_j is a scalar coefficient, each $\mathbf{u}_j^{(i)}$ is simply a column vector of length n_i, and $\times_k$ represents k-mode product of a tensor by a matrix [*]. The column vectors, $\{\mathbf{u}_j^{(i)}\}_{j=1}^{r}$, are not necessarily orthogonal to each other. It is possible to devise a simple greedy algorithm to suboptimally solve the basis vectors in a sequential order. A more efficient algorithm for rank-r approximation can be found in [3]. When r is small, the scalar coefficients along with their associated basis vectors give rise to a compact representation of the original tensor.

A rank-$(r_1, r_2, ..., r_N)$ approximation of $\mathcal{A}$ is formulated as

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

where each basis matrix $\mathbf{U}^{(i)} \in \Re^{n_i \times r_i}$, and the core tensor $\mathcal{B} \in \Re^{r_1 \times r_2 \times \cdots \times r_N}$. The column vectors of each $\mathbf{U}^{(i)}$ are orthonormal to each other. Once the basis matrices are known, $\mathcal{B} = \mathcal{A} \times_1 \mathbf{U}^{(1)^T} \times_2 \mathbf{U}^{(2)^T} \times \cdots \times_N \mathbf{U}^{(N)^T}$. When $r_1, r_2, \ldots, r_N$ are sufficiently small, the core tensor and the basis matrices together give rise to a compact representation. The Alternative Least Square (ALS) algorithm was used in [4], [5] to solve the optimal basis matrices given their reduced ranks. In each iteration, ALS optimizes only one of the basis matrices, while keeping others fixed. A closely related concept is N-mode SVD [6] which also decomposes a tensor into a series of products as shown in (2). However, the resulting basis matrices in N-mode SVD are not truncated and therefore, retain their original ranks.

Tensor algebra has received much attention in computer graphics and computer vision. The principle of rank-r approximation has been applied to image coding and classification in [7]. Concurrent rank-$(r_1, r_2, ..., r_N)$ approximation has been applied to similar problems in [8]. In general, the orthogonality of the basis matrices enables rank-$(r_1, r_2, \cdots, r_N)$ tensor approximation to remove redundancy among different modalities more effectively than rank-r approximation. N-mode SVD and tensor approximation have been applied to 2D face recognition [9] and facial expression decomposition [10]. Such an approach has been further generalized to 3D face modeling and transfer in [11]. Tensor-based multilinear modeling has also been applied to bidirectional texture functions in [12], [13], [14]. Rank-r approximation was used in [12] while rank-$(r_1, r_2, \cdots, r_N)$ approximation was applied in [13], [14]. It has been demonstrated in [14] that rank-$(r_1, r_2, \cdots, r_N)$ tensor approximation can achieve smaller Root Mean Squared Errors (RMSE) than Principal Component Analysis (PCA) given the same compression ratios.

It has been recently demonstrated that non-negative tensor factorization [15] based on rank-r approximation can outperform non-negative matrix factorization [16] for sparse image decomposition. Rank-$(r_1, r_2, ..., r_N)$ tensor approximation has also been generalized to multilinear clustering [17]. The clustering algorithm in [17] first performs rank-$(r_1, r_2, ..., r_N)$ tensor approximation using a criterion based on pairwise distance, followed by K-means clustering on the transformed data. Recently, a related technique called clustered tensor approximation has been developed and applied to precomputed radiance transfer in graphics [18]. Note that most previous applications of tensor approximation consider the input data as a single-level or single-resolution multi-dimensional array without exploiting its inhomogeneous multi-scale structures. The only exception that does multi-scale subdivision is the sparse matrix approximation method based on $\mathcal{H}$-matrices [19]. This method has been recently applied to compact visual data representation and acquisition for computer graphics [20]. We will compare our hierarchical approximation method with $\mathcal{H}$-matrices [19] in Section VI.

[*]The k-mode product of a tensor $\mathcal{A}$ by a matrix $\mathbf{U} \in^{J_k \times n_k}$, denoted by $\mathcal{A} \times_k \mathbf{U}$, is defined as a tensor with entries: $(\mathcal{A} \times_k \mathbf{U})_{i_1 \ldots i_{k-1} j_k i_{k+1} \cdots i_N} = \sum_{i_k} a_{i_1 \ldots i_N} u_{j_k i_k}$.

On the other hand, wavelet analysis is inherently a multi-scale analysis tool and has been frequently applied to visual data compression [21], [22], [23], [24]. Such wavelet-based compression only recursively decomposes the low-frequency components at each scale onto the coarser levels. Efforts have been made to recursively decompose both the low-frequency and high-frequency components at each scale [25], [26]. Such a wavelet packet technique constructs an over-complete collection of wavelet bases, and then chooses a subset of the bases that most compactly approximate the input signal. Even though these bases are adaptively obtained, they are still convolutions of input signals(coefficients) with prescribed filters. Therefore, unlike the bases in tensor approximation, they are only scale-adaptive. When wavelets are applied to multi-dimensional signals, the bases are typically formed as tensor products of the one-dimensional bases. As a result, the multi-dimensional bases are axis-aligned. There has been much work on developing more powerful oriented wavelet bases for multi-dimensional spaces [27], [28], [29]. However, such bases are still prescribed filters that cannot be adapted to specific data, and the gained compression efficiency over axis-aligned bases is limited.

III. Tensor Ensemble Approximation

In many situations, we need to simultaneously approximate an ensemble of tensors, and most often, these tensors have strong correlations. For example, a multi-dimensional array of color values or velocity vectors gives rise to three scalar tensors for the three color channels or three components of the vectors. As we know, color response curves have much overlap with each other and velocity components need to satisfy certain physics-based equations. As a result, these scalar tensors have strong correlations with each other. As will be discussed in the next section, we also subdivide a large tensor into smaller ones and approximate them collectively because these subdivided tensors have local spatial support and may share similar basis matrices among each other.

Suppose the list of tensors that need to be approximated are $\mathcal{A}_1, \mathcal{A}_2, \cdots, \mathcal{A}_m$, where m is the number of tensors and $\mathcal{A}_i \in \Re^{n_1 \times n_2 \times \ldots \times n_N}$, and we look for a rank-$(r_1, r_2, ..., r_N)$ approximation of each $\mathcal{A}_i$, which is denoted as $\tilde{\mathcal{A}}_i$. Because of correlations and redundancies among this list of tensors, approximating each of them separately is suboptimal. We move one step further and approximate all these tensors collectively. To achieve this goal, we organize these m_l N-th order tensors into a $(N+1)$-th order tensor $\mathcal{G} \in \Re^{n_1 \times n_2 \times \ldots \times n_N \times m}$, and obtain a rank-$(r_1, r_2, ..., r_N, r_{N+1})$ tensor $\tilde{\mathcal{G}}$ as its approximation using the ALS algorithm. Note that $r_{N+1} \leq m$. This approximation is compactly represented using $N+1$ basis matrices, $\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(N)}, \mathbf{U}^{(N+1)}$, and a core tensor $\mathcal{H}$. That is,

$$\tilde{\mathcal{G}} = \mathcal{H} \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times \cdots \times_N \mathbf{U}^{(N)} \times_{(N+1)} \mathbf{U}^{(N+1)}, \quad (3)$$

where $\mathbf{U}^{(1)} \in \Re^{n_1 \times r_1}$, $\cdots$, $\mathbf{U}^{(N)} \in \Re^{n_N \times r_N}$ and $\mathbf{U}^{(N+1)} \in \Re^{m \times r_{N+1}}$ and $\mathcal{H} \in \Re^{r_1 \times r_2 \times \cdots \times r_N \times r_{N+1}}$. When necessary, it is actually quite convenient to extract the core tensor $\mathcal{B}_i$ of each N-th order subtensor $\tilde{\mathcal{A}}_i$ out of this ensemble representation.

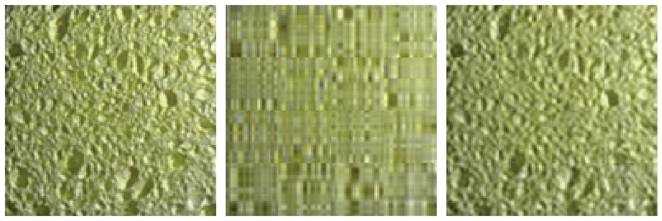

(a) Original (b) Noncollective PSNR 20.13 (c) Collective PSNR 26.17

Fig. 2. A comparison of a reconstructed SPONGE texture from both collective and non-collective tensor approximations. (a) Original image, (b) a reconstructed image from the non-collective approximation, (c) a reconstructed image from our collective approximation. (b)&(c) share the same compression rate which is 87.5%.

