Journal of ELECTRICAL ENGINEERING, VOL. 57, NO. 4, 2006, 226–234

COLOUR IMAGE COMPRESSION USING A MODIFIED ANGULAR VECTOR QUANTIZATION ALGORITHM

Hazem Al-Otum — Walid Shahab — Mamoon Smadi *

This paper introduces a new angular-based VQ scheme based on converting the 3D-Layered (angular) vectors into 2D-Layered (greyscale wise) vectors using the Mahalanobis Colour Distance (MCD). The proposed VQ scheme (VQ-MCD) exhibits a comparable output performance with the conventional 3D VQ schemes at very low bit rates while at higher bit rates VQ-MCD exhibits a better visual reconstruction quality than 3D VQ. The main advantage of using the proposed VQ scheme is the high reduction in the coding time that reaches 58–67 % depending on the implemented codebook size.

K e y w o r d s: image compression, colour spaces, colour coding, vector quantization

1 INTRODUCTION

Digital image compression continues to be a very active area of research as a result of the increasing demand in various fields. The main objective of image compression is to encode the image data into a compact form, minimizing both the number of bits in the representation, and the distortion caused by the compression. The importance of image compression is emphasized by the huge amount of data in raster images. For instance: a typical grey-scale image of 512×512 pixels, each represented by 8 bits, contains 256 Kbytes of data. With the colour information, the number is tripled. Thus, the necessity for compression is obvious and has found a wide spectrum of applications that range from texture and image segmentation [1, 2], computer vision [3, 4], to machine learning [5, 6] *etc.*

Colour Image compression has been approached in a number of ways and for various reasons. Reducing the number of data points is vital for computation when working with a large amount of data whether the goal is to compress data for transmission or storage purposes, or to apply a computationally intensive algorithm. Vector Quantization (VQ) has been recognized as an efficient method for colour image compression due to its superior compression ratio and decoder simplicity.

In VQ, a set of data vectors is represented by a smaller set of codevectors, thus requiring only the codevector to be stored or transmitted. Data points are associated with the nearest codevector generating a lossy compression of the data set. The challenge is to find the set of codevectors (the codebook) that describes data in the most efficient way. A wide range of VQ algorithms exist, and, the most well-known is the LBG algorithm [5].

Most prior works concentrate on VQ encoding of greyscale images [6-8]. Here, there is a lack on investigating the effects of implementing VQ based on 3-D vectors (RGB, YIQ/YUV, LAB *etc*) as a combined codebook which may lead to a considerable reduction in the bit rate at the expense of a slight decrease in the image quality. Here, two methods can be implemented with colour images:

1) Separate component coding where each colour primary is regarded as an independent monochrome image. Hence, encoding of a colour image is roughly equivalent to encoding three monochrome images independently. In prior works, it was considered that for most colour images the spatial bandwidth of IQ or UV components can be reduced to less than 1/3 that of the Y component [9]. Therefore, the UV, IQ were spatially sub-sampled without hurting the overall coding performance.
2) The second issue is to use combined (3D) codebooks and similar results may be achieved noting that the system complexity is reduced to 1/3 (theoretically). In fact, there is a lack of prior works in this direction.

Paliwal and Ramasubramanian [10] proposed the use of a variable scale factor which is a function of the iteration number to be implemented with the modified K-means algorithm, and, it was shown that it works faster than the modified K-means algorithm. However, nothing was said about the implementation of such an algorithm with colour images. Hui *et al*[11] introduced methods for reducing the table storage required for encoding and decoding with unstructured vector quantization (UVQ) or tree-structured vector quantization [7] (TSVQ). Specifically, a low-storage secondary quantizer is used to compress the codevectors (and test vectors) of the primary quantizer. The relative advantages of uniform and nonuniform secondary quantization are investigated. In comparison with conventional methods it is found that a significant storage reduction is possible (typically a factor of two to three compared with TSVQ) with little loss of signal-to-noise ratio (SNR). However, the algorithm is

* EE Department, Jordan University of Science & Technology, Irbid 22110, Jordan. E-Mail: hazem-ot@just.edu.jo

ISSN 1335-3632

complicated even though it is efficient, and, when compared with the non TSVQ techniques it requires higher storage. Finally, Xiaplin Wu [12] proposed a simple product VQ technique for frame buffers that exploits redundancies in both image and colour spaces. This VQ technique can reduce the frame buffer of a colour image by one half from the current algorithms with comparable image quality. The NTSC YIQ rather than RGB colour space was used. However, the technique is designed to be implemented with CRT colour palettes and in fact was not checked for coding applications.

