Color Image Compression using a Learned Dictionary of Pairs of Orthonormal Bases

Xin Hou[*], Karthik S. Gurumoorthy[†] and Ajit Rajwade[‡§]

Abstract

We present a new color image compression algorithm for RGB images. In our previous work [6], we presented a machine learning technique to derive a dictionary of orthonormal basis triples for compact representation of an ensemble of color image patches from a training set. The patches were represented as 3D arrays of size $n \times n \times 3$, and our technique was based on the higher order singular value decomposition (HOSVD), an extension of the singular value decomposition (SVD) to higher order matrices [3]. The learning scheme exploited the cross-coupling between the R,G,B channels by implicitly learning a color space. In this paper, we show the benefits of representing color image patches as 2D matrices of size $n \times 3n$ and learning a dictionary of orthonormal basis pairs. We also present a method to leverage greater representational power from a learned dictionary without increasing its size. We present experimental results on all these variants of our method and compare them to JPEG and JPEG2000.

1 Introduction

Contemporary compression algorithms such as JPEG or JPEG-2000 use fixed bases such as the discrete cosine transform (DCT) or wavelet transform for building dictionaries for compact representation, exploiting the well-known fact that the projection coefficients of natural images or image patches onto these bases are sparse. Nevertheless, these bases are universal - properties specific to restricted classes of data may be better represented by tuning the bases to these data on the fly, using machine learning techniques. Such a philosophy has been previously adopted in papers such as [2], [4] and in our previous work [5], [6]. The gist of our approach from [5] was to learn a small number of matrix orthonormal basis pairs from a training set of patches belonging to gray-scale images in such a way that the projection of the patches onto at least one of those bases was sparse. Given an image to be compressed and an allowed error value, its patches were projected onto the particular basis pair that yielded the sparsest possible representation without exceeding the error value.

In color (RGB) image compression, it is a well-known fact that independent compression of the R, G, B channels is sub-optimal as it ignores the inherent coupling between the channels. Commonly, the RGB images are converted to YCbCr or some other decorrelated color space followed

[*] Department of Electrical and Computer Engineering, University of Florida, Gainesville, USA

[†] Department of Computer and Information Sciences and Engineering, University of Florida, Gainesville, USA

[‡] Department of Computer and Information Sciences and Engineering, University of Florida, Gainesville, USA

[§] Contact author: avr@cise.ufl.edu

by independent compression in each channel. This method is also part of the JPEG/JPEG-2000 standards. In [6], our approach from [5] was extended to handle color (RGB) images by representing color image patches as 3D matrices of size $n \times n \times 3$ and learning triples of orthonormal bases. The overall learning scheme was based on the HOSVD [3], which can be regarded as the analog of the SVD to higher order matrices. In this method, the RGB images were not converted to any decorrelated color space. Instead, a color space tuned to the training data was learned implicitly using tensor algebra.

In this paper, we revisit the approach from [6]. We switch from the 3D representation of the color image patch to a 2D representation in the form of matrices of size $n \times 3n$. We demonstrate superior experimental results and also discuss intuitive reasons why this representation is superior to the original one. Secondly, we also provide a method to increase the representational power of a dictionary of orthonormal basis pairs without increasing its size.

This paper is organized as follows. In Section 2, we review our previous work for gray-scale and color image compression.

2 Previous Work

2.1 Compression of gray-scale images

Consider a set of images, each of size $M_1 \times M_2$, divided into (say) N non-overlapping patches $\{P_i\}$, $1 \leq i \leq N$ of size $m_1 \times m_2, m_1 \ll M_1, m_2 \ll M_2$. We exploit the similarity across these patches by representing them as sparse projections onto some $K \ll N$ orthonormal bases $\{(U_a, V_a)\}$ $(1 \leq a \leq K)$, learned from the training set itself.

