A Novel Color Image Compression Method Using Eigenimages

A. Abadpour[1] S. Kasaei[2]

[1] MSc. Student, Sharif University of Technology, Department of Mathematics ScienceP.O. Box 11155-9517, Tehran, Iran

Email: abadpour@math.sharif.edu

[2] Associate Professor, Sharif University of Technology, Department of Computer EngineeringP.O. Box 11155-9517, Tehran, Iran

Email: skasaei@sharif.edu

Abstract:

Since the birth of multi–spectral imaging techniques, there has been a tendency to consider and process this new type of data as a set of parallel gray–scale images, instead of an ensemble of an *n*–D realization. Although, even now, some researchers make the same assumption, it is proved that using vector geometries leads to better results. In this paper, first a method is proposed to extract the eigenimages from a color image. Then, using the energy compaction of the proposed method, a new color image compression method is proposed and analyzed. The proposed compression method, which uses vector–based operations, applies a grayscale compression algorithm on the eigenimages. Experimental results show that, at the same bandwidth, the proposed method produce 3.6dB and 1.9dB enhancement in the quality, compared to JPEG and JPEG2000, respectively.

Keywords: Principle Component Analysis, Color Image Processing, Image Compression, Eigenimages.

Submission date: Aug. 28, 2005

Acceptance date: July 4, 2008

Corresponding author: S. Kasaei

Address of corresponding author: Department of Electrical Engineering, Sharif University of Technology Tehran, Iran

1. Introduction

Although, color is one of the most important tools for object discrimination by human observers, it is overlooked in the past [18]. In fact, discarding the intrinsic characteristics of color images, as vector geometries, some researchers have considered color images as parallel gray–scale images (*e.g.*, see [7]). However, it has been proved that the Principle Component Analysis (PCA) is an appropriate vector–based descriptor for natural color images [8]. In this paper, we show how using PCA yields a vector–based image compression method which outperforms the conventional ones. In the next paragraph, we briefly discuss some of the relevant developments.

In 1988, *Klinker*, *Shafer*, and *Kanade* presented a novel approach to measuring the highlights in color images [14]. In that work, they developed a proper model for the reflected light from an arbitrary point of a dielectric object. In 1990, they applied their approach to color image understanding [15]. However, more than a decade passed since the idea was successfully incorporated into a practical algorithm. In 2003, without paying too much attention to the theoretical aspects, *Cheng* and *Hsia* used the *principal component analysis* (PCA) for color image processing [8]. Then, in 2004, *Nikolaev* and *Nikolayev* started the work again from the theory and proved that the PCA is a proper tool for color image processing [17]. The next necessary step had been introduced in 1991, when *Turk* and *Pentland* proposed their eigenface method [23], in a completely different context. There, they developed a novel idea which connected the eigenproblems in the color domain and the spatial domain. Although, there is this rich theoretical background for the linear local models of color, it is quite common to see research procedures which are based on the old color space paradigm, even published in 2005. For a more comprehensive discussion of this topic refer to [2].

The early approach to color image compression is based on decorrelating the color planes using linear or nonlinear invertible coordinate transformations such as *YCbCr* [1], *YIQ*[5] or *YUV* [22]. Then, the color planes will independently undergo a standard gray–scale compression method, such as Differential Pulse Code Modulation (DPCM) [16] or transform coding [19] (also see [10]). This approach is inefficient, because none of the available color spaces is able to completely decorrelate the color planes in an arbitrary image.

In [9], using the PCA in the neighboring pixels, the author discusses the idea of separating the spatial and the spectral compression stages. As the paper shows, the maximum theoretical compression ratio for an ideal spectral compression method is 1:3. The main shortcoming of the method in [9] is neglecting the fact that in non–homogeneous regions, the PCA does not perform energy compaction [6]. In [6], the author combines the spatial and the spectral information to reach a higher compression ratio. Although, the method is based on expensive computation, the Peak Signal to Noise Ratio (PSNR) results are not acceptable. The main shortcoming of the method in [6] is the block artifacts produced after decompression.

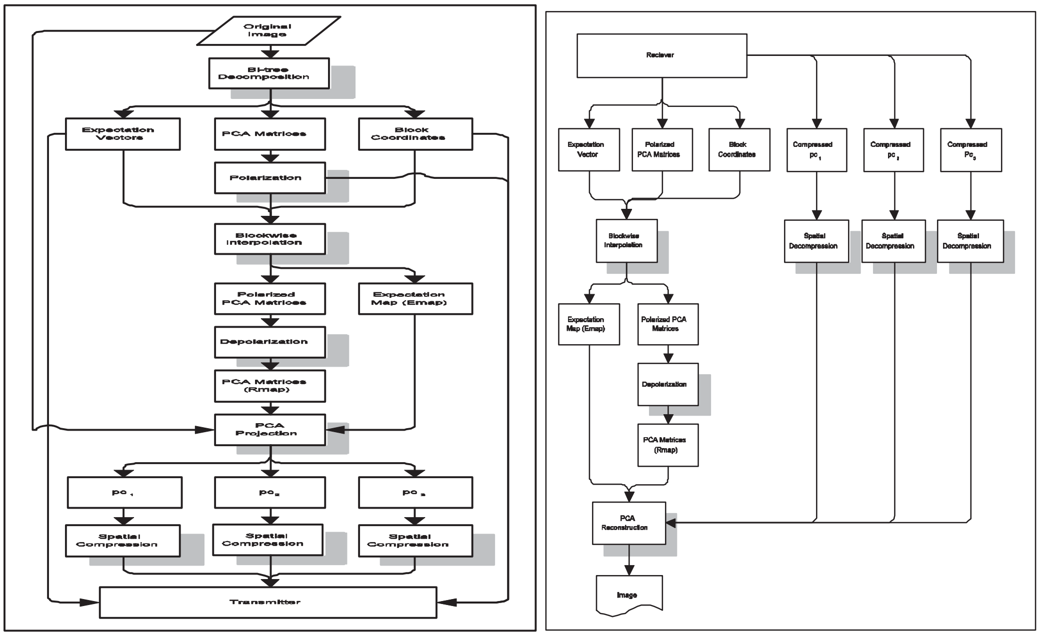
In this paper, we apply a tree decomposition using a novel color homogeneity criterion to cut a given image into homogeneous patches. These patches will be analyzed using mathematical tools proposed in this paper to extract eigenimages. The main contribution of this paper is a new color image compression method using the proposed eigenimage extraction technique. Experimental results are also carried to analyze the performance of the proposed method and also to compare it to the available approaches.