This paper describes a modified image VQ scheme based on Mahalanobis Colour Distance (MCD) and denoted as VQ-MCD. This technique calculates a one dimensional value (colour distance value) for each 3D pixel (RGB, YIQ/YUV, LAB *etc*). The nearest neighbour search using Mean-Square-Error (MSE), based on MCD value, is expected to considerably reduce the computational complexity. The organization of this paper is as follows: in section 2, colour spaces are briefly discussed, the MCD measure is introduced and the possibility of its implementation in colour image coding is explained. In section 3, the proposed VQ-MCD scheme is presented. Experimental results are given in section 4 for various test images. Finally, concluding remarks are given in section 5.

2 COLOR SPACES AND MCD

A Colour space is a way of referring to a model that represents all the possible colours that can be produced by a particular output device [13], such as a monitor, colour printer, photographic film or printing press. In the following subsections, well known colour models are discussed.

2.2 Colour Models

The three most popular colour models used in computer images are RGB. In video systems YIQ/YUV, $YCbCr\&LAB$ are commonly used, while $CMYK$ is used in colour printing. Other colour spaces such as HSI/HSV are related to the hue, saturation and brightness, and reflect the human perception of colour [13]. Below, a brief discussion is given.

1) The RGB tricolor system is considered as an additive colour mixture. Any source colour ($\mathbf{F}$) can be matched by a linear combination of three colour primaries, *ie*, Red, Green and Blue, provided that none of those three can be matched by a combination of the other two. Here, $\mathbf{F}$ can be as:

$$\mathbf{F} = r\mathbf{R} + g\mathbf{G} + b\mathbf{B} \tag{1}$$

where r, g, and b are scalars indicating how much of each of the three primaries ($\mathbf{R}$, $\mathbf{G}$ and $\mathbf{B}$) are contained in $\mathbf{F}$. The normalized form can be as:

$$\mathbf{F} = R\mathbf{R} + G\mathbf{G} + B\mathbf{B} \tag{2}$$

Where

$$\begin{aligned} R &= r/(r+g+b)\,; \\ G &= g/(r+g+b)\,; \\ B &= b/(r+g+b)\,; \end{aligned} \tag{3}$$

2) The $CIELAB$ model (denoted as $L^*a^*b^*$) that was created in 1976 as a refined version of the CIE XYZ model. Here, L^* serves as a correlate of lightness; a^* is a value for which $-a^*$ is green, and $+a^*$ is red, b^* is a value for which $-b^*$ is blue, and $+b^*$ is yellow. The $L^*a^*b^*$ can be obtained from XYZ as [13]:

$$\begin{aligned} L^* &= 116 f\Big(\frac{Y}{Y_n}\Big) - 16 \\ a^* &= 500 f\Big(\frac{X}{X_n}\Big) - f\Big(\frac{Y}{Y_n}\Big) \\ b^* &= 200 f\Big(\frac{Y}{Y_n}\Big) - f\Big(\frac{Z}{Z_n}\Big) \end{aligned} \tag{4}$$

where

$$f(t) = \begin{cases} t^{1/3}, & \text{if } t > 0.008856 \\ 7.787t + 16/116\,, & \text{if } t \le 0.008856 \end{cases} \tag{5}$$

In the a^*b^* plane, the radial distance $\sqrt{(a^*)^2 + (b^*)^2}$ correlates the chroma while the angular position (a^*/b^*) serves as correlates of the hue.

3) The $HSI/HSV/HSL/HCI/HVC$ models that represents a wealth of similar colour spaces:

HSL: Hue Saturation Lightness HSI (intensity) HSV (value) HCI (chroma / colourfulness) HVC, TSD (hue saturation and darkness)

The design of the HSI model reflects the way humans see colour, and, it offers advantages for image processing as well. In HSI model, I stands for intensity or brightness. It is, for our purposes, just the average of R, G, and B grey level values (sometimes with different weights). The two parameters that contain the colour information are the hue (H) and saturation (S).