The SVD of an image patch $P \in R^{m_1 \times m_2}$ is given by $P = USV^T$, where $S \in R^{m_1 \times m_2}$ is a diagonal matrix of singular values. Now, P can also be represented as a combination of *any* set of orthonormal bases $\bar{U}$ and $\bar{V}$, different from those obtained from the SVD of P. In this case, we have $P = \bar{U}S\bar{V}^T$ where S is non-diagonal. We now seek to answer the following question: What *sparse* matrix $W \in \mathcal{R}^{m_1 \times m_2}$ will reconstruct P from a pair of orthonormal bases $\bar{U}$ and $\bar{V}$ with the least error $\|P - \bar{U}W\bar{V}^T\|^2$? Sparsity is quantified by an upper bound T on the number of non-zero elements in W (denoted as $\|W\|_0$). The *optimal* W with this sparsity constraint is obtained by nullifying the least (in absolute value) $m_1m_2 - T$ elements of the estimated projection matrix $S = \bar{U}^T P \bar{V}$ (due to the orthonormality of $\bar{U}$ and $\bar{V}$).

The overall objective function to learn $\{(U_a, V_a)\}$ for lossy compression is given as follows:

$$E(\{U_a, V_a, S_{ia}, M_{ia}\}) = \sum_{i=1}^{N} \sum_{a=1}^{K} M_{ia} \|P_i - U_a S_{ia} V_a^T\|^2 \tag{1}$$

subject to the following constraints:

$$\forall a \; U_a^T U_a = V_a^T V_a = I \tag{2}$$

$$\forall (i,a) \; \|S_{ia}\|_0 \leq T \tag{3}$$

$$\forall i \; \sum_a M_{ia} = 1 \text{ and } \forall (i,a) \; M_{ia} \in \{0,1\} \tag{4}$$

where S_{ia} is the projection of the i^{th} patch onto the a^{th} basis, and M_{ia} indicates a membership of the i^{th} patch onto the a^{th} basis. Starting with random orthonormal bases for all $\{U_a, V_a\}$ and $M_{ia} = \frac{1}{K} \; \forall (i,a)$, the matrix S_{ia} is computed using $S_{ia} = U_a^T P_i V_a$ and its $m_1m_2 - T$ elements with smallest magnitude are nullified. The updates for U_a are given as follows:

$$Z_{1a} = \sum_i M_{ia} P_i V_a S_{ia}^T \tag{5}$$

$$U_a = Z_{1a}(Z_{1a}^T Z_{1a})^{-\frac{1}{2}} \tag{6}$$

$$U_a = (\Gamma_{1a}\Psi\Upsilon_{1a}^T)((\Gamma_{1a}\Psi\Upsilon_{1a}^T)^T(\Gamma_{1a}\Psi\Upsilon_{1a}^T))^{-\frac{1}{2}} = \Gamma_{1a}\Upsilon_{1a}^T. \tag{7}$$

where the SVD of Z_{1a} will give us $Z_{1a} = \Gamma_{1a}\Psi\Upsilon_{1a}^T$ with Γ_{1a} and Υ_{1a} being orthonormal matrices and Ψ being a diagonal matrix. The bases V_a are updated in a similar manner. The membership values are relaxed so that $M_{ia} \in [0,1]$ leading to a deterministic annealing framework [7]. The membership values are now obtained by:

$$M_{ia} = \frac{e^{-\beta \|P_i - U_a S_{ia} V_a^T\|^2}}{\sum_{b=1}^K e^{-\beta \|P_i - U_b S_{ib} V_b^T\|^2}}. \tag{8}$$

where β is an annealing parameter. The matrices $\{S_{ia}, U_a, V_a\}$ and M are then updated sequentially following one another in a deterministic annealing framework (increasing β across iterations) until M_{ia} turns out to be (nearly) binary. For more details, refer to [6].

For lossy compression of an unseen image (not part of the training set), we fix a mean reconstruction error δ. A patch Q from this image is now projected onto the particular basis pair $(U_i^\star, V_i^\star)$ (from the set $\{U_a, V_a\}$) which produces the *sparsest* projection matrix $S_i^\star$ that yields an error $\frac{\|P_i - U_i^\star S_i^\star V_i^{\star T}\|^2}{m_1 m_2} \leq \delta$. Note that different test patches will lead to projection matrices of different L_0 norms, depending upon their inherent ‘complexity’.