Quad–tree decomposition is the well–known method for splitting an image into homogeneous sub-blocks, resulting in a very coarse but fast segmentation [20]. To use the quad–tree decomposition, a suitable homogeneity criterion is needed. In [3], the authors proposed to use the error made by neglecting the two least important principal components (the second and the third) as a likelihood measure, called the Linear Partial Reconstruction Error (LPRE). The LPRE distance of vector $\vec{c}$ to cluster r is defined as $r_r(\vec{c}) = \left\|\vec{v}^T(\vec{c}-\vec{\eta})\vec{v}-(\vec{c}-\vec{\eta})\right\|$, where $\vec{v}$ denotes the direction of the first principal component and $\|x\|$ is the normalized L_1 norm, $\|x\| = \sum_{i=1}^{N}|x|/N$. In [3], the authors proposed to use the following stochastic margin to compute the homogeneity of the selected region r, $\|f\|_{r,p} = \arg_e\left\{P_{\vec{x}\in r}\left\{f(\vec{x})\le e\right\}\ge p\right\}$, where p is the inclusion percentage and $P_{\vec{x}\in r}\left\{f(\vec{x})\le e\right\}$ denotes the probability of x being less than or equal to e. It is proved that $\|\tau_r\|_{r,p}$ is a proper homogeneity criterion for quad–tree decomposition [4]. The comparison of the LPRE–based homogeneity criterion with the Euclidean and Mahalanobis measures has proved its superiority [4].

2. Proposed Algorithms

2.1. Basis Vector Polarization

Consider the space R^n and a set of n basis vectors $\vec{v_i}, i = 1, ..., n$. Storing this set of vectors needs n^2 units of memory (when neglecting the redundancy among data). Having in mind that a set of basis vectors is an orthonormal set, the actual needed memory can be reduced. In fact, a set of basis vectors of R^n is a member of R^{n^2}, with n constraints of normality ($\|\vec{v_i}\| = 1, i = 1, ..., n$) and $\frac{n(n-1)}{2}$ constraints of orthogonality ($\vec{v_i} \perp \vec{v_j}, i, j = 1, ..., n, i \neq j$). Thus, the above–mentioned set



(a) (b)

Fig . 1: (a) Flowchart of the proposed color image compression method. (b) Flowchart of the proposed color image decompression method.

of basis vectors is an unconstrained member of an m–D space, with m equal to $n^2-n-n(n-1)/2$ or $\frac{n(n-1)}{2}$. Thus, storing a set of the basis vectors of R^n in $\frac{n(n-1)}{2}$ memory cells contains zero redundancy. To make this representation unique, it is crucial to make the set of basis vectors right–rotating (RR). In 2–D spaces, RR means $(\overrightarrow{v_1}\times\overrightarrow{v_2}).j>0$ where, $\times$ and $\cdot$ denote the outer and the inner products, respectively. In 3–D spaces, RR means $(\overrightarrow{v_1}\times\overrightarrow{v_2}).\overrightarrow{v_3}>0$. Setting $n = 2$ leads to $m = 1$, which means that any set of RR basis vectors in the *xy* plane can be specified uniquely by a single parameter (the angle). Similarly, the case of $n = 3$ results in $m = 3$, which is used in this paper. We will prove that the parameters in the 3–D case are angular too. Thus, we call this method of representing a set of basis vectors, the polarization method. Here we propose a method for finding these angles.

Consider the three RR vectors $\vec{v_1},\vec{v_2},\vec{v_3}$ in R^3. We define the angles θ, φ, and ψ as a manipulated version of the well–known set of Euler angles. Using $\overrightarrow{v^p}$ as the projection of $\vec{v}$ on plane p (*e.g.*, $\overrightarrow{v_1^{xy}}$), the three angles are defined as:

$$\begin{cases}\theta=\angle(\overrightarrow{v_1^{xy}},[1,0]^T)\\ \phi=\angle((R_\theta^{xy}\overrightarrow{v_1})^{yz},[1,0]^T)\\ \psi=\angle(R_\theta^{xz}R_\theta^{xy}\overrightarrow{v_2})^{yz},[1,0]^T)\end{cases}\qquad(1)$$

where, $\angle(\vec{v},\vec{u})$ denotes the angle between two vectors $\vec{v},\vec{u}\in R^2$, computed as $\angle(\vec{v},\vec{u})=\mathrm{sgn}((\vec{v}\times\vec{u}).j)\cos^{-1}\vec{v}.\vec{u}/\|\vec{v}\|\|\vec{u}\|$ where $sgn(x)$ is the signum function. Also, R_α^P is the 3×3 matrix of α radians rotated counter–clock–wise in the p plane. Composing the 3×3 matrix V with $\vec{v_i}$ as its i–th column, we get $R_\psi^{yz}R_\phi^{xz}R_\theta^{xy}V=I.As(R_\alpha^P)^{-1}=R_{-\alpha}^P$ we have $V=R_{-\theta}^{xy}R_{-\psi}^{yz}R_{-\phi}^{xz}$. While (1) computes the three angles θ, φ, and ψ out of the basis vectors (polarization), the above equation reproduces the base from θ, φ, and ψ angles (depolarization).