Let $\mathbf{u}_i^{(N+1)}$ be the vector representing the transposed i-th row of $\mathbf{U}^{(N+1)}$. Then,

$$\mathcal{B}_i = \mathcal{H} \times_{(N+1)} \mathbf{u}_i^{(N+1)^T}. \quad (4)$$

We have compared our tensor ensemble approximation against individual tensor approximation. Fig. 2 shows one of such comparisons on texture images. In this example, the original image is partitioned into 16 blocks each of which has three color channels. Our ensemble approximation models the data as a list of 48 subtensors, approximates them collectively, and achieves a peak signal-to-noise ratio (PSNR) [†] of 26.17 at 87.5% compression rate. On the other hand, individual approximation needs to store a distinct set of basis matrices for each color channel and each subtensor even when these bases are similar, and can only achieve a PSNR of 20.13 at the same compression rate. Such a difference is primarily caused by the excessive basis overhead in individual approximation.

IV. Hierarchical Tensor Approximation

Given a collection of multi-dimensional datasets with the same size and dimensionality, our multilevel approximation algorithm produces a compact hierarchical representation based on tensor approximation by removing the redundancies among different datasets as well as within each dataset. In this section, we first introduce a lossless hierarchical transformation of multi-dimensional matrices, or tensors. This transformation decomposes the original data into multiple levels and removes the redundancy at each level by exploiting the correlation among different spatial regions. To exploit spatial inhomogeneity of the original data, further lossy approximation (quantization and pruning) is performed on the resulting multilevel data. These two steps together give rise to a very compact representation.

A. Hierarchical Transformation of Tensors

The basic idea of hierarchical transformation is to first recursively partition the entire domain into smaller blocks and define a truncated tensor-product basis for each block. Every point in the domain is then covered by a series of blocks

[†]PSNR = $20 \log_{10} \frac{\text{Range of Signal}}{\text{RMSE}}$, where RMSE stands for Root Mean Squared Errors.

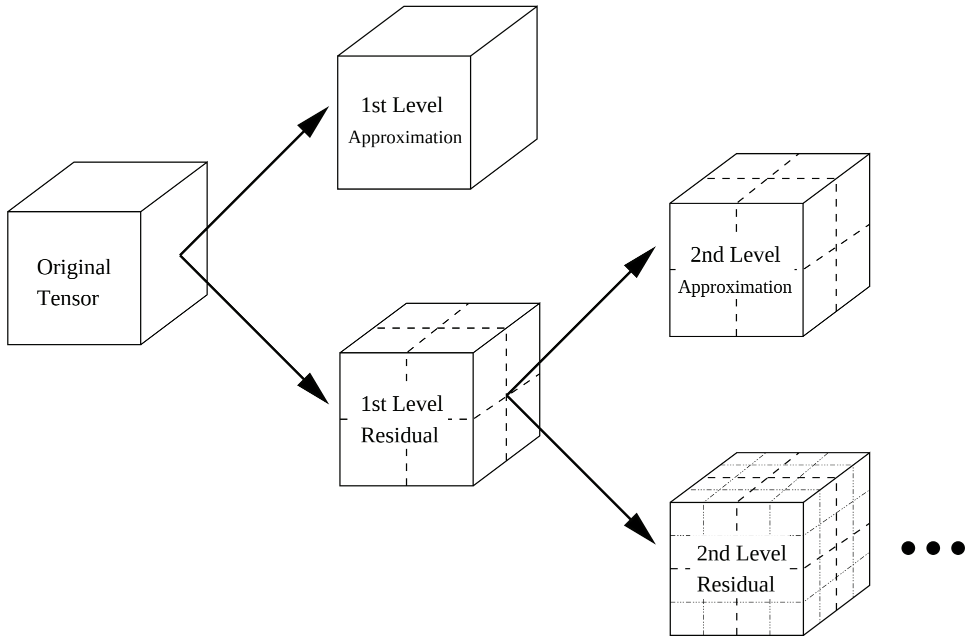


Fig. 3. In our hierarchical tensor transformation, an original tensor is represented as the summation of incomplete tensor approximations at multiple levels. The tensors at each level are subdivided residual tensors passed from the higher level.

with increasingly larger scales. The basis over each of these blocks can provide a partial approximation of the data item at that point. The summation of these partial approximations is the actual approximation of the data item. Let the input dataset be a multidimensional array defined by the tensor, $\mathcal{A} \in \Re^{n_1 \times n_2 \times \ldots \times n_N}$, where we assume $n_i = 2^{k_i}$. Let the set of indices for the i-th mode be $I_i = [0, 1, \ldots, n_i - 1]$. The domain of the input dataset is then defined as the Cartesian product of these sets of indices, $D = I_0 \times I_1 \times \cdots \times I_N$. We perform recursive binary partition over each index set so that the original dataset is at level 0, and at level l, each index set has been partitioned into 2^l subsets. For example, $I^l_{i,j} = [j2^{k_i - l}, \ldots, (j+1)2^{k_i - l} - 1]$ represents the j-th subset of the i-th mode at level l. As a result, at level l, the original domain is subdivided into 2^{Nl} blocks. The block $D^l_{[j_1, j_2, \ldots, j_N]}$ represents $I^l_{1,j_1} \times I^l_{2,j_2} \times \ldots \times I^l_{N,j_N}$.

We define a common set of truncated basis matrices for the blocks at each level. The basis matrices for the blocks at level l are denoted as $\mathbf{U}^{l,(1)}, \mathbf{U}^{l,(2)}, \ldots, \mathbf{U}^{l,(N)}$, where $\mathbf{U}^{l,(i)} \in \Re^{2^{k_i - l} \times r^l_i}$ with $r^l_i \le 2^{k_i - l}$. We further define a tensor $\mathcal{P}^l_{[j^l_1, j^l_2, \ldots, j^l_N]} \in \Re^{2^{k_1 - l} \times 2^{k_2 - l} \times \ldots \times 2^{k_N - l}}$ over the block $D^l_{[j^l_1, j^l_2, \ldots, j^l_N]}$ at level l. There is a core tensor $\mathcal{Q}^l_{[j^l_1, j^l_2, \ldots, j^l_N]} \in \Re^{r^l_1 \times r^l_2 \times \ldots \times r^l_N}$ so that

$$\mathcal{P}^l_{[j^l_1, j^l_2, \ldots, j^l_N]} = \mathcal{Q}^l_{[j^l_1, j^l_2, \ldots, j^l_N]} \times_1 \mathbf{U}^{l,(1)} \times_2 \mathbf{U}^{l,(2)} \times \cdots \times_N \mathbf{U}^{l,(N)}. \quad (5)$$

Let $\mathcal{A}(a_1, a_2, \ldots, a_N)$ be an element of the original tensor $\mathcal{A}$ defined by the indices $a_1, a_2, \ldots, a_N$ with $0 \le a_i < n_i, 1 \le i \le N$. We can form an approximation of this element using the aforementioned core tensors and basis matrices at different levels. There is only one core tensor from each level in this approximation. The approximated element can be expressed as

$$\tilde{\mathcal{A}}(a_1, a_2, \ldots, a_N) = \sum_{l=0}^{L} \mathcal{P}^l_{[b^l_1, b^l_2, \ldots, b^l_N]}(c^l_1, c^l_2, \ldots, c^l_N), \quad (6)$$

where $b^l_i = \lfloor a_i / 2^{k_i - l} \rfloor$ and $c^l_i = a_i \bmod 2^{k_i - l}$ for $1 \le i \le N$.

If the ranks of the basis matrices at all levels, $\{r^l_i\}^{L,N}_{l=0,i=1}$, are given, the basis matrices and core tensors at all levels can be potentially solved by minimizing the following summed squared errors

$$\sum_{a_1, a_2, \ldots, a_N} \|\tilde{\mathcal{A}}(a_1, a_2, \ldots, a_N) - \mathcal{A}(a_1, a_2, \ldots, a_N)\|^2. \quad (7)$$

However, data compression is in general a more complicated problem than approximation because compression needs to deal with two conflicting goals, reducing approximation errors while achieving higher compression ratios. Furthermore, relative importance of these two goals changes for different applications, and thus different objective functions can be formulated. For example, one objective could be minimizing the approximation errors when the compression ratio has a lower bound; or achieving the highest compression ratio when the PSNR has a lower bound. Since the ranks of the basis matrices are directly related to the compression ratio, they need to be adjusted as well.