4) The YUV, YIQ, $YCbCr$, YCC models that are basically used in colour television transmission YIQ and YUV are analogue spaces for NTSC and PAL systems respectively while YcbCr is a digital standard. These colour spaces separate RGB into luminance and chrominance information and are useful in compression applications (both digital and analogue). The YIQ can be obtained from the RGB model as follows:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.299 & 0.578 & 0.144 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{6}$$

On the other hand, PAL analog video uses the YUV. Y is the same as in the NTSC system, while U and V can be calculated as:

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \begin{bmatrix} 0.299 & 0.578 & 0.114 \\ -0.147 & -0.289 & -0.436 \\ -0.615 & -0.514 & -0.101 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{7}$$

2.3. Colour Compression and MCD Measure

The extension of greyscale compression to colour images is not straightforward. The generalization of colour compression can be reduced to a key idea: How to define the similarity or closeness of two colour pixels in an image. The fact that the fundamental concept of colour compression do not apply to colour images makes it difficult to define "angular compression". A straightforward extension is to apply separately appropriate compression techniques to each colour component and, hence, the image components are processed independently which leads to the possibility of altering the spectral composition of the image, *eg*, the colour balance and object boundaries. This effect near spatial edges in an image leads to the so-called "edge jitter". Another way is to deal with the 3D angular pixels as being one component instead of being composed of 3 different components and then to apply a suitable compression technique.

An interesting colour measure is the Mahalinobis Colour Distance (MCD) that has been commonly used in pattern recognition analysis. It makes uniform the influence of the distribution of each attribute considering the correlation between each term. The MCD can be defined for the attributes R, G and B (RGB space) as [14]:

$$\Delta d = \sqrt{[\,\Delta R\;\;\Delta G\;\;\Delta B\,]\,\mathbf{M}^{-1}\,[\,\Delta R\;\;\Delta G\;\;\Delta B\,]^{\top}} \qquad (8)$$

The variance-covariance matrix $\mathbf{M}$ in can be calculated as:

$$\mathbf{M} = \begin{bmatrix} \sigma_{RR} & \sigma_{RG} & \sigma_{RB} \\ \sigma_{GR} & \sigma_{GG} & \sigma_{GB} \\ \sigma_{BR} & \sigma_{BG} & \sigma_{BB} \end{bmatrix} \qquad (9)$$

The elements of matrix $\mathbf{M}$ in (9) can be calculated as (eg, σ_{RG}):

$$\sigma_{RG} = \sigma_{GR} = \frac{1}{n-1}\sum_{i=0}^{n}(R_i - \overline{R})(G_i - \overline{G}) \qquad (10)$$

Where

$R_i, G_i, \,\&\, B_I$ – the values of the red, green, blue components of the current pixel respectively.

$\overline{R}, \overline{G} \,\&\, \overline{B}$ – the mean values of the R, G and B components in the given image respectively.

In fact, MCD has been implemented in various applications such as in [14]. In [15], MCD is implemented for colour morphology, which is directed to the field of image segmentation and nothing was said about the possibility for implementing MCD for coding applications. Now, a question rises: how can we implement the MCD measure for coding applications, and if it is possible, what about the reconstruction quality. Here, the concept of MCD has been adopted from [15] and modified to be implemented to VQ image compression instead of image segmentation. One possible application for this perceptual distance is to reduce the computational cost in searching for the nearest neighbour in 3D VQ coding process as well as in 3D codebook design, since we need only to compare the vector colour distance instead of the whole colour values within the vector. Here, it is expected that the output quality will be slightly reduced. It will be shown below that a considerable reduction in the encoding time can be achieved at the expense of a slight decrease in the reconstruction quality.

3 THE PROPOSED ANGULAR VQ–MCD SCHEME

3.1 Conventional 3D-VQ scheme

As described, VQ is a generalization of scalar quantization technique where the number of possible values (pixels) is reduced. Here, in conventional 3D-VQ, the input image is portioned into contiguous non-overlapping 3D blocks (typically $4 \times 4 \times 3$) and called *3D-vectors*. A representative 3D-block (*3D-codevector*) is selected from a pre-designed set (*3D-codebook*). Here, the 3D-VQ maps each input vector to a codevector, which is the representative vector of the partition. Typically the space is partitioned so that each vector is mapped to its nearest codevector minimizing certain *distortion function*. The distortion is commonly the Euclidean distance between the two vectors:

$$d(X, Y) = \sqrt{\sum_{i=1}^{M}(X_i - Y_i)^2} \qquad (11)$$

where X and Y are the 3D-vector and the selected 3D-codevector respectively.