2.2. Block–wise Interpolation

Consider a partition of the $N_W\times N_H$ as a set of rectangular regions $\{r_i|\,i=1,...,n\}$, with corresponding (given) values of $\{\lambda_i|\,i=1,...,n\}$, satisfying $\forall i,\forall\vec{c}\in r_i,\ f(\vec{c})\approx\lambda_i$ for an arbitrary function $f:R^2\to R$. The problem is to find $\tilde{f}$ as a good approximation of f. We address this problem as a block–wise interpolation of the set $\{(r_i,\lambda_i)|\,i=1,...,n\}$. Note that in the case that the partition is a conventional rectangular grid, the problem reduces to an ordinary 2–D interpolation task. Here, we use a reformulated version of the well–known low–pass Butterworth filter as the interpolation kernel,

$$\begin{cases}B_{\tau,N}(x)=(1+(\frac{x}{\tau})^{2N})^{-\frac{1}{2}}\\ N=rnd(\log_{\frac{a}{b}}(\frac{\beta\sqrt{1-\alpha^2}}{\alpha\sqrt{1-\beta^2}}))\\ \tau=a\sqrt[2N]{\frac{\alpha^2}{1-\alpha^2}}\end{cases}\qquad(2)$$

The function $B_{\tau,N}(0)$ satisfies the conditions of $B_{\tau,N}(a)=a$ and $B_{\tau,N}(b)=\beta$. The 2–D version of this function is defined as $B_{\tau,N,w,h}(x,y)=B_{w\tau,N}(x)B_{h\tau,N}(y)$ where w and h control the spread of the function in the x and y directions, respectively. Figure 4 shows the typical shape of the function $B_{\tau,N}^{w,h}(x,y)$ with a = 0.9, α = 0.7, b = 1, β = 0.5, and w = h = 16.

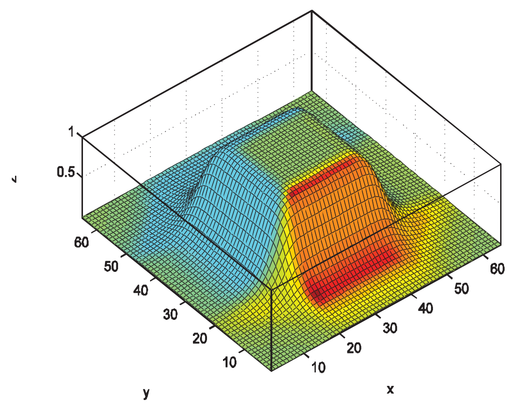


Fig.4: The typical shape of the proposed interpolation kernel, $B_{\tau,N}^{w,h}(x,y)$.

Assuming that the region r_i is centered on (x_i,y_i) while its height and width are ω_i and h_i, respectively, we propose the function $\tilde{f}$ to be defined as:

$$\tilde{f}(x,y)=\frac{\sum_{i=1}^{N}\lambda_i B_{\tau,N,\frac{\omega_i}{2},\frac{h_i}{2}}(x-x_i,y-y_i)}{\sum_{i=1}^{N} B_{\tau,N,\frac{\omega_i}{2},\frac{h_i}{2}}(x-x_i,y-y_i)} \tag{3}$$

Note that $\tilde{f}(x,y)$, is a smooth version of the initial step case function $\forall[x,y]^T\in r_i:f_o(x,y)=\lambda_i$. Also, by setting proper values of the parameters a, b, α, and β, the function $\tilde{f}(x,y)$ will satisfy the constraints. The proper set of the parameters must force $B_{\tau,N}^{\frac{\omega_i}{2},\frac{h_i}{2}}(x-x_i,y-y_i)$ to become nearly one in entire r_i (except for the borders neighborhoods) and also to prevent r_i to intrude the interior with points of r_j, for $i\neq j$. Selecting a value near unity but smaller than it for a and α, limits the decline of the ceil of the function, while setting b = 1 and a not too big value for β (*e.g.*, 0.4) controls the effects of neighbor regions on each other. Note that setting $a=1^-$, $\alpha=1$, $b=1^+$, and $\beta=0$, is the marginal choice leading to no smoothing (the same as $f_\circ$).

As the generalization of the block–wise interpolation problem, assume the set of regions $\{(r_i;\lambda_{ij})|i=1,\ldots,n,j=1,\ldots,m\}$, satisfying $\lambda_{ij}=\arg_\lambda(\forall\vec{c}\in r_i,f_j(\vec{c})\approx\lambda)$ for a set of arbitrary functions $f_i:R^2\to R, i=1,\ldots,m$. In a similar manner with (3), we propose:

$$\tilde{f}_j(x,y)=\frac{\sum_{i=1}^{N}\lambda_{ij} B_{\tau,N,\frac{\omega_i}{2},\frac{h_i}{2}}(x-x_i,y-y_i)}{\sum_{i=1}^{N} B_{\tau,N,\frac{\omega_i}{2},\frac{h_i}{2}}(x-x_i,y-y_i)} \tag{4}$$

Here, because the set of the base regions for all $\tilde{f}_j$ are the same, the total performance is increased by computing $B_{\tau,N}^{\frac{\omega_i}{2},\frac{h_i}{2}}(x-x_i,y-y_i)$ for each value of i, just once. Then, the problem reduces to m times computation of a weighted average.