Directly minimizing (7) with respect to the ranks and basis matrices of the subdivided tensors at all levels would be extremely computationally expensive if possible at all. Therefore, we take a greedy approach and construct the basis matrices of the subdivided tensors level by level from top to bottom. The original input tensors are placed at the top level, which is also the first level. At each level l, there is an initial list of N-th order tensors, $\mathcal{A}^l_1, \mathcal{A}^l_2, \cdots, \mathcal{A}^l_{m_l}$. We exploit correlation among these tensors and remove redundancy by performing tensor ensemble approximation as discussed in the previous

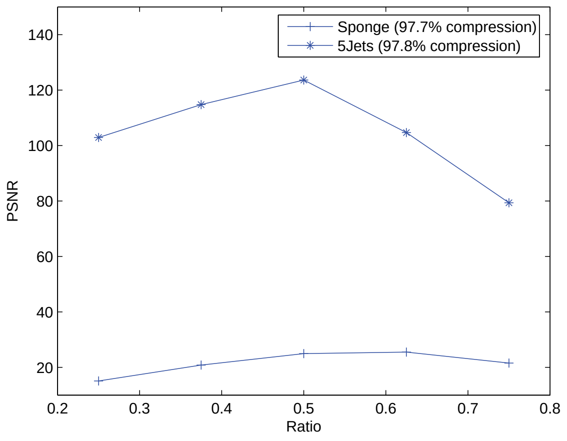


Fig. 4. The dependency of PSNR on the common ratio among ranks at different levels. Experimental results for two datasets are shown here. A fixed compression ratio is used for each dataset. These results confirm that a common ratio of 0.5 is (near) optimal.

section. We only perform an incomplete approximation in the sense that the ranks of the truncated basis matrices are set to be smaller than necessary and the residual error is not necessarily reduced to the desired level [‡] . From this incomplete approximation, we can obtain an approximated version, $\tilde{\mathcal{A}}_i^l$, of each tensor $\mathcal{A}_i^l$. A residual tensor, $\mathcal{E}_i^l = \mathcal{A}_i^l - \tilde{\mathcal{A}}_i^l$, is subsequently defined for each tensor. Each residual tensor is then subdivided into up to 2^N smaller tensors by dividing the index set of each mode in half unless an index set has only one element. These subdivided residual tensors are passed to the next lower level in the hierarchy for further approximation. Such tensor subdivision and approximation can be repeated until all index sets have only one element. It can be easily verified that the original tensors at the top level can be faithfully reconstructed by first reconstructing the (residual) tensors at each level from their corresponding core tensors and basis matrices, and then accumulating the tensors at all levels together (Fig. 3).

Meanwhile, in our hierarchical transformation, we need to determine the ranks of the basis matrices at each level. One potential solution would be a global nonlinear optimization with the ranks at all levels as unknowns. Such a large-scale nonlinear optimization is extremely expensive. Since there are many unknowns, the optimization is also very likely to be trapped in local minima. Inspired by the fact that the size of the residual tensors at different levels follows a geometric progression, we have designed a relatively efficient scheme that achieves a suboptimal solution. At the first level, this scheme chooses the set of desired ranks, $r_1^1, r_2^1, ..., r_N^1$, for the basis matrices either automatically or under user guidance. At subsequent levels, each of the ranks follows a geometric progression. Though not optimal, this scheme significantly reduces the number of adjustable parameters and has been very effective in our experiments. The common ratio we used for the geometric progression was always set to 0.5. This is because with this common ratio, the size of the core tensors at different levels follows the same geometric progression as the size of the original residual tensors and, thus, the compression ratio achieved at each individual level remains approximately the same, which further makes it possible to automatically provide a rough estimation of the ranks at the top level once a desired compression ratio is given. Experimental results shown in Fig. 4 confirm that a common ratio of 0.5 is at least near optimal.

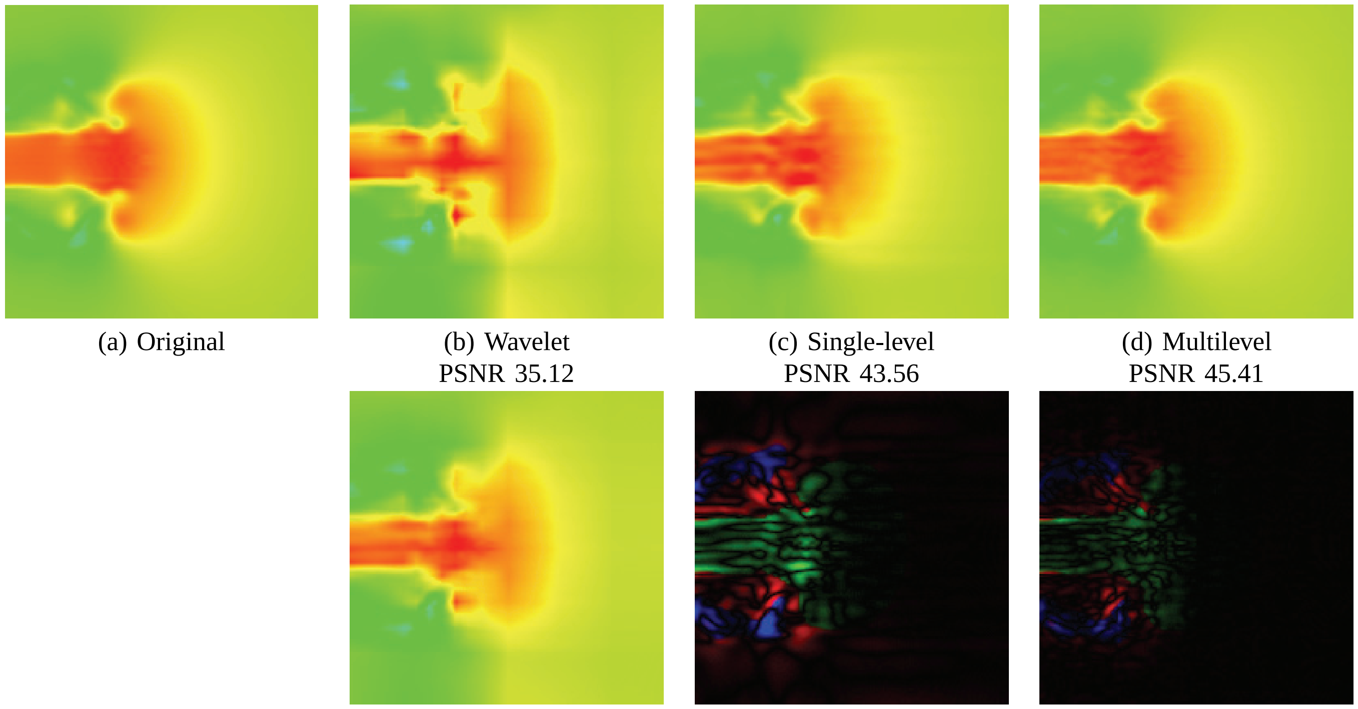
[‡]A possibly large local error at one hierarchy level is likely to be well captured and compensated at the next finer level since the magnitude of the residual will influence the choice of the best basis on the finer level

B. Nonlinear Approximation

To achieve a more compact representation, we need to perform further lossy approximation of the original data from the above hierarchical transformation. We achieve this goal by performing uniform quantization on the core tensor coefficients followed by a tensor pruning step. Coefficients with a magnitude smaller than the quantization step are set to zero. The elements of the basis matrices are also uniformly quantized. In our experiments, we always use 8 bits per element for the basis matrices, and 8-20 bits per coefficient for the core tensors. After quantization, we further perform a pruning step on core tensors by introducing a separate pruning threshold which can simply be zero. For each core tensor $\mathcal{B}_i^l$ defined in (4), we compute the square of each coefficient and obtain the summation of these squared coefficients. If the summation is less than the pruning threshold, the entire core tensor is eliminated. If the pruning threshold is set to zero, a core tensor is eliminated only when all of its coefficients have been quantized to zero. Note that the number of residual tensors increases when we descend to lower levels of the hierarchy. Therefore, effectively eliminating entire core tensors at lower levels plays a crucial role in achieving high compression ratios using our hierarchical transformation. This tensor pruning step bears resemblance to coefficient pruning in wavelet-based image compression [21], [22]. Since the input data has spatially varying details, the coefficients of the core tensors corresponding to smooth regions of the data are likely to be small. Thus, these core tensors are more likely to be pruned. The tradeoff between the compression ratio and the peak signal-to-noise ratio (PSNR) is achieved by adjusting the tensor pruning threshold. Given a PSNR and a quantization step, we perform a search for the tensor pruning threshold that can achieve the desired PSNR. We currently do not further encode the coefficients of the remaining core tensors because the focus of this paper is on the effectiveness of hierarchical transformation and approximation in compact data representation. More importantly, applications, such as real-time rendering, only require data reduction, but not coding. In fact, coding may complicate matters in such applications since decoding consumes extra computing resources. When coding is really necessary, there are many existing techniques, such as arithmetic coding, entropy coding and zero-tree coding [22], from which one can choose.

V. Applications and Experiments

In this section, we discuss the potential applications of our hierarchical tensor approximation in multi-dimensional



(a) Original
(b) Wavelet PSNR 35.12
(c) Single-level PSNR 43.56
(d) Multilevel PSNR 45.41
(e) Wavelet Packet PSNR 39.48
(f) Residual of (c)
(g) Residual of (d)

Fig. 5. A comparison of the scalar density field of a time-varying volume dataset reconstructed from a bi-orthogonal wavelet transform, a corresponding wavelet packet transform, the single-level tensor approximation from [14] and our hierarchical tensor approximation. The resolution of the original time-varying volume data is 128x128x128. (a) A visualization of a cross section of an original volume. (b)-(e) Visualizations of three reconstructed volumes at the same cross section. The volumes in (b)-(e) were reconstructed from the wavelet transform, the single-level tensor approximation, our multilevel tensor approximation, and the wavelet packet transform, respectively. They share the same compression ratio which is 1200. The same color transfer function is applied to all four volumes in (a)-(e). Our result in (d) agrees with the original data very well. The result from the wavelet transform deviates most significantly from the original, which indicates a large RMSE. (f)&(g) show the magnified (x10) residual images of (c) and (d), respectively.

data visualization, data-driven graphics rendering and texture synthesis. We further conduct experiments and comparisons to demonstrate that our technique exhibits advantages in all these areas.