2.2 Extension to color images

We now consider a set of color images represented as 3D matrices, each of size $M_1 \times M_2 \times 3$. We divide the images into totally N non-overlapping patches $\{P_i\}$ of size $m_1 \times m_2 \times 3, m_1 \ll M_1, m_2 \ll M_2$. We treat each patch as a separate tensor and seek to represent these patches by sparse projections onto triples of some $K \ll N$ exemplar orthonormal bases $\{(U_a, V_a, W_a)\}$ learned from the very same training set. For color image patches, the matrices $\{W_a\}$ actually represent color spaces learned from the training data and adapted to the specific patches that ‘belong’ to a particular ‘cluster’ (see Equation 15). This is unlike contemporary color image compression algorithms that use *fixed* color spaces such as RGB or YCbCr.

Let $P \in R^{m_1 \times m_2 \times 3}$ be an image patch. Using HOSVD [3], we can represent P as a combination of orthonormal bases $U \in \mathcal{O}(m_1)$[1], $V \in \mathcal{O}(m_2)$ and $W \in \mathcal{O}(3)$ in the form $P = S \times_1 U \times_2 V \times_3 W$, where $S \in R^{m_1 \times m_2 \times 3}$ is termed the core-tensor. The operators $\times_i$ refer to tensor-matrix multiplication over different axes. The core-tensor has special properties such as all-orthogonality and ordering. See [3] for more details. Now, P can also be represented as a combination of *any* set of orthonormal bases $\bar{U}$, $\bar{V}$ and $\bar{W}$, different from those obtained from the HOSVD of P. In this case, we have $P = S \times_1 \bar{U} \times_2 \bar{V} \times_3 \bar{W}$ where S is not guaranteed to be an all-orthogonal tensor, nor is it guaranteed to obey the ordering property.

Again, we ask the following question: What *sparse* tensor $Q \in R^{m_1 \times m_2 \times 3}$ will reconstruct P from a triple of orthonormal bases $(\bar{U}, \bar{V}, \bar{W})$ with the least error $\|P - Q \times_1 \bar{U} \times_2 \bar{V} \times_3 \bar{W}\|^2$? Sparsity is again quantified by an upper bound T on $\|Q\|_0$. The *optimal* Q with this sparsity constraint is obtained by nullifying the least (in absolute value) $3m_1m_2 - T$ elements of

[1] We refer to the group of orthogonal matrices of size $n \times n$ as $\mathcal{O}(n)$.

the estimated projection tensor $S = P \times_1 \bar{U}^T \times_2 \bar{V}^T \times_3 \bar{W}^T$ (due to the orthonormality of $\bar{U}$, $\bar{V}$ and $\bar{W}$).

The overall objective function to learn $\{(U_a, V_a, W_a)\}$ for lossy compression is given as follows:

$$E(\{U_a, V_a, W_a, S_{ia}, M_{ia}\}) = \\ \sum_{i=1}^{N}\sum_{a=1}^{K} M_{ia}\|P_i - S_{ia} \times_1 U_a \times_2 V_a \times_3 W_a\|^2 \tag{9}$$

subject to the following constraints:

$$\forall a \; U_a^T U_a = V_a^T V_a = W_a^T W_a = I \tag{10}$$

$$\forall(i,a) \; \|S_{ia}\|_0 \leq T \tag{11}$$

$$\forall i \; \sum_a M_{ia} = 1 \text{ and } \forall(i,a) \; M_{ia} \in \{0,1\}. \tag{12}$$

Here M_{ia} is a binary matrix of size $N \times K$ which indicates whether the i^{th} patch belongs to the space defined by (U_a, V_a, W_a). We first initialize $\{U_a\}$, $\{V_a\}$ and $\{W_a\}$ to random orthonormal matrices $\forall a$, and $M_{ia} = \frac{1}{K}$, $\forall(i,a)$. Using the fact that $\{U_a\}$, $\{V_a\}$ and $\{W_a\}$ are orthonormal, the projection matrix S_{ia} is computed by the rule:

$$S_{ia} = P_i \times_1 U_a^T \times_2 V_a^T \times_3 W_a^T, \forall(i,a). \tag{13}$$

We have an update rule for U_a:

$$Z_{Ua} = \sum_i M_{ia} P_{i(1)} (V_a \otimes W_a) S_{ia(1)}^T; \\ U_a = Z_{Ua}(Z_{Ua}^T Z_{Ua})^{-\frac{1}{2}} = \Gamma_{1a}\Upsilon_{1a}^T. \tag{14}$$