When working in the polar coordinates, because of the 2π discontinuity, ordinary algebraic operations on the variables lead to spurious results (for example $\frac{0+2\pi}{2}=\pi$, while the average of 0 radians and 2π radians equals $0\equiv 2\pi$ radians). To overcome this problem, we propose a new method: for the given problem $\{(r_i,\theta_i)|i=1,\ldots,n\}$, solve the problem $\{(r_i,\cos\theta_i,\sin\theta_i)|i=1,\ldots,n\}$ and then find θ_i using ordinary trigonometric methods. Note that interpolating both $\sin\theta_i$ and $\cos\theta_i$ is performed to avoid ambiguity in the polar plane.

2.3. The Eigenimage

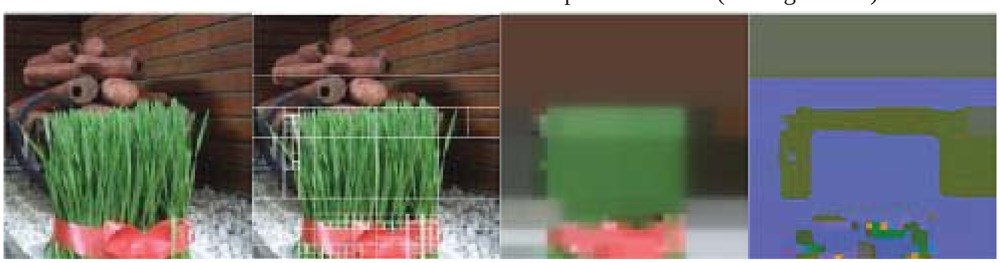
Consider the PCA matrix, V_r, and the expectation vector, $\vec{\eta}_r$, corresponding to the homogeneous cluster r. Then, for the color vector $\vec{c}$ belonging to r we get the PCA coordinates as $\vec{c}^{\,T}=V_r^{-1}(\vec{c}-\vec{\eta}_r)$. Assume that we can somehow find the color cluster $r_{\vec{c}}$ for each color vector $\vec{c}$, where $r_{\vec{c}}$ describes the color mood of $\vec{c}$, in the sense that $\vec{c}^{\,T}=V_{r_{\vec{c}}}^{-1}(\vec{c}-\vec{\eta}_r)$ satisfies $\sigma_{c'_1}\gg\sigma_{c'_2}\gg\sigma_{c'_3}$, where $\vec{c}^{\,T}=[c'_1,c'_2,c'_3]^T$. We denote the 2–D arrays made by c'_1, c'_2, and c'_3 as the pc_1, pc_2, and pc_3, respectively. The original image can be perfectly reconstructed using these three channels, except for the numerical errors as $\vec{c}\approx\vec{c_3}=V_{r_{\vec{c}}}\vec{c}^{\,T}+\vec{\eta}_{r_{\vec{c}}}$. It is proved in [3] that for homogeneous swatches, neglecting pc_3 (or even both pc_2 and pc_3), gives good approximations of the original image. Here, we generalize the approach. Note that the perfect reconstruction of the image from all eigenimages does not rely on the energy compaction, while the partial reconstructions defined as $\vec{c_2}=V_{r\vec{c}}[c'_1,c'_2,0]^T+\vec{\eta}_{r\vec{c}}$ and $\vec{c_1}=V_{r\vec{c}}[c'_1,0,0]^T+\vec{\eta}_{r\vec{c}}$ do rely on it. Although, the above scheme gives a 1–D representation of a given color image, if the computation of $V_{r\vec{c}}$ and $\vec{\eta}_{r\vec{c}}$ gets expensive the scheme although being theoretically promising it actually is not applicable. Thus, we seek for a method for describing $V_{r\vec{c}}$ and $\vec{\eta}_{r\vec{c}}$ in a simple way. The case for defining $r_{\vec{c}} = N_{\vec{c}}$ (the neighborhood pixels) is automatically rejected (because to compute $V_{r\vec{c}}$ and $\vec{\eta}_{r\vec{c}}$ we need all the neighborhood points of $\vec{c}$ leading to a high redundancy and computation cost).

Here, we propose a fast and reliable method to compute the corresponding $V_{r\vec{c}}$ and $\vec{\eta}_{r\vec{c}}$ for all image pixels. Assume feeding the given image I to the bi–tree (or equivalently the quad–tree) decomposition method. The output of the decomposition method is the matrix Υ containing the coordinates of r_i along with the expectation matrix $\vec{\eta}_i$ and the polarized version of the PCA matrix $(\theta_i, \phi_i, \psi_i)$. Storing this portion of the Υ matrix needs $10n$ bytes. For ordinary values of $n \approx 200$ in a 512×512 image, Υ will take about $\frac{1}{400}$ of the original image data. Now, assume solving the problem $\{(r_i;\zeta_i)|i=1,\ldots,n\}$ using the block–wise interpolation, where ζ_i is the row vector containing $\bar{\eta}_{i1}, \bar{\eta}_{i2}, \bar{\eta}_{i3}, \theta_i, \phi_i$ and ψ_i. Note that the three values of θ_i, ϕ_i and ψ_i are of angular type. Assume the solutions of the problem as the functions $\bar{\eta}_{i1}, \bar{\eta}_{i2}, \bar{\eta}_{i3}, \theta_i, \phi_i$ and ψ_i. Now we compute the functions $\tilde{\bar{\eta}} : R^2 \to R^3$ and $\bar{V} : R^2 \to R^9$, as the value of the expectation vector and the PCA matrix in each pixel, respectively. This leads to the computation of the three eigenimages pc_1, pc_2 and pc_3. We call the function $\tilde{\bar{\eta}} : R^2 \to R^3$ as the expectation map (Emap) and the polarized version of $\bar{V} : R^2 \to R^9$ as the rotation map (Rmap), respectively. As the PCA theory states [11], we expect the standard deviation of the three planes to be in descending order.