A. Multi-Dimensional Data Visualization

There has been an increasing amount of multi-dimensional medical and scientific data that need to be visualized and analyzed. Such data include 3D or 4D medical images and 4D time-varying, multivariate volume data from scientific computing. Effective data compression possibly with progressive transmission would be desired when visual data needs to be communicated between two remote hosts over a link with limited bandwidth. When the amount of original data used by an interactive application exceeds the memory capacity, it would be desired to perform computation directly using a compressed form to reduce data accessing cost. Compression based on our hierarchical tensor approximation is suited for such purposes because it can achieve high compression ratios, support progressive transmission and allow partial decompression.

To measure and compare compression performance, we conducted experiments on a 4D time-varying scientific dataset and the Visible Human dataset. The 4D time-varying dataset is a simulated volume sequence of five jets. Every frame in the sequence is a multivariate 3D volume with a scalar density and energy value, and a velocity vector at each voxel. The resolution of the volume is 128x128x128. The color cryosection images of the Visible Human dataset consists of 1871 scans of the entire body taken at 1mm intervals and amounts to 15 Gbytes. We have compared our hierarchical tensor approximation against wavelets, wavelet packet analysis, and the single-level tensor approximation technique from [14] on the velocity field of the time-varying volume dataset and a subset of the color cryosection images of the Visible Human dataset. We construct multi-dimensional wavelet bases using one-dimensional biorthogonal wavelet bases from JPEG 2000 [30]. More specifically, given a pair of one-dimensional scaling function and wavelet function, we form all possible tensor products between these two 1D functions to obtain a complete set of separable multi-dimensional wavelet basis functions. The wavelet packet algorithm we use follows [26]. In our experiments, each dataset is initially constructed as three third or fourth order tensors with one tensor for each color channel or velocity component. In our hierarchical approximation, these three tensors are placed at the top level and approximated collectively. There are typically 4-5 levels in the hierarchy. We did not perform any optimization over the parameters. The reduced ranks were chosen in a straightforward way. We ran five tests with different reduced ranks for each dataset. The reduced ranks used for the top-level approximation are respectively $1/2, 1/4, 1/8, 1/16$, and $1/32$ of the original rank.

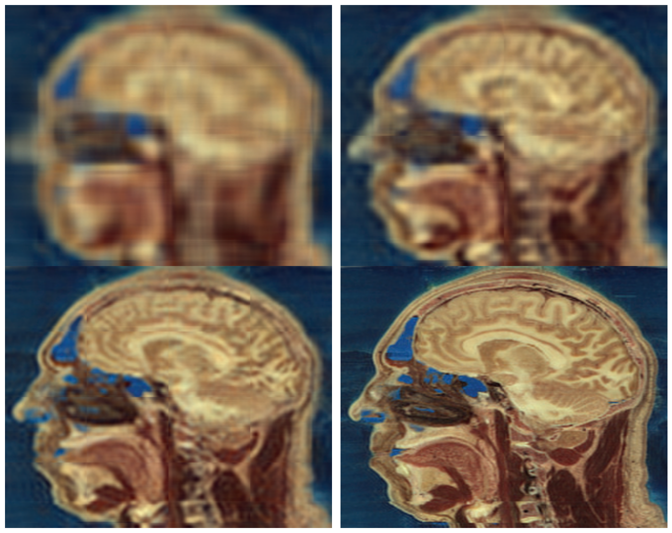


Fig. 6. Our hierarchical tensor approximation supports progressive transmission and decompression. Shown here are four images of a cross section in a 3D medical dataset. They have progressively more details. These four images correspond to the decompressed data at four different levels of a hierarchical approximation.

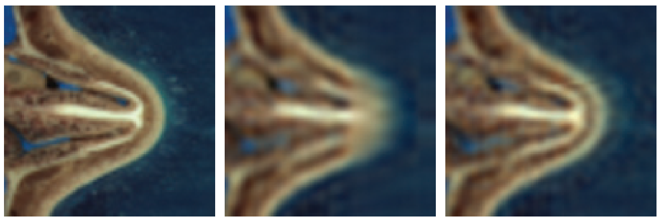

(a) Original (b) Single-level PSNR 29.89 (c) Multilevel PSNR 31.05

Fig. 7. A comparison of the Visible Human dataset reconstructed from the single-level tensor approximation in [14] and our hierarchical tensor approximation. (a) A magnified view of a cross section of the nose region. (b) A reconstructed image from the single-level tensor approximation. (c) A reconstructed image from our multilevel tensor approximation. (b)&(c) share the same compression ratio which is 15.7.

The common ratio between the ranks at two adjacent levels is always 0.5.

Fig. 8(a)-(b) show comparisons of compression ratios that can be achieved by each of the aforementioned four techniques over a wide range of PSNR values. Fig. 8(a) has the results for a subset of the Visible Human dataset, and Fig. 8(b) has the results for the velocity field of the time-varying volume dataset. Except for very few large PSNR values, our hierarchical tensor approximation achieved the highest compression ratios. And in most cases, the compression ratio it can achieve is at least one order of magnitude larger than that achieved by the wavelet transform. Meanwhile, our multilevel technique also maintains a fairly constant improvement over single-level tensor approximation. The curves in Fig. 8 indicate how the compression ratio and PSNR depend on the reduced ranks of the basis matrices. In general, both the approximation error and the compression ratio increase when the ranks decrease.

Fig. 5 shows visualization results for a cross section of the original time-varying volume dataset and four reconstructed ones. The original dataset is a four dimensional array of scalar density values. We apply the same color transfer function to the four cross sections. If a reconstructed result is similar to the original data, their visualizations should be very close to each other. Otherwise, significant deviation in color will occur. In Fig. 5, the result from our hierarchical approximation only has very minor color deviations while achieving a very high compression ratio. The result from wavelet transform has most obvious color deviations. Fig. 7 shows a visual comparison between the single-level and multi-level schemes on a local region from the Visible Human dataset. The original data has an extruding feature which the single-level method has failed to approximate well while the reconstruction from our multilevel method still preserves the important details.

Despite the existence of advanced visualization techniques such as direct volume rendering and isosurface rendering, in medical imaging and applications, a popular method for visualizing multi-dimensional medical images is still based on 2D cross sections since all the necessary details are displayed clearly and can be interpreted in a straightforward way. Our hierarchical tensor approximation offers better performance over single-level approximation in terms of extracting 2D cross sections from compressed data. Suppose we have a single-level rank-($r_1, r_2, ..., r_N$) approximation of $\mathcal{A} \in \Re^{n_1 \times n_2 \times ... \times n_N}$. The time complexity for decompressing this tensor is $O(\min(r_1, r_2, ..., r_N)\Pi_i n_i)$. To achieve a reasonable RMSE, $\min(r_1, r_2, ..., r_N)$ needs to be proportional to $\min(n_1, n_2, ..., n_N)$. On the other hand, our hierarchical technique only needs to decompress the subdivided tensors that intersect with the intended cross section at each level. Since the subdivided tensors become smaller when we descend in the hierarchy, the decompressed data points become more and more concentrated around the cross section and the decompression cost drops significantly. Therefore, the total decompression cost is dominated by the first level. Our experiments indicate that the minimum rank at the first level of a hierarchical approximation can be one order of magnitude smaller than the minimum rank of an equivalent single-level approximation. Thus, our hierarchical method can achieve significant speedup in decompression.

Although visualization is not the focus of this paper, we have built a simple visualization system based on viewing 2D cross sections of a multi-dimensional dataset. We have conducted experiments on direct visualization from compressed data using this system. To achieve faster decompression, we initially subdivide an original dataset into smaller blocks with a dimensionality of 64 for each elementary vector space. These blocks form the list of tensors at the top level of our hierarchy. They are approximated and further subdivided at lower levels. During each step of visualization, only those blocks intersecting with the intended cross section are decompressed. Visualization speed is dependent on the number of levels we need to decompress. Given a dataset with 80 blocks on the top level, our system can decompress all levels and continuously display cross sections at 5 frames per second on a 3.0GHz Pentium processor. An example of progressive decompression of a subset of the Visible Human dataset is shown in Fig. 6.