Here Γ_{1a} and Υ_{1a} are orthonormal matrices obtained from the SVD of Z_{Ua}, and $P_{i(1)}$ is the first unfolding of tensor P_i [3]. V_a and W_a are updated similarly by second and third unfolding of the tensors respectively. Using the deterministic annealing framework described in Section 2.1, the membership values are relaxed from being binary to lie in the interval $[0,1]$. They are updated as follows:

$$M_{ia} = \frac{e^{-\beta\|P_i - S_{ia}\times_1 U_a \times_2 V_a \times_3 W_a\|^2}}{\sum_{b=1}^{K} e^{-\beta\|P_i - S_{ib}\times_1 U_b \times_2 V_b \times_3 W_b\|^2}}. \tag{15}$$

The tensors $\{S_{ia}\}$, the bases $\{U_a, V_a, W_a\}$ and the memberships M_{ia} are then updated sequentially by deterministic annealing until M_{ia} is (nearly) binary.

The lossy compression of an unseen image, under allowed error δ follows a procedure very similar to that described in Section 2.1.

3 Suggested Improvements

3.1 Matrix representation for color image patches

In Section 2.2, a 3D matrix $X \in R^{m_1 \times m_2 \times m_3}$ was expressed in the form $X = S \times_1 U \times_2 V \times_3 W$ where $U \in \mathcal{O}(m_1), V \in \mathcal{O}(m_2), W \in \mathcal{O}(m_3), S \in R^{m_1 \times m_2 \times m_3}$. This is equivalently expressed as follows:

$$X_{(1)} = U \cdot S_{(1)} \cdot (V \otimes W)^T \tag{16}$$

Table 1: Dictionary storage for different patch representations, for K bases/basis-pairs/basis-triples. Color image patches are of size $m_1 \times m_2 \times 3$.

Patch Representation	Dictionary Storage
Vector ($3m_1m_2 \times 1$)	$K \times 9m_1^2m_2^2$
2D ($m_1 \times 3m_2$)	$K \times (m_1^2 + 9m_2^2)$
3D ($m_1 \times m_2 \times 3$)	$K \times (m_1^2 + m_2^2 + 3^2)$

where $X_{(1)}$ is the first unfolding of X with $X_{(1)} \in R^{m_1 \times m_2 m_3}$ and $\otimes$ stands for the matrix Kronecker product [3].

Now consider that we represent the color image patch not as a 3D matrix of size $m_1 \times m_2 \times 3$, but as a 2D matrix of size $m_1 \times 3m_2$. Therefore, we will switch to learning orthonormal basis pairs, with bases of size $U \in \mathcal{O}(m_1)$ and $V \in \mathcal{O}(3m_2)$, as opposed to learning triples of the form $U \in \mathcal{O}(m_1)$, $V \in \mathcal{O}(m_2)$ and $W \in \mathcal{O}(3)$ as in Section 2.2. Learning orthonormal bases from the space $\mathcal{O}(3m_2)$ provides greater representational power as compared to learning bases that are constrained to the specific form $V \otimes W$ (as in Equation 16 and equivalently in Section 2.2) where $V \in \mathcal{O}(m_2)$ and $W \in \mathcal{O}(3)$ (since we have $V \otimes W \in \mathcal{O}(m_2) \times \mathcal{O}(3)$). For this reason, we employ this 2D representation of a color image patch. Note that in such a representation, the learning of the column space of the patch matrices as well as the color space occurs together in a coupled manner.

In the extreme case, Equation 16 can be expressed as

$$X_{(1)}^{vec} = (U \otimes V \otimes W)^T S_{(1)}^{vec} \tag{17}$$

where the superscript ‘vec’ stands for the representation of an array of size $m_1 \times m_2 \times 3$ as a vector of size $3m_1m_2 \times 1$. This simple vectorial representation of the patch affords good representational capability, as the orthonormal bases will turn out to have size $3m_1m_2 \times 3m_1m_2$. The matrix based representation of the image patch followed in our work certainly gives rise to an artificial row-column segregation, unlike a vectorial patch representation. However, vectorial representations lead to a computationally expensive algorithm and lead to larger dictionary storage. We believe, therefore, that the $m_1 \times 3m_2$ representation yields the best tradeoff between representational capability and algorithm efficiency (optimization speed and dictionary size). We have also observed that patches of this size yield the best compression performance (see experimental results). From color image patches of size $m_1 \times m_2 \times 3$ and assuming K bases/basis-pairs/basis-triples are learned, the dictionary storage sizes for the following three patch representations are shown in Table 1: (1) vectors of size $3m_1m_2 \times 1$, (2) 2D arrays of size $m_1 \times 3m_2$, (3) 3D arrays of size $m_1 \times m_2 \times 3$.