From linear algebra we know that for orthonormal matrices V_r the eigenimages satisfy $\sigma_{pc1}^2 + \sigma_{pc2}^2 + \sigma_{pc3}^2 = \sigma_r^2 + \sigma_g^2 + \sigma_b^2$. Thus, $\kappa_i = \sigma_{pc1}^2 / (\sigma_{pc1}^2 + \sigma_{pc2}^2 + \sigma_{pc3}^2)$ shows the amount of information available in the i–th eigenimage, satisfying $\kappa_1 + \kappa_2 + \kappa_3 = \kappa_r + \kappa_g + \kappa_b = 1$.

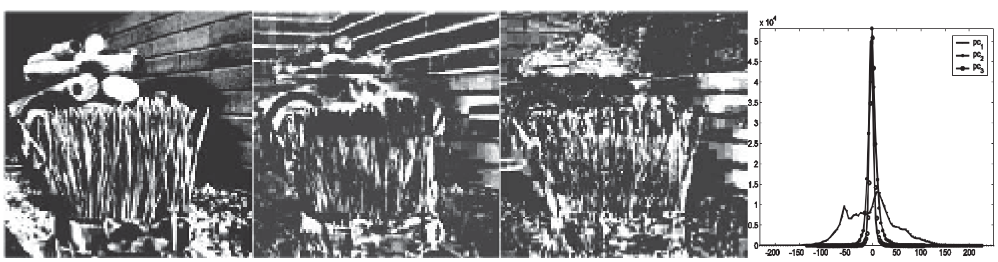
2.4. Color Image Compression

Consider the image I and its corresponding eigenimages pc_1, pc_2 and pc_3. Due to the energy compaction condition, this scheme is actually an spectral image compression method. Reconstructing the image using just one or two eigenimage(s) gives the compression ratios of 3:1 (the theoretical margin) and $\frac{3}{2}:1$, respectively. To add the spatial compression ability to the proposed method, we use the PU–PLVQ gray–scale image compression technique [12] for each eigenimage with different compression ratios (see Figure 1–a).



(a) (b) (c) (d)

Fig . 2: (a) Original image adopted from [21], (b) result of the bi–tree decomposition method, (c) Emap, and (d) Rmap.



(a) (b) (c) (d)

Fig . 3: Eigenimages of the image shown in Figure 2–(a), (a) pc_1, (b) pc_2, (c) pc_3, (d) Corresponding Histograms

As Figure 1–a shows, the transmitted information contains the compressed versions of the pc_1, pc_2 and pc_3, along with the Υ, α, a, β, and b (for block–wise interpolation). Assume that the image to be compressed is a $H \times W$ color image, decomposed into n blocks. The total amount of information to be sent equals: $10n$ bytes for storing $x_{i1}, x_{i2}, y_{i1}, y_{i2}, \eta_{i1}, \eta_{i2}, \eta_{i3}, \theta_1, \phi_1$ and ψ_1 plus $WH(\lambda_1^{-1}+\lambda_2^{-1}+\lambda_3^{-1})$ for storing pc_1, pc_2 and pc_3 eigenimages compressed with compression ratios of λ_1, λ_2 and λ_3, respectively (where $\lambda_1 > \lambda_2 > \lambda_3$). Thus, the total compression ratio equals $\lambda = 3(\lambda_1^{-1}+\lambda_2^{-1}+\lambda_3^{-1}+\frac{10n}{WH})^{-1}$. A nominal value of $\lambda_2 = \lambda_1$ and $\lambda_3 = \infty$ leads to $\lambda \simeq 1.5\lambda_1$. Note that using a pure spatial compression, all three channels must be compressed with almost the same compression ratios, resulting in a total compression ratio of $\tilde{\lambda} \simeq 3WH(\frac{WH}{\lambda_1}+\frac{WH}{\lambda_1}+\frac{WH}{\lambda_1})^{-1} = \lambda_1$. As shown in Figure 1–b, in the decompression process the Emap and the Rmap are computed just like what performed in the encoding process. Using these information along with the decoded versions of pc_1, pc_2 and pc_3, the original image is reconstructed.

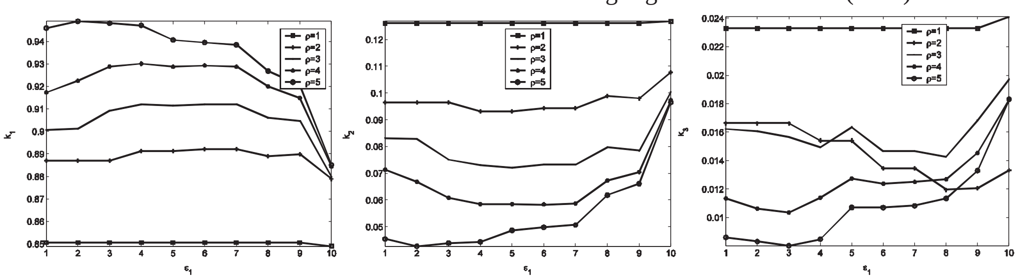
3. Experimental Results

The proposed algorithms are developed in MATLAB 6.5, on an 1100 MHz Pentium III personal computer with 256*MB* of RAM. The codes are available online at http://math.sharif.edu/~abadpour. For a typical 512×512 image it takes 8.3 seconds to extract the eigenimages. Then, another 4.6 seconds are needed to reconstruct the image. Adding the less than one second needed to do the spatial compression the total operations take slightly less than 16 seconds for a typical image. Note that these values are measured using MATLAB code and can be further reduced if the code is to be implemented using a higher–level programming language.