The 3D-codevector is commonly chosen as the *centroid* of the vectors in the partition, that is

$$C = (c_1, c_2, \ldots, c_m) = (\overline{X}_1, \overline{X}_2, \ldots, \overline{X}_m) \qquad (12)$$

where $\overline{X}_i$ is the average value of the ith component of the vectors belonging to the partition. This selection minimizes the Euclidean distortion within the partition. The 3D-codebook consists of all the 3D-codevectors. Next, the index of the codevector is then sent to the decoder by $\lceil \log_2 K \rceil$ bits (as shown in Fig. 1), where K is the number of codevectors in the codebook. For example, in the case of $4 \times 4 \times 3$ pixel blocks and a codebook of size 256, the bit rate is $\log_2(256)/(4 \times 4 \times 3) = 0.167$ bpp, and the corresponding compression ratio is $(4 \times 4 \times 24)/8 = 48$.

The codebook is usually constructed on the basis of a *training set* of vectors. The training set consists of sample vectors from a set of images. Denote the number of vectors in the training set by N. The object of the codebook construction is to design a codebook of K vectors so that the average distortion is minimized in respect to the training set. For example, if a training set of 100 training vectors are given as shown in Fig. 2a, then one possible partitioning into 10 sets is shown in Fig. 2b.

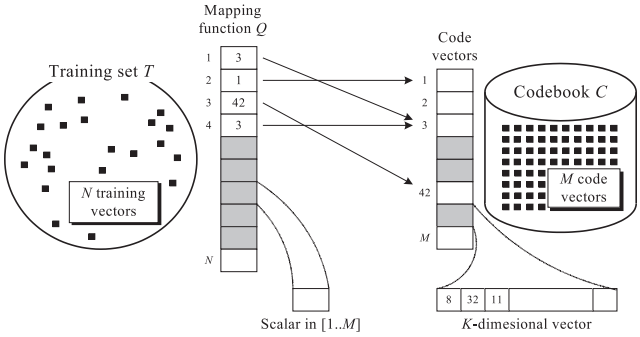


Fig. 1. Basic structure of VQ scheme.

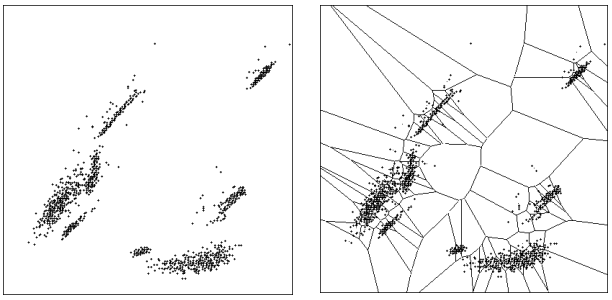


Fig. 2. a) An example of 100 sample training set consisting of vectors plotted into the vector space, and, b) one possible partitioning the vectors into 10 sample regions.

A crucial point is the computational cost of the codebook design as well as the VQ encoding. For codebook generation, even the time complexity is usually very high, it is not so critical, since the codebook is typically generated only once as a preprocessing stage. However, the search of the best match is applied K times for every block in the image. For example, in the case of 4×4 blocks and $K = 256$, $16 \cdot 256$ multiplications are needed for each block. For an image of 512×512 pixels there are 16384 blocks in total, thus the number of multiplications required is over 67 million.

3.2 Colour Image Using the VQ-MCD Scheme

In full search encoding process, the MSE is calculated for the whole colour vector to find the nearest neighbour vector in the codebook. The VQ-MCD algorithm searches for the nearest neighbour vector using the MCD matrix instead of using the colour vector itself, which reduces the computational cost to, theoretically 1/3. The VQ-MCD encoding/decoding is discussed in the following subsections.

3.2.1. VQ-MCD Encoding

Before starting with the encoding step, it is very important to note that it is assumed that the codebook is already built. Of course, this can be done using any preferable algorithm for codebook production. In fact, the VQ-MCD can be also used to produce the codebook, however, it was not produced based on the MCD concept due to the following assumptions:

- VQ-MCD is basically designed to reduce the computational cost during the encoding/decoding procedures (online action).
- The codebook is usually produced in advance (offline action).
- Despite of the fact that the computational cost is expected to be dramatically reduced when using VQ-MCD, however, the VQ-MCD codebook will lead to reduction in the output image quality.