3.2 Improving the representational power of the dictionary

In our previous techniques (see equations 1 and 9), we restricted ourselves to learning K basis-pairs or basis-triples. However, this force-fits a coupling between the U and V bases in Equation 1, or between the U, V and W bases in Equation 9. Instead, we can learn K bases $\{U_a\}$, $1 \leq a \leq K$ and $\{V_b\}$, $1 \leq b \leq K$ just as before, but now allow all K^2 pairings. Similarly for the 3D case, we could learn K bases $\{U_a\}$, $\{V_b\}$ and $\{W_c\}$ where $1 \leq a, b, c \leq K$ and allow all K^3 triples. This allows for a greater variety of basis-pairs to be represented without increasing the storage required for the dictionary. We shall refer to this variant as the ‘2D method with cross-indices’ and the one from the previous section as the ‘2D method without cross-indices’.

3.3 Objective function

We now present the objective function for learning bases by combining the ideas from the two previous sections. Consider the training (color, RGB) images divided into N non-overlapping patches $\{P_i\}$ of size $n \times n \times 3$. We shall represent each patch P_i as a matrix of size $n \times 3n$ and learn K orthonormal matrix pairs of the form $\{(U_a, V_b)\}$ where $1 \leq a, b \leq K = \sqrt{K}$ and $U_a \in \mathcal{O}(n)$, $V_b \in \mathcal{O}(3n)$. The objective function is given as follows:

$$E(\{U_a, V_b, S_{iab}, M_{iab}\}) = \sum_{i=1}^{N} \sum_{a=1}^{K} \sum_{b=1}^{K} M_{iab} \|P_i - U_a S_{iab} V_b^T\|^2 \tag{18}$$

subject to the following constraints:

$$\forall (a,b)\ U_a^T U_a = V_b^T V_b = I \tag{19}$$

$$\forall (i,a,b)\ \|S_{iab}\|_0 \leq T \tag{20}$$

$$\forall i\ \sum_a \sum_b M_{iab} = 1 \text{ and } \forall (i,a,b)\ M_{iab} \in \{0,1\} \tag{21}$$

where S_{iab} is the projection of the i^{th} patch onto the $(a,b)^{th}$ basis pair, and M_{iab} indicates a fuzzy membership of the i^{th} patch onto the $(a,b)^{th}$ basis pair. Starting with random orthonormal bases for all $\{U_a\}$,$\{V_b\}$ and $M_{iab} = \frac{1}{K}\ \forall (i,a,b)$, the matrix S_{iab} is computed using $S_{iab} = U_a^T P_i V_b$ and its $3m_1m_2 - T$ elements with smallest magnitude are nullified. The updates for U_a are given as follows:

$$Z_{1a} = \sum_{i,b} M_{iab} P_i V_b S_{iab}^T \tag{22}$$

$$U_a = Z_{1a}(Z_{1a}^T Z_{1a})^{-\frac{1}{2}} \tag{23}$$

$$U_a = (\Gamma_{1a} \Psi \Upsilon_{1a}^T)((\Gamma_{1a} \Psi \Upsilon_{1a}^T)^T (\Gamma_{1a} \Psi \Upsilon_{1a}^T))^{-\frac{1}{2}} = \Gamma_{1a} \Upsilon_{1a}^T. \tag{24}$$

where the SVD of Z_{1a} will give us $Z_{1a} = \Gamma_{1a} \Psi \Upsilon_{1a}^T$ with Γ_{1a} and Υ_{1a} being orthonormal matrices and Ψ being a diagonal matrix. The bases V_b are updated in a similar manner. The membership values are obtained by:

$$M_{iab} = \frac{e^{-\beta \|P_i - U_a S_{iab} V_b^T\|^2}}{\sum_{c=1}^{K} \sum_{d=1}^{K} e^{-\beta \|P_i - U_c S_{icd} V_d^T\|^2}}. \tag{25}$$

The matrices $\{S_{iab}, U_a, V_b\}$ and M_{iab} are then updated sequentially following one another in a deterministic annealing framework until M_{iab} turns out to be (nearly) binary.