A database of color images (140 samples) including the standard images of *Lena*, *Mandrill*, *Peppers*, and *Couple* and also some professional color photographs [21] is used. All images have the size of 512×512, in *RGB* color space, and compressed using standard JPEG compression with qualities above 95. Prior the operation all the images are converted to the standard BMP format.

3.1. Block–wise Interpolation

Figure 7–(a) shows a sample problem set given to the proposed block–wise Interpolation. Figure 7–(b) illustrates the corresponding $\tilde{f}$. In this sample the resulting Signal to Noise Ratio (SNR) is more than 22*dB*.



(a) (b) (c)

Fig . 5: Distribution of the energy between the three eigenimages for different values of ε_1 and Q. (a) κ_{pc1}. (b) κ_{pc2}. (c) κ_{pc3}.

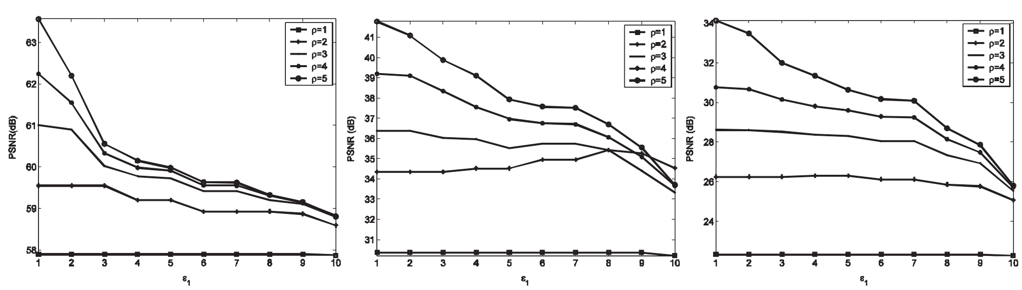
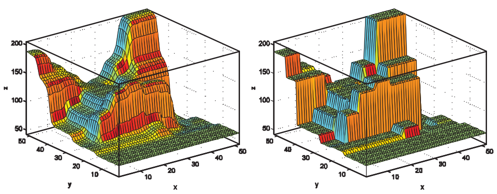


Fig . 6: PSNR values of image reconstruction using. (a) Three eigenimages. (b) Two eigenimages. (c) One eigenimage for different values of ε_1 and Q

3.2. The Eigenimage

Consider the image shown in Figure 2–(a). It is decomposed with parameters of $p = 0.5, \varepsilon_1 = 5$ and $Q = 5$ into 91 blocks (see Figure 2–(b)). Figures 2–(c) and 2–(d) show the corresponding EMap and RMap. Figure 3 shows the three pc_i channels corresponding to the image shown in figure 2–(a). In all eigenimages the dynamic range of the image is exaggerated to give a better visualization.



(a) (b)

Fig .7: Proposed block–wise interpolation, (a) sample problem and (b) the solution with $SNR > 22dB$.

The stochastic distributions of pc_i are investigated in Figure 3–d. It shows the histogram of the three produced planes for the image shown in Figure 2–(a). In this example, the standard deviations of the pc_i planes are computed as: $\sigma_{pc1} = 52, \sigma_{pc2} = 12$ and $\sigma_{pc3} = 6$. Note the perfect compaction of the energy in pc_1.

Figure 5–(a), 5–(b), and 5–(c) show the values of κ_1, κ_2 and κ_3 for the image shown in Figure 2–(a) for different values of ε_1 and Q. Note that rather than the trivial cases of $Q \le 2$ and $\varepsilon_1 > 9$ (which are never used actually), more than 90% of the image energy is stored in pc_1, while pc_2 and pc_3 hold about 9% and 1% of the energy, respectively. Having in mind that in the original image $\kappa_r = 38\%, \kappa_g = 32\%$ and $\kappa_b = 30\%$, the energy compaction of the eigenimages are considerable.

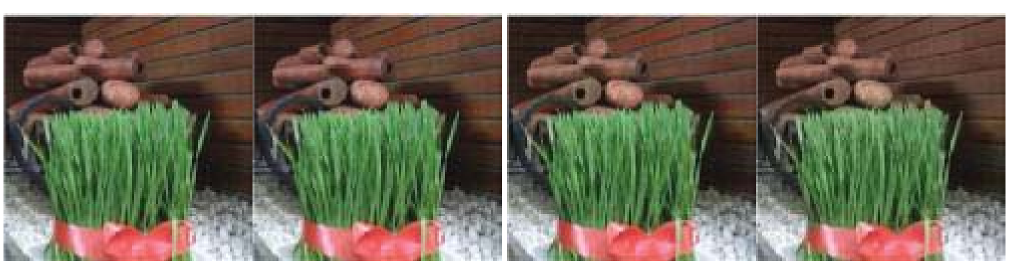
Figure 8 shows the results of reconstructing the image of Figure 8–(a) from the eigenimages. While Figure 8–(b) shows the result of reconstructing the image using all three eigenimages, Figures 8–(c) and 8–(d) show the results of ignoring pc_3 and both pc_3 and pc_2, respectively. The resulting *PSNR* values are $60dB$, $38dB$, and $31dB$, respectively. Note that $PSNR = 60dB$ (instead of infinity), for reconstructing the image using all eigenimages is caused by the numerical errors, while the two other *PSNR* values ($38dB$, $31dB$) show some loss of information. The *PSNR* values of above $38dB$ are visually satisfactory even for professionals [13].