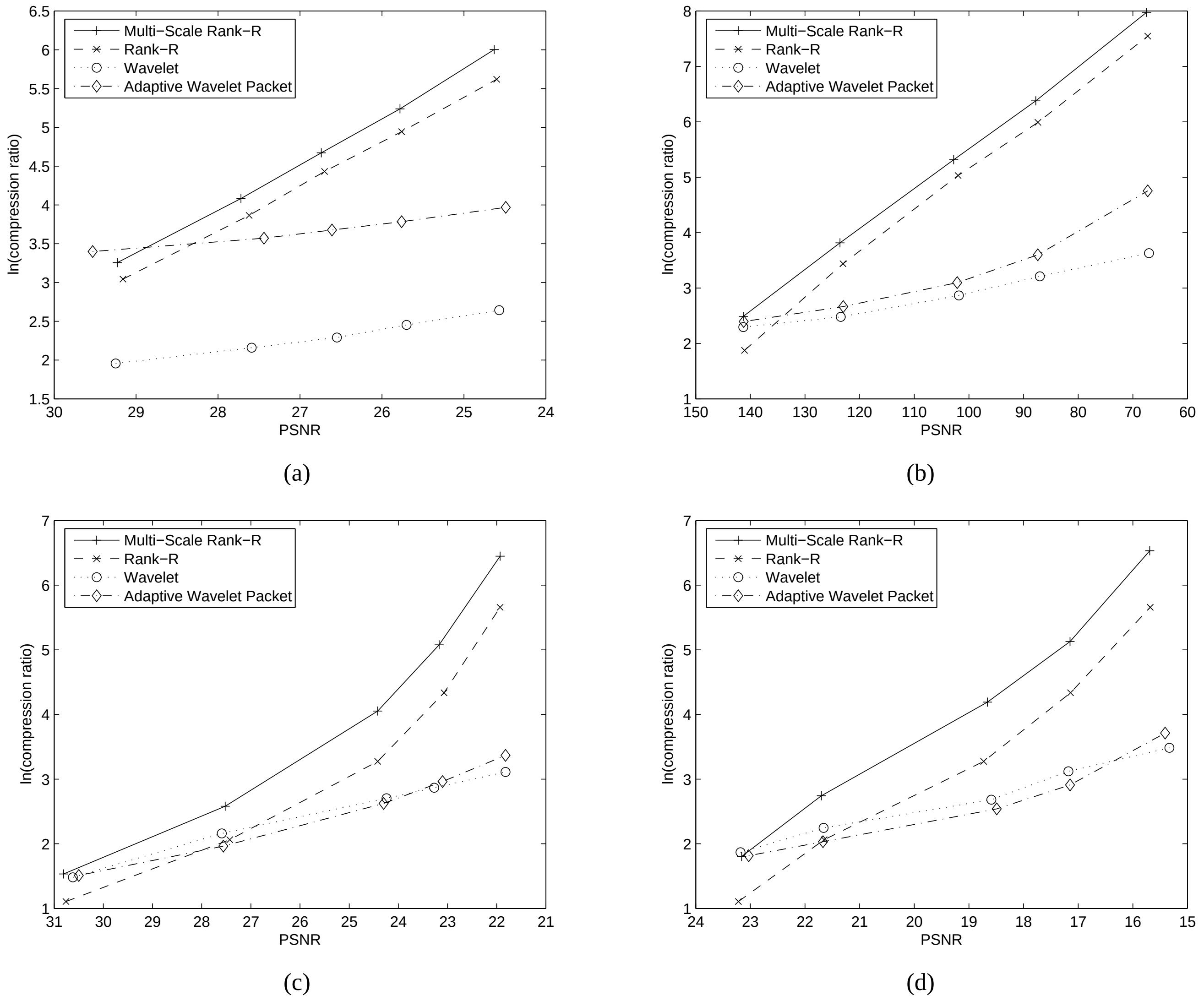


Fig. 8. Comparisons of data compression ratios achieved on four datasets by a bi-orthogonal wavelet transform (dotted), a corresponding wavelet packet transform (dash-dotted), the single-level rank-$(r_1, r_2, ..., r_N)$ tensor approximation (dashed), and our multilevel tensor approximation (solid). The datasets include (a) a subset of the Visible Human dataset, (b) the velocity field of the 4D time-varying volume dataset, (c) a SPONGE BTF, and (d) a LICHEN BTF. Overall, our hierarchical tensor approximation can achieve the highest compression ratios over a wide range of PSNR values. The wavelet transform exhibits the worst performance. The adaptive wavelet packet transform is in general better than the original wavelet transform and can occasionally achieve the highest compression ratios on high-end PSNR values. In many cases, the compression ratio achieved by our technique is at least one order of magnitude larger than that achieved by the wavelet transform. Meanwhile, our multilevel technique also outperforms single-level tensor approximation in all scenarios. Since we use logarithmic scales for compression ratios, a small difference actually represents a significant improvement. In fact, on the four datasets in (a)-(d), the compression ratios achieved by our technique are respectively 41.0%, 52.9%, 93.9% and 121.8% higher than those achieved by the single-level tensor approximation.

B. Data-Driven Rendering

Data-driven approaches have been a popular choice in rendering recently, including image-based rendering [31], [32], bidirectional texture functions (BTFs) [33], [34] and precomputed radiance transfer [35], [18]. These approaches typically involve large amount of acquired or precomputed data and rely on compact representation of visual data or their transfer operators to achieve efficient rendering of final images. Our hierarchical tensor approximation has the potential to improve the efficiency in data representation for all these approaches. We choose to measure the performance of our technique on BTFs because there has been extensive research on representing BTFs using tensor approximation [12], [13], [14], and these previous results can serve as a base for comparison.

Since it has been demonstrated in [14] that their single-level tensor approximation can outperform PCA and TensorTexture [13], we only compare our hierarchical tensor approximation against the wavelet bases used in JPEG 2000[30], a wavelet packet algorithm [26], and the single-level tensor approximation scheme from [14] on the BTF datasets presented in [36]. In our experiments, the parameter settings we use for compressing BTFs are the same as those we use for compressing the Visible Human dataset and the 4D time-varying scientific dataset. According to the results shown in Fig. 8(c)-(d), our hierarchical approximation achieves the

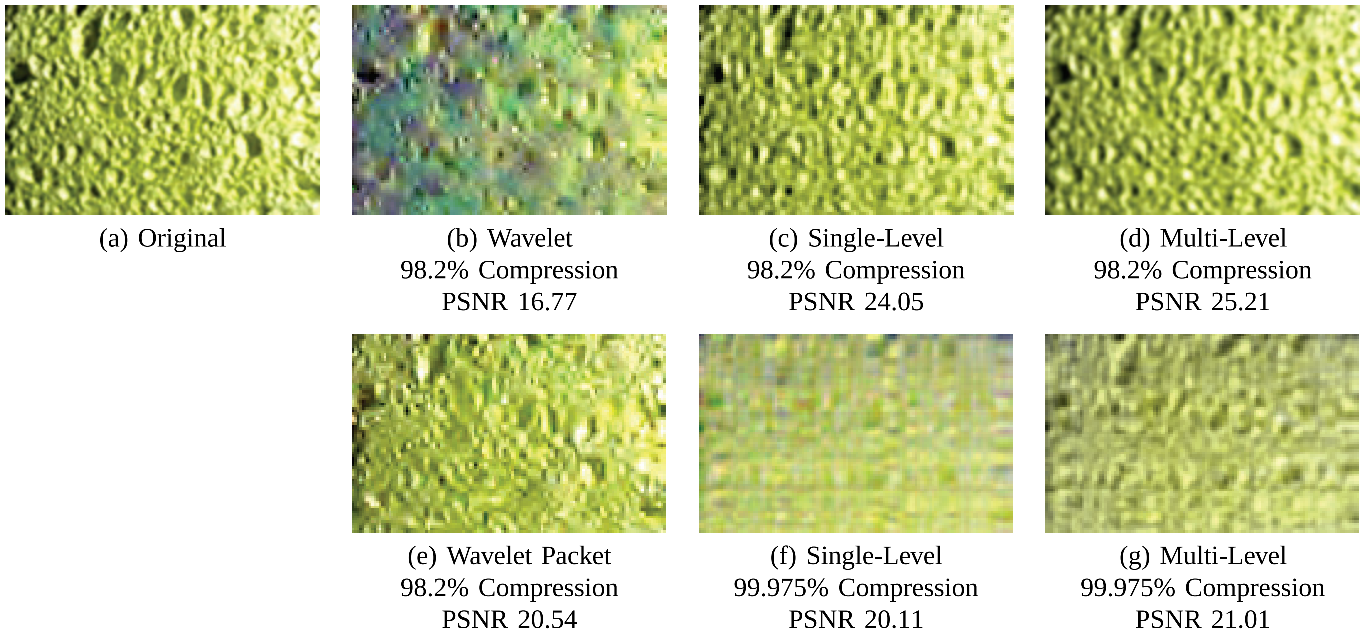


Fig. 9. A comparison of reconstructed BTF images from a bi-orthogonal wavelet transform, a corresponding adaptive wavelet packet transform, the single-level tensor approximation from [14] and our hierarchical tensor-based representation. The original SPONGE BTF has 45 views and 60 illumination directions, and the image resolution is 128x128. (a) An original BTF image. (b) A reconstructed image from the wavelet transform. (c)&(f) Reconstructed images from the single-level tensor approximation. (d)&(g) Reconstructed images from our multilevel tensor approximation. (e) A reconstructed image from the wavelet packet transform. The compression ratio for (b)-(e) is 55 while the compression ratio used for (f)-(g) is 3922. Overall, our results exhibit the best visual quality under the same compression ratio.