4 Experimental Results

We now present our experimental results for compression of color images. The first experiment was performed on a subset of 54 images from the CMU-PIE database[2]. The CMU-PIE database contains images of several people against cluttered backgrounds with a large variation in pose, illumination, facial expression and occlusions created by spectacles. All the images are available in an uncompressed (.ppm) format, and their size is 631×467 pixels. We chose 54 images belonging to one and the same person (labelled in the database as ‘04055.jpg’), and used patches from exactly

[2] `http://vasc.ri.cmu.edu/idb/html/face/index.html`

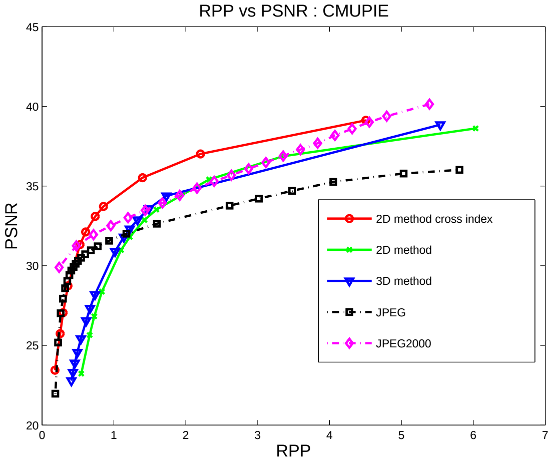

Figure 1: RPP-PSNR curves for several competing methods: (1) 3D method, (2) 2D method without cross-indices, (3) 2D method with cross-indices, (4) JPEG, (5) JPEG-2000.

one image for training. The remaining 53 images were used for testing. During training, we used a sparsity factor of $T_0 = 10$ and used patches of size $12 \times 12 \times 3$ for all three variants of our dictionary learning method: (1) the 3D method from Section 2.2, (2) the 2D method from Section 3.1, and (3) the 3D method from Section 3.2 with cross-indices. For the 3D method, we set $K = 100$, whereas for the 2D methods, we set $K = 20$, which produced dictionaries of nearly the same size. (Note that the dictionary size for the 3D case was $100 \times (12^2 + 12^2 + 3^2) = 29700$ and $20 \times (12^2 + 36^2) = 28800$ for the 2D case. Also see Table 1.) The RPP-PSNR curves for all these methods are shown in Figure 1. The performance of the 2D method without cross indices was superior to the 3D method for PSNR values of 34 to 38. The 2D method with the cross indices, however, produced results that were clearly superior to the other two techniques. These results were pitted against JPEG and the Jasper implementation for JPEG2000 [1]. Between RPP values from 0.5 to 4.5, the PSNR values of the 2D method with cross-indices were distinctly superior to those produced by JPEG as well as Jasper.

The second experiment was conducted on a more general database, consisting of 150 images taken from the Uncompressed Colour Image Database (UCID), Version 2[3]. The database contains images of sceneries and man-made objects, all of size around 512×384. We chosen around 20-25 pictures each of seven different categories, using exactly one image per category for training and the remaining for testing. One sample training and test image each for five different categories are shown in Figure 2. For training, we used a sparsity parameter $T_0 = 20$ and patches of size $12 \times 12 \times 3$. We performed two experiments: one on images of the original resolution, and the other on their downsampled versions (of size 256×192). The RPP-PSNR curves are shown in Figure 3. We should sample reconstructions of one of the images under four different error values in Figure 4. For this experiment, our method produced curves clearly superior to JPEG beyond RPP of around 3.8 (on a scale from 0 to 24), though Jasper was the clear winner. We wish to emphasize here that Jasper (JPEG2000) employs a highly advanced quantization scheme developed jointly by several people over a period of many years. In [6], we have demonstrated that wavelet based compression with a simple quantization scheme is unable to produce such good results and our learning-based method is superior to such an implementation. Some images from the UCID database, such as