The VQ-MCD takes place in the following steps:

Step1: Given the input RGB colour image with a size of $M \times N \times 3$ as:

$$\mathbf{RGB} \underline{\underline{\Delta}} \{RGB(i,j,q) : i = 1,2,\ldots,M,\ j = 1,2,\ldots,N,\ q = 1,2,3\} \quad (13)$$

Create the colour distance matrix for RGB by calculating $\Delta \boldsymbol{d}$ matrix for each pixel in the input image, using equ.8–10, as:

$$\Delta \boldsymbol{d} \underline{\underline{\Delta}} \{\Delta d(i,j) : i = 1,2,\ldots,M\,;\ j = i = 1,2,\ldots,N\}\,. \quad (14)$$

Here, the increment $\Delta d(i,j)$ requires a reference colour, for example, $\Delta R = R_i - R_0$ where R_i is the R current value and R_0 is the R-value of the reference colour. The reference can be defined locally or globally. To avoid incompatibility and to simplify our task, the reference colour was taken to be the pure black with $RGB_0 \equiv (0,0,0)$. In this case, each colour pixel $RGB(i,j)$ has a corresponding $\Delta d(i,j) = d(i,j)$.

Step 2: Create the input vectors by dividing the input image RGB and the corresponding MCD matrix $\Delta \boldsymbol{d}$ into contiguous and non-overlapping $m \times n \times 3$ and $m \times n$ blocks respectively. Thus:

$$\begin{aligned} &\{\mathbf{RGB}(k) : k = 1,2,\ldots,P\} \\ &\{\vec{d}(k) : \ = 1,2,\ldots,P\} \end{aligned} \quad (15)$$

where P – the number of blocks in the input image $P = \frac{M \times N}{m \times n}$.

Step 3: Create another MCD matrix $\mathbf{\Delta CB}$ (using equ. 8–10) for the codebook CB (produced in advance) as:

$$\begin{aligned} &\mathbf{CB} \underline{\underline{\Delta}} \{CB_{m \times n \times 3}(l) : l = 1,2,\ldots,L\} \\ &\mathbf{\Delta CB} \underline{\underline{\Delta}} \{\Delta CB_{m \times n}(l) : l = 1,\ldots,L\} \end{aligned} \quad (16)$$

Where

L – the codebook size.

$CB_{m \times n \times 3}(l)$ – the l-th 3D codevector in the codebook CB.

$\Delta CB_{m \times n}(l)$ – the l-th 2D MCD matrix for the l-th 3D codevector in $\mathbf{CB}$.

Step 4: For each input vector $\mathbf{RGB}(k) = \{RGB_{m\times n\times 3}(k)\colon k = i = 1,\ldots,P\}$ in the input image, compare its MCD $\boldsymbol{d}(k) = \{d_{m\times n}(k)\colon k = i = 1,\ldots,P\}$ with MCDs of the codebook vectors $\mathbf{\Delta CB} \underline{\underline{\Delta}} \{\Delta CB_{m\times n}(l)\colon l = 1,\ldots,L\}$ to find the nearest neighbour using the MSE value:

$$MSM_{m\times n}(k,l) = \frac{1}{m\times n}\sum_{l=1}^{L}\left[d_{m\times n}(k) - \Delta CB_{m\times n}(l)\right]^2 \tag{17}$$

Note that instead of comparing the 3D $RGB_{m\times n\times 3}(k)$ with $CB_{m\times n\times 3}(l)$, we used to compare the 2D $d_{m\times n}(k)$ with $\Delta CB_{m\times n}(l)$.

Step 5: Next we find the index of the codevector that exhibits the minimum MSE value, and extract it, if required. Thus, the current output vector has an index:

$$Indx(k) = \{l \to \log_2(L)\colon MSE_{m\times n}(k,l) = \min\} \tag{18}$$

Here, the index of the selected codevector is transmitted to the channel. Then, the next vector is applied to step 4 and the process is repeated until all vectors are passed for comparison.