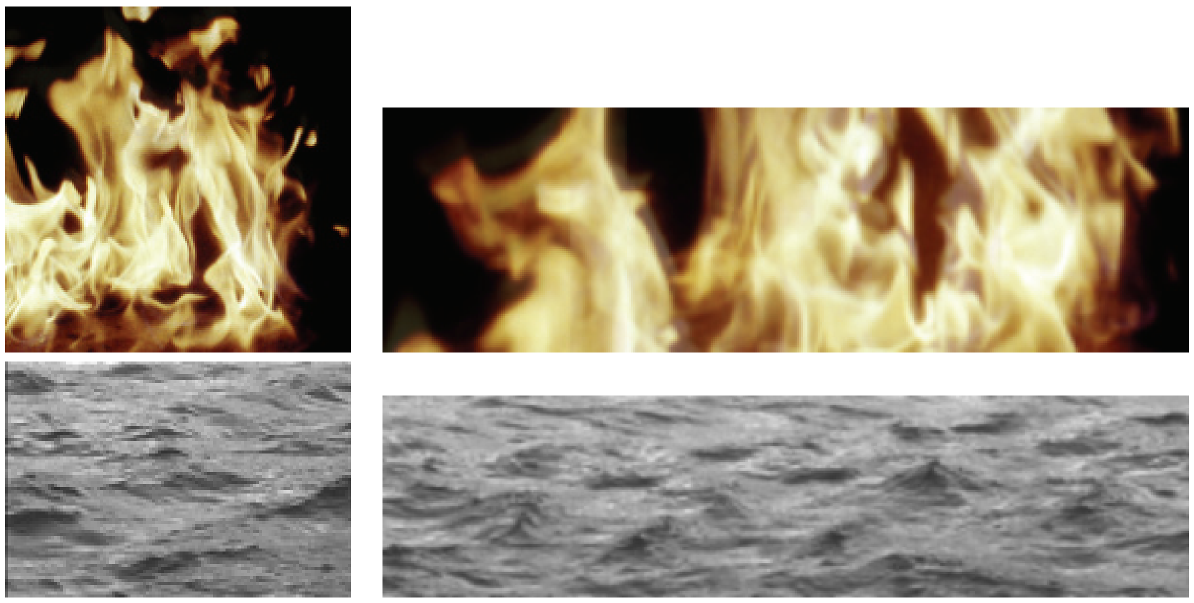


Fig. 10. Synthesis results for dynamic textures. Left: sample frames from the input sequences. Right: sample frames from the synthesized sequences.

highest compression ratio for almost all PSNR values we have tested. It maintains a significant improvement over single-level tensor approximation. Wavelet transform has the worst overall performance. Given the reconstructed images shown in Fig. 9, we can also conclude that our technique can effectively preserve the fine details of the BTF and achieve the best visual quality among the four.

In all experiments, the compression ratios for the wavelet transform are optimistically estimated according to the number of nonzero coefficients after quantization without considering the cost in coding their positions. We compute the compression ratios for the single-level tensor approximation according to the size of its basis matrices and core tensor. For our multilevel tensor approximation, we estimate the compression ratios according to the amount of storage required by all remaining tensors and basis matrices in the hierarchy after pruning.

C. Texture Synthesis

Template matching is the most frequent and costly step in most contemporary texture synthesis algorithms, especially in

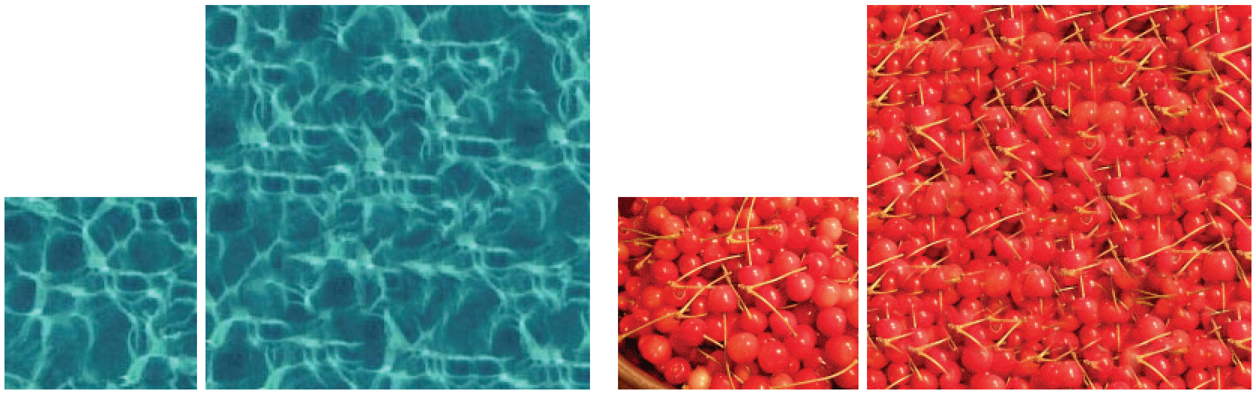


Fig. 11. Synthesis results for 2D textures. Small: sample textures. Large: synthesized textures. The resolution of the synthesized textures is 256x256.

neighborhood-based texture synthesis [37], [38], [39], [40], [41]. It can be formulated either as a convolution or a nearest neighbor search, which lead to different acceleration schemes such as FFT, kd-trees or tree-structured vector quantization. Although such schemes produce acceptable performance for 2D texture synthesis, they are inadequate for higher-dimensional textures, such as 3D dynamic textures where every patch is a 3D block with a much larger number of pixels than in the 2D case. Tensor approximation proves to be useful here because each 3D texture block can be considered as a small third-order tensor itself.

Suppose two 3D texture blocks are represented as two tensors, $\mathcal{P}_1$ and $\mathcal{P}_2$. Given three basis matrices with orthogonal columns and reduced ranks (r_1, r_2, and r_3), the rank-(r_1, r_2, r_3) approximation of $\mathcal{P}_i$ $(i = 1, 2)$ is given as $\tilde{\mathcal{P}}_i = \mathcal{Q}_i \times_1 \mathbf{U}^{(1)} \times_2 \mathbf{U}^{(2)} \times_3 \mathbf{U}^{(3)}$, where $\mathcal{Q}_i$ is the core tensor of $\tilde{\mathcal{P}}_i$. It can be easily shown that

$$\|\tilde{\mathcal{P}}_1 - \tilde{\mathcal{P}}_2\|^2 = \|\mathcal{Q}_1 - \mathcal{Q}_2\|^2. \tag{8}$$

Since $\|\mathcal{P}_1 - \mathcal{P}_2\|^2 \approx \|\tilde{\mathcal{P}}_1 - \tilde{\mathcal{P}}_2\|^2$, the summed squared differences (SSD) between two tensors can be well approximated by the SSD between their core tensors which may have a much smaller size and require much less computation. We use such tensor approximation to accelerate template matching in texture synthesis.

In practice, we generalize the multilevel 2D synthesis algorithm in [42] to 3D dynamic texture synthesis, and use tensor approximation together with kd-trees to perform nearest block search. The original synthesis algorithm refines the synthesis result using multiple levels of block size and texture resolution. Therefore, in a precomputing stage of our revised algorithm, all sample texture blocks for the same level are first collectively approximated using tensors with reduced ranks as in Section III. The resulting smaller core tensors are inserted into a kd-tree. We precompute such a kd-tree for each synthesis level. During the actual synthesis stage, given a query texture block, we first compute the core tensor of that query block and then search the corresponding kd-tree for the nearest core tensors, which further point to their corresponding original texture blocks from the sample texture.

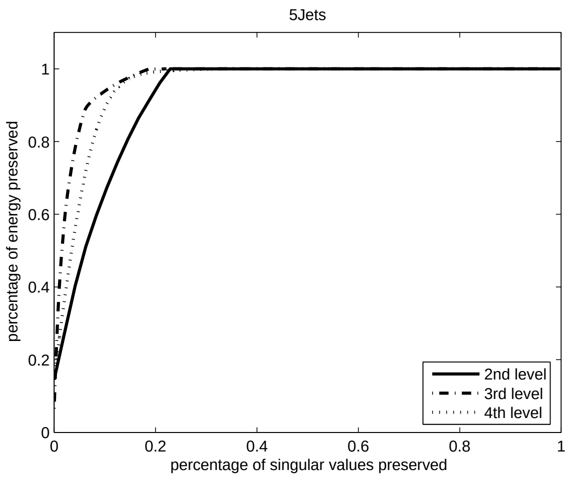
We have tested our tensor-based block search technique on both 2D textures (Fig. 11) and dynamic textures (Fig. 10 and the accompanying video). A typical size of the original 3D texture blocks is 32x32x32 and we use 5x5x5 core tensors to approximate them. Although the number of vectors in a kd-tree is still the same, the dimensionality of each vector has been reduced from 32768 to 125. In our experiments, this reduction in dimensionality makes each nearest neighbor search in the kd-tree more than 200 times faster. In comparison to template matching using 3D FFT, our accelerated kd-tree search is also at least 5 times faster, which makes block-based dynamic texture synthesis more computationally tractable. Surprisingly, our technique can achieve the same speedup over FFT even in 2D texture synthesis. Note that tensor approximation gives rise to a small amount of error in the texture data. Therefore, the compression ratio of the tensor approximation should be chosen carefully.