Figure 6 shows the *PSNR* values obtained by reconstructing the image using all the three channels (Figure 6–(a)), only two channels (Figure 6–(b)), and just one channel (Figure 6–(c)), for different values of ε_1 and Q. Note that for values of $\varepsilon_1 \le 8$ and $Q \ge 3$, reconstructing the image using all eigenimages gives the high *PSNR* value of about $60dB$, while neglecting one and two eigenimages results in $PSNR \ge 35dB$ and $PSNR \ge 28dB$, respectively.

3.3. Color Image Compression

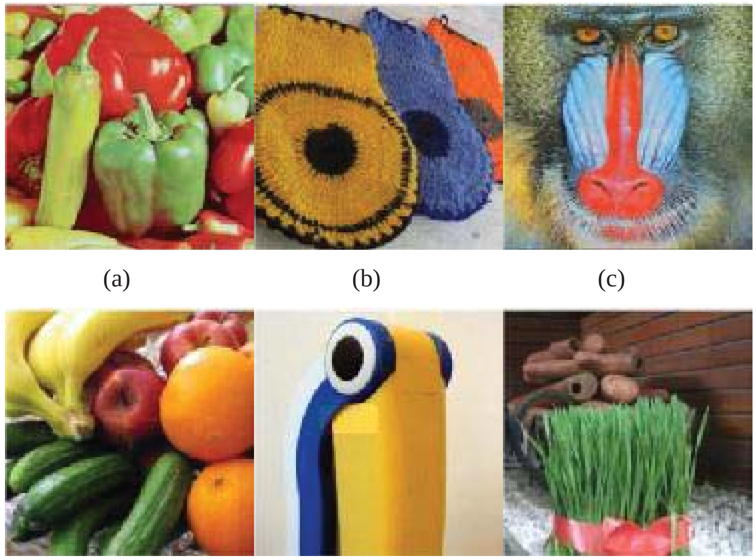
Figure 9 shows the results of the proposed compression method. Table 1 lists the compression ratio used for compressing the eigenimages and the resulting compression ratio and *PSNR* values. These results has been acquired while setting $p = \frac{1}{2}, \varepsilon_1 = 5$,

and $Q = 5$. Figure 10 shows the exaggerated difference between the reconstructed images shown in Figure 9 and the original images. Here, the scheme is defined as $x^* = (x - [\eta - \sigma]) / (2\sigma)$, where η and σ denote the expectation and the standard deviation of *x*, respectively. Note the high compression ratio of about 70 : 1 in all cases, while the *PSNR* is mostly above $25dB$. Among other region–based coding approaches the method by *Carveic et. al.* is one of the best [6]. They mixed the color and texture information into a single vector and performed the coding using a massively computationally expensive algorithm. The final results show *PSNR* values



(a) (b) (c) (d)

Fig . 8: Results of reconstructing an image from its eigenimages. (a) Original image adopted from [21]. (b) Using all eigenimages ($PSNR = 60dB$). (c) ignoring one eigenimage ($PSNR = 38dB$). (c) ignoring two eigenimages ($PSNR = 31dB$).

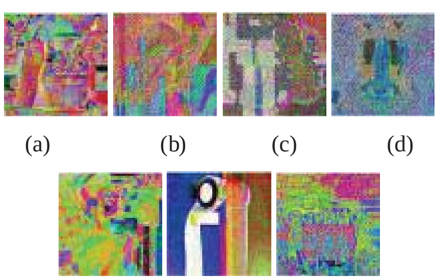


(a) (b) (c)

(d) (e) (f)

Fig . 9: Results of the proposed compression method on sample images. For details about compression ratio and PSNR see Table 1.

of about $20:1$ for compression ratios of about $40dB$. In [9], the researchers use the same separation scheme between compression in the two disjoint domains of spectral and spatial redundancy using a PCA neural network. They reached the compression ratio of $3.7{:}1$ with value of *PSNR* around $25dB$, while almost all test samples are homogeneous. In [24], the method gives the compression ratio of about $14.5{:}1$ but with the same range of *PSNR*as ours. The only drawback of the proposed compression method is some rectangular artifacts in ultra simple images, as seen in Figure 9–(f). Our future plan is to overcome this problem.



(a) (b) (c) (d)

(e) (f) (g)

Fig . 10: Exaggerated error of the proposed compression method.

Table 1 also compares the proposed algorithm with JPEG and JPEG2000. To do this comparison, each sample image is compressed using each one of these algorithms with the exact com pression ratio acquired from the proposed method. As expected, Table 1 shows that JPEG2000 is always giving a better result compared to JPEG. Comparing the proposed method with JPEG2000 we understand that except for the case of Figure 9–(f) the proposed method is giving a higher PSNR than both others. Followed by the discussion given in the last paragraph we omit Figure 9–(f) from the investigation. Numerically, the proposed method is 14% and 7% better than JPEG and JPEG2000, respectively. This means more than $3.6dB$ and $1.9dB$ increase in the quality when comparing the results of the proposed method with JPEG and JPEG2000, respectively.

4. Conclusion

The performance of the proposed eigenimage extraction method is analyzed both in terms of energy compaction and partial reconstruction. The method shows the ability to reduce the energy content of the third eigenimage to a few percents while the first one encompasses more than ninety percent of the total energy. Also, the subjective quality of the partially reconstructed images is investigated. Then the performance of the proposed color image compression method is analyzed. Comparison of the results with the available literature shows the superiority of the proposed color image compression method.