[3] `http://www-staff.lboro.ac.uk/~cogs/datasets/UCID/ucid.html`

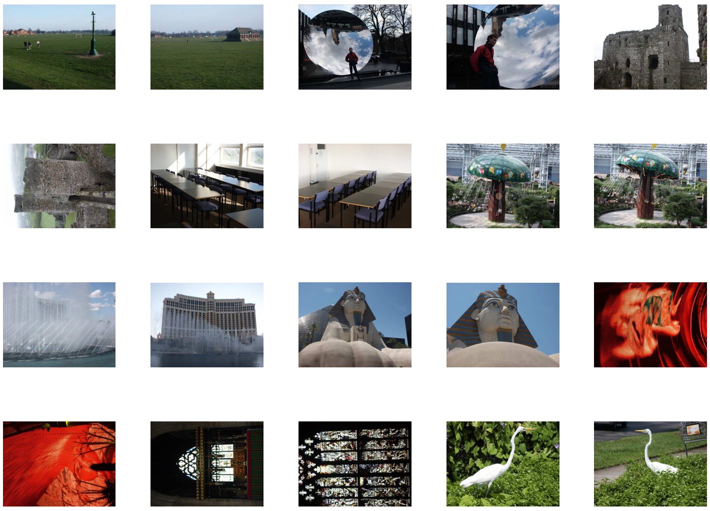

Figure 2: Images from five different categories of the UCID database. Zoom into the pdf file for a better view.

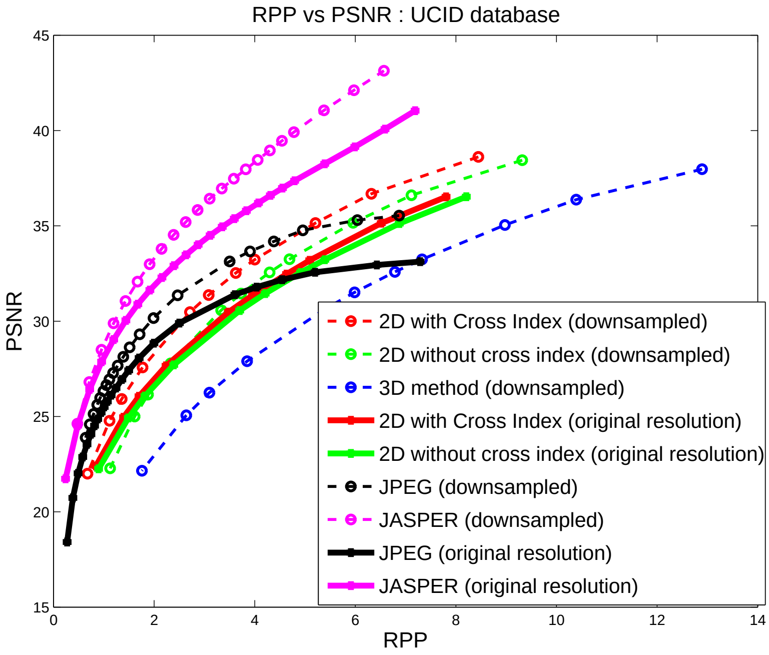

Figure 3: RPP-PSNR curves for several competing methods on UCID database: For downsampled images, (1) 2D method without cross-indices, (2) 2D method with cross-indices, (3) 3D method, (4) JPEG and (5) JASPER. For images of the original resolution, (6) 2D method without cross-indices, (7) 2D method with cross-indices, (8) JPEG and (9) JASPER.

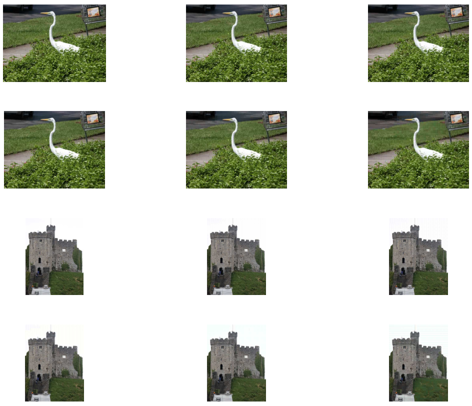

Figure 4: Left to right, top to bottom: original image, reconstructions under errors of 0.0002, 0.0003, 0.0006, 0.001, 0.003. Zoom into the pdf file for a better view. Avg. RPP values for first image: 10.46, 8.81, 6.35, 4.83, 2.38 respectively. Avg. RPP values for second image: 6.34, 5.069, 3.28, 2.28, 0.88 respectively.