VI. Discussions and Analysis

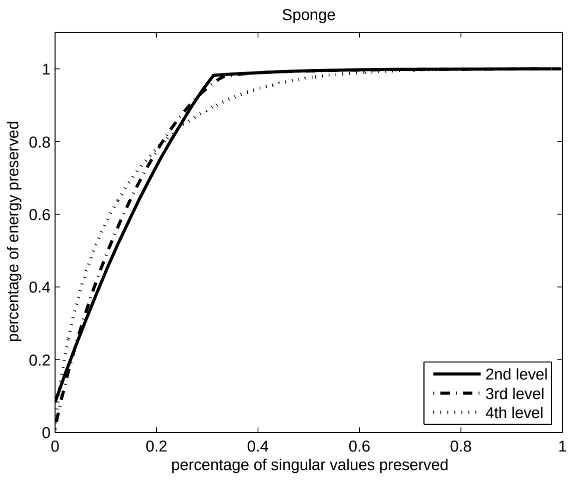
Resembling a multiresolution analysis such as wavelet transform, our approximation represents significant and typically low frequency components at higher levels of the hierarchy and less important (high frequency) components at lower levels. Because high frequency components have smaller spatial support, they can be approximated using shorter basis vectors. That is one of the reasons we keep subdividing the residual tensors from level to level and use increasingly shorter basis matrices to approximate them. Shorter basis matrices impose less overhead on storage.

More importantly, traditional multiresolution analysis simply applies scaled versions of a prescribed basis to signals at various different resolutions while our hierarchical approximation extracts basis matrices specifically tailored for the data being approximated. Therefore, our method is much better at removing redundancies in a specific dataset. In practice, we have found that the gained efficiency of our method in representing three or higher dimensional data surpasses the storage overhead for the adaptively extracted basis matrices. Note that we count these basis matrices when computing compression ratios.

As mentioned in Section II, neither rank-($r_1, r_2, ..., r_N$) approximation nor rank-r approximation performs a hierarchical transformation of the original data. Instead of reducing the



(a)



(b)

Fig. 12. Correlation among residual tensors. (a) The 4D time-varying dataset. (b) The SPONGE BTF. The horizontal axis in these diagrams indicates the percentage of singular values while the vertical axis indicates the percentage of total energy. Stronger correlation needs fewer number of singular values to capture the same percentage of energy. Three curves for three different levels are shown for each dataset.

ranks of the basis matrices as in single-level tensor approximations, our hierarchical approach relies on eliminating small insignificant core tensors at lower levels of the hierarchy to achieve compact representations. Because our technique integrates the incomplete approximations at multiple levels to produce the final approximation, for the same level of accuracy, the minimum rank of the core tensor at the first level can be much smaller than the minimum rank of an equivalent single-level tensor approximation. Although blockwise partitioning of the original tensor was performed in [14], it was only for the purpose of out-of-core computing. Our hierarchical tensor subdivision, on the other hand, is designed to exploit the inherent multi-scale structures of the data, and is performed on the residual tensors passed from higher levels.

Our hierarchical approximation shares common intuitions with H-matrices proposed in [19]. Nonetheless, there exist important distinctions between the two. First, in a H-matrix, only the diagonal subblocks and those subblocks adjacent to them are recursively subdivided. This restriction has recently been lifted for compressing reflectance fields [20] which nonetheless only adopts a rank-1 tensor to approximate every subblock. Our method, on the other hand, is designed for compressing generic multidimensional visual data. It subdivides any subblocks when necessary and approximates every subblock using a more powerful rank-($r_1, r_2, ..., r_N$) tensor. Second, H-matrices only approximate every subblock at the bottom level of the subdivision tree using a single tensor approximation while our method does multilevel approximation with a finer level approximating the residual errors from the coarser level. As discussed in Section I, visual data are superposition of signals at multiple frequencies or scales. It is more effective to decompose the original data into components with different scales and then compress these components separately. Further discussion and comparisons regarding this can be found by the end of Section VI-A. Third, our method performs ensemble approximation at each level to further improve the compression ratio while H-matrices generate a distinct set of basis vectors for each subblock. Ensemble approximation makes use of a common set of basis matrices for multiple subblocks to much reduce the basis overhead.

One limitation of our method is that it is computationally more expensive than both wavelets and single-level tensor approximation even though we have taken a relatively fast greedy approach. On average, our hierarchical method is three times as slow as single-level tensor approximation and fifteen times as slow as wavelets.

A. Covariance Analysis

At each level of the hierarchical transformation, we perform tensor ensemble approximation, which can achieve a more compact representation than individual tensor approximation when the collection of tensors at each level exhibit a certain degree of correlation. In the following, we give both mathematical justification and experimental evidence to demonstrate such correlation does exist.

Suppose the list of tensors at level l is $\mathcal{A}_1^l, \mathcal{A}_2^l, \cdots, \mathcal{A}_{m_l}^l$, where m_l is the number of tensors and $\mathcal{A}_i^l \in \Re^{n_1^l \times n_2^l \times \ldots \times n_N^l}$. We unfold each tensor, $\mathcal{A}_i^l$, into a vector, $\mathbf{X}_i$. Denote the mean of all these vectors as $\bar{\mathbf{X}}$. We further arrange mean-subtracted vectors, $\hat{\mathbf{X}}_i = \mathbf{X}_i - \bar{\mathbf{X}}$ as columns of a matrix, $\mathbf{S}$, which has a singular value decomposition (SVD), $\mathbf{S} = \mathbf{U\Sigma V}^T$, where $\mathbf{U}$ and $\mathbf{V}$ are basis matrices with orthonormal columns and $\mathbf{\Sigma}$ is a diagonal matrix. As we know, the covariance matrix of $\mathbf{S}$ is $\mathbf{SS^T}$, which is equivalent to $\mathbf{U\Sigma}^2\mathbf{U}^T$.

With modest assumptions, there exists correlation among different local tensors at the same level of the hierarchy. First, a common assumption in image compression is correlation among spatially close pixels [1]. Second, we assume $\{\mathbf{X}_i\}_{i=1}^{m_l}$ are different realizations (observations) of the same underlying statistical process, $\mathbf{X}$, which means different regions from the same dataset have shared frequency-domain characteristics. The second assumption is supported by the evidences and experimental results from [1], [43]. This assumption gives rise to correlation among the basis vectors used for representing each local region rather than direct correlation among the data in different regions. Such correlation among the basis vectors exists when a small subset of the singular values of $\mathbf{S}$ are more important than the rest and the column vectors of $\mathbf{S}$ can be reasonably approximated by linear combinations of a corresponding subset of basis vectors in $\mathbf{U}$. In our hierarchical approximation, since higher levels already approximate the large-scale low-frequency components, the residual tensors at a certain level largely represent components at a scale equal to or smaller than the scale of that level. Nevertheless, the second assumption implies correlation at all scales.

We have performed experiments to verify the existence of strong correlation among subdivided residual tensors at each level. In all our experiments, the singular values from the aforementioned SVD are highly nonuniform and typically 30% of the singular values can capture more than 90% of the total residual energy. This means most of the singular values are close to zero, and the residuals at each level can be well approximated using a subset of principal components whose size is less than one third of the number of residual tensors. Fig. 12 show correlation results for two of the datasets. Correlation results for three different levels are shown for these datasets. As we can see, there exists strong correlation at all three levels, and the degree of correlation varies slightly from level to level. In the 4D time-varying dataset, the third level clearly exhibits the strongest correlation while in the SPONGE dataset, the degree of correlation is comparable at all levels.

We have further compared the degree of correlation with and without the approximations at higher levels. It turned out that the correlation among the residual tensors is slightly weaker than the correlation without higher-level approximations. Nevertheless, it should be clarified that these two correlations should not be directly compared, nor should they be used for predicting the performance of a compression algorithm. The degree of correlation of the residual tensors at a specific level is only indicative of the degree of compression achievable on that level but not the overall compression of the entire dataset. A more accurate prediction of the overall compression performance should be based on the total number of tensors surviving pruning because we need to store a core tensor for each of the remaining tensors. Our multilevel approximation can achieve better performance because the residual tensors at the lower levels tend to have smaller magnitudes than those tensors without higher-level approximation and, thus, are more likely to be pruned. Pruning lower-level tensors is advantageous because the number of tensors at each level is exponentially increasing from top to bottom. This prediction has been confirmed by a large number of comparisons we have performed between these two schemes. Take the SPONGE dataset as an example. If we set PSNR=24.97, the compression ratio achieved by our multilevel approximation is 44.35 while the compression ratio achieved with multilevel subdivision but without higher-level approximation is only 31.70.

VII. Conclusions and Future Work

In this paper, we developed a compact data representation technique based on a hierarchical tensor-based transformation. Experimental results indicate that our technique can achieve higher compression ratios and quality than previous methods, including wavelet transforms, wavelet packet transforms and single-level tensor approximation on three or higher dimensional visual data. We have successfully applied our technique to multiple tasks involving multi-dimensional visual data, including medical and scientific data visualization, data-driven rendering and texture synthesis.