Acknowledgement

We wish to thank the respected referees for their constructive suggestions. The _rst author wishes to thank Ms. *Azadeh Yadollahi* for her encouragement and invaluable ideas.

Table 1: Numerical information relating to the samples shown in Figure 9. [n: block count. λ: Compression Ratio, bpp: bit per pixel. PSNR: Peak signal to noise ratio in dBs.]

Sample	Proposed Method										JPEG	JPEG2000
	n	pc_1		pc_2		pc_3						
		λ	bpp	λ	bpp	λ	bpp	λ	bpp	PSNR	PSNR	PSNR
9-(a)	334	42.6:1	0.19	58.8:1	0.14	∞	0	71.1:1	0.34	28.4	26.8	27.1
9-(b)	161	39.8:1	0.20	55.1:0	0.15	∞	0	68.0:1	0.35	30.4	27.7	29.5
9-(c)	311	44.4:1	0.18	58.0:1	0.14	∞	0	72.5:1	0.33	24.5	19.8	21.7
9-(d)	203	41.3:1	0.19	56.9:1	0.14	∞	0	70.1:1	0.34	33.3	30.5	32.4
9-(e)	32	45.5:1	0.18	62.8:1	0.13	∞	0	78.8:1	0.30	26.1	38.4	42.4
9-(f)	91	41.3:1	0.19	57.7:1	0.14	∞	0	71.4:1	0.34	35.4	30.2	32.7

References

[1] ITU–R Recommendation BT–601–5: Studio Encoding Parameters of Digital Television for Standard 4:3 and Widescreen 16:9 Aspect Ratios. Geneva: www.itu.ch/, 1994.

[2] A. Abadpour, "Color image processing using principal component analysis," Master's thesis, Sharif University of Technology, Mathematics Science Department, Tehran, Iran, June 2005.

[3] A. Abadpour and S. Kasaei, "A new parametric linear adaptive color space and its PCA–based implementation," in The 9th Annual CSI Computer Conference, CSICC, Tehran, Iran, 2004, pp. 125–132.

[4] ——, "Performance analysis of three likelihood measures for color image processing," in IPM Workshop on Computer Vision, Tehran, Iran, 2004.

[5] K. Benson, Television Engineering Handbook, J. C. Whitaker, Ed. New York, London: Mc Graw–Hill, 1992.

[6] D. Carevic and T. Caelli, "Region–based coding of color images using karhunen–loeve transform," Graphical Models and Image Processing, vol. 59(1), pp. 27–38, 1997.

[7] T. Chaira and A. Ray, "Fuzzy approach for color region extraction," Pattern Recognition, vol. 24, pp. 1943–1950, 2003.

[8] S.-C. Cheng and S.-C. Hsia, "Fast algorithm's for color image processing by principal component analysis," Journal of Visual Communication and Image Representation, vol. 14, pp. 184–203, 2003.

[9] C. Clausen and H. Wechsler, "Color image compression usTable 1: Numerical information relating to the samples shown in Figure 9. [n: block count. λ: Compression Ratio, bpp: bit per pixel. PSNR: Peak signal to noise ratio in dBs.] ing PCA and back propagation learning," Pattern Recognition, vol. 33, pp. 1555–1560, 2000.

[10] S. Fukuma, M. Iwahashi, and N. Kambayashi, "Lossless color coordinate transform for lossless color image coding," 1998, pp. 595–598.

[11] A. Hyvrinen and E. Oja, "Independent component analysis: Algorithms and applications," Neural Networks, vol. 13(4– 5), pp. 411–430, 2000.

[12] S. Kasaei, M. Dericher, and B. Biashash, "A novel _ngerprint image compression technique using wavelet packets and pyramid lattice vector quantization," IEEE Transactions on Image Processing, vol. 11(12), pp. 1365–1378, 2002.

[13] S. Katzenbeisser and A. Petitcolas, Information Hiding Techniques for Steganography and Digital Watermarking. Artech House Inc., 2000.

[14] G. J. Klinker, S. A. Shafer, and T. Kanade, "The measurement of highlights in color images," International Journal of Computer Vision, vol. 2, pp. 7–32, 1988.

[15] ——, "A physical approach to color image understanding," International Journal of Computer Vision, vol. 4, pp. 7–38, 1990.

[16] J. O. Limb, C. Rubinstein, and J. Thompson, "Digital coding of color video signals– a review," IEEE Transaction of Communication, vol. 25(11), pp. 1349–1385, 1977.

[17] D. O. Nikolaev and P. O. Nikolayev, "Linear color segmentation and its implementation," Computer Vision and Image Understanding, vol. 94, pp. 115–139, 2004.

[18] N. Papamarkos, C. Strouthopoulos, and I. Andreadis, "Multithresholding of color and gray–level images through a neural network technique," Image and Vision Computing, vol. 18, pp. 213–222, 2000.

[19] W. Pratt, "Spatial transform coding of color images," IEEE Transaction on Communication Technology, vol. 19(6), pp. 980–992, 1971.

[20] H. Samet, "Region representation: Quadtrees from boundary codes," Comm. ACM, vol. 21, p. 163:170, 1980.

[21] S. T. Seifi and A. Qanavati, "Digital color image archive," Qnavati@mehr.sharif.edu.

[22] J. Slater, Modern Television System to HDTV and Beyond. London: Pitman, 1991.

[23] M. Turk and A. Pentland, "Eigenfaces for recognition," Journal of Cognitive Neuroscience, vol. 3(1), pp. 71–86, 1991.

[24] Y.-D. Yu, D.-S. Kang, and D. Kim, "Color image compression based on vector quantization using PCA and LEBD," in IEEE IENCON'99, 1999, pp. 1259–1262.