3.2.2 VQ-MCD Decoding

At the decoder, assuming ideal channel, the received indices $\{Indx(k),\ k = 1,2,\ldots,P\}$ are applied to the VQ decoder which is normally a lookup table and the codevector $CB_{m\times n\times 3}(l)$ with an index $l = Indx(k)$ is extracted from the codebook as:

$$Out_{m\times n\times 3}(k) = \{CB_{m\times n\times 3}(Ind(k)) = CB_{m\times n\times 3}(l)\,,\ \forall\, k = 1,\ldots,P\ \&\ l = 1,\ldots,L\} \tag{19}$$

3.3 The Computational Cost in VQ-MCD

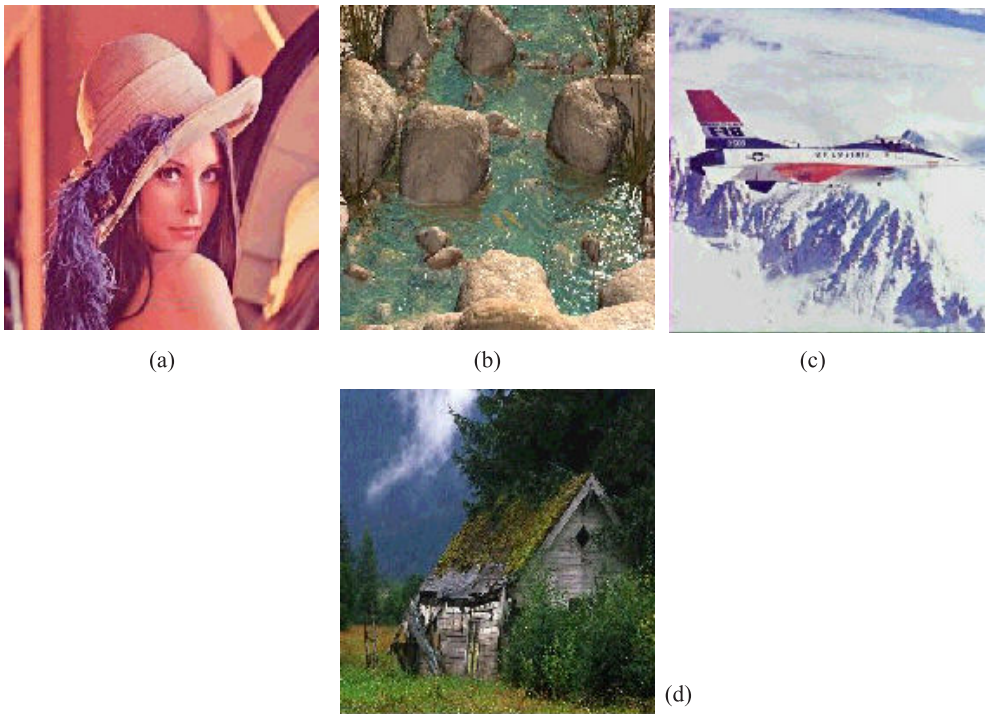
We, now, turn to calculate the computational cost for both conventional 3D-VQ and VQ-MCD. In fact, it is very important to reduce the number of calculations at the encoding as well as the decoding steps. This is attributed to the fact that the encoding/decoding procedures are required to be performed in real-time (on line actions). In conventional 3D-VQ, if the vector dimension is $s = m\times n\times 3$, codebook dimension is L and number of vector

Table 1. Computational Cost for VQ-MCD Encoding

3D-VQ	VQ-MCD			
	Stage 1: MCD Calculation			
	Calc. of $\overline{R},\overline{G}\,\&\,\overline{B}$	Calc. of M^{-1}	Calc. of Δd	Calc. of MCD
	$3(MN)$	$9(MN\cdot 3)$	$12(MN)$	$42(MN)$
—	Stage 2: VQ-MCD Codebook			
	$3L(mn)$	$9L(mn\cdot 3)$	$12L(mn)$	$42L(mn)$
	VQ Encoding Step			
$3((mn)+1)LP$	$((mn)+1)LP$			

Table 2. Demonstrative Examples on the Computational Cost of VQ-MCD

	$M=512,\ N=512, L=128,\ m=4,\ n=4,\ P=16384$		$M=512,\ N=512,\ L=256,\ m=4,\ n=4,\ P=16384$	
	Input Image	CodeBook	Input Image	CodeBook
Calc. of $\overline{R},\ \overline{G}\ \&\ \overline{B}$	786432 (0.74 %)	6144 (0.00574 %)	786432 (0.37 %)	12288 (0.0057 %)
Calc. of $\mathbf{M}^{-1}$	7077888 (6.62 %)	55296 (0.0517 %)	7077888 (3.31 %)	110592 (0.0517 %)
Calc. of Δd	3145728 (2.94 %)	24576 (0.023 %)	3145728 (1.47 %)	49152 (0.023 %)
Calc. of MCD	11010048 (10.29 %)	86016 (0.080 %)	11010048 (5.15 %)	172032 (0.0804 %)
VQ Encoding	35651584 (32.96 %)		71303168 (33.33 %)	
Total VQ+MCD	46747648 (43.71 %)		82485248 (38.56 %)	
3D_VQ	106954752 (100 %)		213909504 (100 %)	