There exist a few directions for future work. First, on 2D images, our current method does not perform as well as wavelets and wavelet packets because our adaptive bases require a more significant storage overhead in 2D than in higher dimensions. We would like to investigate adaptive methods with less basis overhead for the 2D domain. Second, it is possible to extend tensor ensemble approximation to tensors across multiple scales and, thus, achieve even higher compression ratios. However, such a method would be more computationally expensive since basis matrices across multiple scales would need to be optimized simultaneously.

Acknowledgments

We wish to thank Kwan-Liu Ma for sharing his time-varying volume data, the authors of [36] for sharing their BTF database, and the National Library of Medicine for licensing the Visible Human dataset. This work was partially supported by National Science Foundation (CCR-0132970) and UIUC Research Board.

References

[1] E. Simoncelli and B. Olshausen, “Natural image statistics and neural representation,” *Annu. Rev. Neurosci.*, vol. 24, pp. 1193–1216, 2001.
[2] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.
[3] T. Zhang and G. Golub, “Rank-one approximation to high order tensors,” *SIAM J. Matrix Analysis and Applications*, vol. 23, no. 2, pp. 534–550, 2001.
[4] P. Kroonenberg and J. de Leeuw, “Principal component analysis of three-mode data by means of alternating least squares algorithms,” *Psychometrika*, vol. 45, pp. 324–1342, 1980.
[5] L. D. Lathauwer, B. de Moor, and J. Vandewalle, “On the best rank-1 and rank-$(R_1, R_2, ..., R_n)$ approximation of higher-order tensors,” *SIAM J. Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1324–1342, 2000.
[6] ——, “A multilinear singular value decomposition,” *SIAM J. Matrix Analysis and Applications*, vol. 21, no. 4, pp. 1253–1278, 2000.
[7] A. Shashua and A. Levin, “Linear image regression and classification using the tensor-rank principle,” in *IEEE Conf. Computer Vision and Pattern Recognition*, 2001.
[8] D. Xu, S. Yan, L. Zhang, H.-J. Zhang, Z. Liu, and H.-Y. Shum, “Concurrent subspaces analysis,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2005, pp. 203–208.
[9] M. A. O. Vasilescu and D. Terzopoulos, “Multilinear analysis of image ensembles: Tensorfaces,” in *European Conference on Computer Vision*, 2002, pp. 447–460.
[10] H. Wang and N. Ahuja, “Facial expression decomposition,” in *Int. Conf. on Computer Vision*, 2003, pp. 958–965.
[11] D. Vlasic, M. Brand, H. Pfister, and J. Popovic, “Face transfer with multilinear models,” *ACM Transactions on Graphics*, vol. 25, no. 3, pp. 426–433, 2005.
[12] R. Furukawa, H. Kawasaki, K. Ikeuchi, and M. Sakauchi, “Appearance based object modeling using texture database: Acquisition, compression, and rendering,” in *13th Eurographics Workshop on Rendering*, 2002, pp. 257–265.

[13] M. Vasilescu and D. Terzopoulos, "Tensortextures: Multilinear image-based rendering," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 334–340, 2004.

[14] H. Wang, Q. Wu, L. Shi, Y. Yu, and N. Ahuja, "Out-of-core tensor approximation of multi-dimensional matrices of visual data," *ACM Transactions on Graphics*, vol. 24, no. 3, 2005.

[15] T. Hazan, S. Polak, and A. Shashua, "Sparse image coding using a 3d non-negative tensor factorization," in *Tenth International Conference on Computer Vision*, Washington, DC, USA, 2005, pp. 50–57.

[16] D. Lee and H. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, 1999.

[17] X. He, D. Cai, H. Liu, and J. Han, "Image clustering with tensor representation," in *13th annual ACM international conference on Multimedia*, 2005, pp. 132–140.

[18] Y.-T. Tsai and Z.-C. Shih, "All-frequency precomputed radiance transfer using spherical radial basis functions and clustered tensor approximation," *ACM TOG*, vol. 25, no. 3, pp. 967–976, 2006.

[19] W. Hackbusch, "A Sparse Matrix Arithmetic based on $\mathcal{H}$-Matrices. Part I: Introduction to $\mathcal{H}$-matrices," *Computing*, vol. 62, no. 2, pp. 89–108, 1999.

[20] G. Garg, E.-V. Talvala, M. Levoy, and H. P. A. Lensch, "Symmetric Photography: Exploiting Data-sparseness in Reflectance Fields," in *Proceedings Eurographics Symposium on Rendering*, 2006, pp. 251–262.

[21] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image coding using wavelet transform," *IEEE Transactions on Image Processing*, vol. 1, no. 2, pp. 205–220, 1992.

[22] J. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," *IEEE Transactions on Signal Processing*, vol. 41, no. 12, pp. 3445–3462, 1993.

[23] I. Ihm and S. Park, "Wavelet-based 3d compression scheme for interactive visualization of very large volume data," *Compuer Graphics Forum*, vol. 18, no. 1, pp. 3–15, 1999.

[24] F. Rodler, "Wavelet based 3d compression with fast random access for very large volume data," in *Proceedings of the 7th Pacific Conference on Computer Graphics and Applications*, 1999, pp. 108–117.

[25] F. Meyer, A. Averbuch, and J.-O. Stromberg, "Fast adaptive wavelet packet image compression," *IEEE Transactions on Image Processing*, vol. 9, no. 5, pp. 792–800, 2000.

[26] A. Jensen and A. la Cour-Harbo, *Ripples in Mathematics: The Discrete Wavelet Transform*. Springer, 2001.

[27] E. Candès and D. Donoho, *Curvelets – a surprisingly effective nonadaptive representation for objects with edges*. Vanderbilt University Press, 1999.

[28] ——, "Ridgelets: a key to higher-dimensional intermittency?" *Phil. Trans. R. Soc. Lond. A.*, pp. 2495–2509, 1999.

[29] M. Do and M. Vetterli, "The contourlet transform: an efficient directional multiresolution image representation," *IEEE Transactions on Image Processing*, vol. 14, no. 12, pp. 2091–2106, 2005.

[30] M. Boliek, C. Christopoulos, and E. M. (editors), "Jpeg 2000 image coding system (jpeg 2000 part i final committee draft version 1.0)," ISO/IEC FCD15444-1, http://www.jpeg.org/jpeg2000/CDs15444.html, March 2000.

[31] D. Wood, D. Azuma, K. Aldinger, B. Curless, T. Duchamp, D. Salesin, and W. Stuetzle, "Surface light fields for 3d photography," in *SIGGRAPH'00*, 2000, pp. 287–296.

[32] W.-C. Chen, J.-Y. Bouguet, M. Chu, and R. Grzeszczuk, "Light field mapping: Efficient representation and hardware rendering of surface light fields," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 447–456, 2002.

[33] K. J. Dana, B. van Ginneken, S. K. Nayar, and J. J. Koenderink, "Reflectance and texture of real world surfaces," *ACM Transactions on Graphics*, vol. 18, no. 1, pp. 1–34, 1999.

[34] X. Liu, Y. Hu, J. Zhang, X. Tong, B. Guo, and H.-Y. Shum, "Synthesis and rendering of bidirectional texture functions on arbitrary surfaces," *IEEE Trans. Visualization and Computer Graphics*, vol. 10, no. 3, pp. 278–289, 2004.

[35] P.-P. Sloan, J. Kautz, and J. Snyder, "Precompted radiance transfer for real-time rendering in dynamic, low-frequency lighting environments," *ACM TOG*, vol. 21, no. 3, pp. 527–536, 2002.

[36] M. Koudelka, S. Magda, P. Belhumeur, and D. Kriegman, "Acquisition, compression, and synthesis of bidirectional texture functions," in *3rd Intl. Workshop on Texture Analysis and Synthesis*, 2003, pp. 59–64.

[37] A. Efros and T. Leung, "Texture synthesis by non-parametric sampling," in *Intl. Conf. Computer Vision*, 1999, pp. 1033–1038.

[38] L.-Y. Wei and M. Levoy, "Fast texture synthesis using tree-structured vector quantization," in *Proceedings of Siggraph*, 2000, pp. 479–488.

[39] A. Efros and W. Freeman, "Image quilting for texture synthesis and transfer," in *SIGGRAPH'01*, 2001, pp. 341–346.

[40] L. Liang, C. Liu, Y. Xu, B. Guo, and H.-Y. Shum, "Real-time texture synthesis using patch-based sampling," *ACM Trans. Graphics*, vol. 20, no. 3, pp. 127–150, 2001.

[41] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: Image and video synthesis using graph cuts," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 277–286, 2003.

[42] V. Kwatra, I. Essa, A. Bobick, and N. Kwatra, "Texture optimization for example-based synthesis," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 795–802, 2005.

[43] K. Nishino, S. Nayar, and T. Jebara, "Clustered blockwise pca for representing visual data," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1675–1679, 2005.