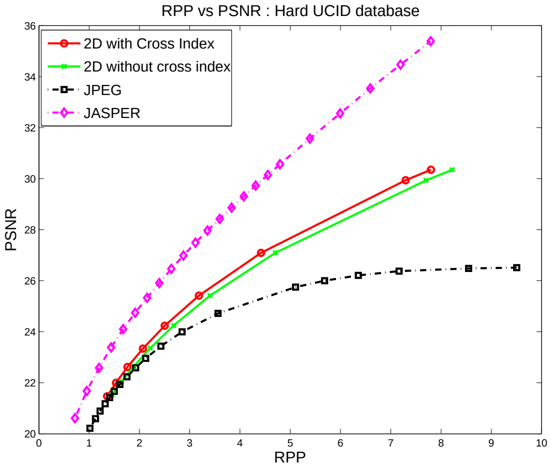

Figure 5: RPP-PSNR curves for several competing methods on challenging images from the UCID database (for images of the original resolution): (1) 2D method without cross-indices, (2) 2D method with cross-indices, (3) JPEG and (4) JASPER.

the last six images from Figure 2 (zoom into the figure in the pdf file for a better view), contain a huge amount of textured regions and are challenging for any compression algorithm. We present PSNR/RPP curves on a subset of 25 such images in Figure 5.

5 Conclusions

In this paper, we have presented two variants of our machine learning based techniques for color image compression. The new variants use a matrix based representation of the color image patch. We show that this representation produces superior results to the original 3D method. We also clearly demonstrate the benefits of using cross-indices for the learned bases as it allows for a greater variety of coupling between the row-row and column-column bases for the patches, without increasing dictionary size. Our algorithms are tested on two large databases and are pitted against JPEG and Jasper with promising results. Future work will involve a deeper look at the quantization scheme employed in our algorithms for improving the RPP/PSNR curves further, and method for taking care of the block artifacts that occur due to independent compression of each patch.

We wish to highlight two subtle aspects of our methods. Firstly, our methods can be employed very effectively for compression of image databases, by training on patches from subsets of the images, or randomly selected patches from all the images. For applications such as these, the training parameters can even be selected on a trial and error basis, settling in on those specific values that produced the best RPP-PSNR curves. These values can then be used for generating the compressed database along with the dictionaries. Secondly, our matrix based representation allows for significantly faster training than methods such as [2], [4] which use a vector-based representation, since our bases are of smaller size.

References

[1] M. Adams and R. Ward. Jasper: A portable flexible open-source software toolkit for image coding/processing. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 241–244, 2004.

[2] M. Aharon, M. Elad, and A. Bruckstein. The K-SVD: an algorithm for designing of overcomplete dictionaries for sparse representation. *IEEE Trans. Signal Process.*, 54(11):4311–4322, 2006.

[3] L. de Lathauwer. *Signal Processing Based on Multilinear Algebra*. PhD thesis, Katholieke Universiteit Leuven, Belgium, 1997.

[4] A. Ferreira and M. Figueiredo. On the use of independent component analysis for image compression. *Signal Processing: Image Communication*, 21(5):378–389, 2006.

[5] K. Gurumoorthy, A. Rajwade, A. Banerjee, and A. Rangarajan. Beyond SVD: Sparse projections onto exemplar orthonormal bases for compact image representation. In *Int. Conf. Pattern Recognition*, pages 1–4, 2008.

[6] K. Gurumoorthy, A. Rajwade, A. Banerjee, and A. Rangarajan. A method for compact image representation using sparse matrix and tensor projections onto exemplar orthonormal bases. *IEEE Trans. Image Process.*, 19(2):322–334, 2010.

[7] R. Hathaway. Another interpretation of the EM algorithm for mixture distributions. *Statistics and Probability Letters*, 4:5356, 1986.