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

Fig. 3. Examples of the implemented test images: a) "Lena", b) "Water", c) "Airplane" and d) "House" with 512×512 pixels and 24 bit/pixel resolution.

in the image is P, then the number of multiplication-addition operations can be as:

$$C_n = sLP \tag{20}$$

Now, if every codevector requires a number of bits $B = Rs = \log_2(L)$, then:

$$Cn = sP2^{Rs} \tag{21}$$

Thus, the computational cost increases exponentially with the codevector size as well as with the assigned bit rate R. In the case of implementing the VQ-MCD scheme, C_n becomes:

$$C_n = (s/3)LP = (s/3)P2^{R(s/3)} \tag{22}$$

This can be also written as:

$$C_n = mnLP = mnP2^{Rmn} \tag{23}$$

In order to include all steps in VQ-MCD, the calculation of the MCD value for each pixel in the image can be approximated as shown in Table 1. For better illustration, Table 2, depicts selected demonstrative examples of the Normalized Computational Cost (NCC) which is given in brackets along with the actual number of addition-multiplication operations. Here, NCC is computed by dividing the corresponding time required for corresponding encoding process on the maximum computational execution time which is typical for 3D-VQ.

As well seen, the MCD calculation takes 5–10 % of the whole encoding process (depending on the VQ parameters: $L, P, m, \& n$.). Also, the VQ-MCD scheme provides a high reduction in NCC that reaches 58–67 % which proves the advantage of using the proposed VQ-MCD scheme over the conventional 3D-VQ scheme.

4 SIMULATION AND EXPERIMENTAL RESULTS

Simulations have been performed on a set of test images to evaluate the performance of the proposed VQ-MCD encoding algorithm in comparison with the full search VQ coding in terms of Peak Signal-to-Noise Ratio (PSNR), Normalized Computational Cost (NCC), and visual quality. In order to keep the work simple and to take the advantage of using the conventional 3D VQ (which is usually the reference for all works), the codebook is generated by the well-known LBG algorithm for different codebook sizes $L = 1024$, 512, 256, 128, 64, and 32, and the vector dimension was chosen to be $m \times n = 4 \times 4 = 16$ colour pixels. The training set consists of a number RGB test images such that shown in Fig. 3, all of 512×512 colour pixels with 24-bits resolution.

To evaluate the output image quality the PSNR has been implemented. Here, since the images are $RGBs$, the mean value of the PSNR for of the R, G and B layers

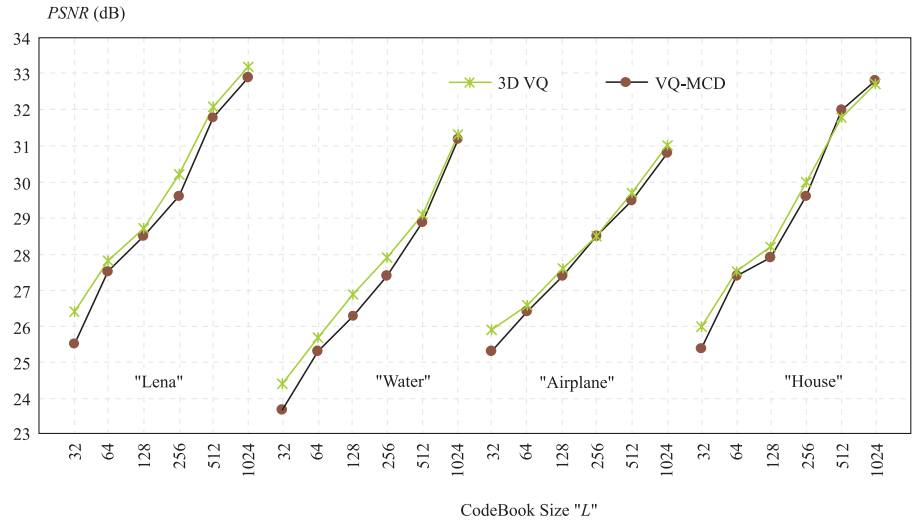


Fig. 4. A plot of the PSNR vs. L in case of implementing conventional 3D VQ and VQ-MCD with the test images "Lena", "Water", "Airplane" and "House".

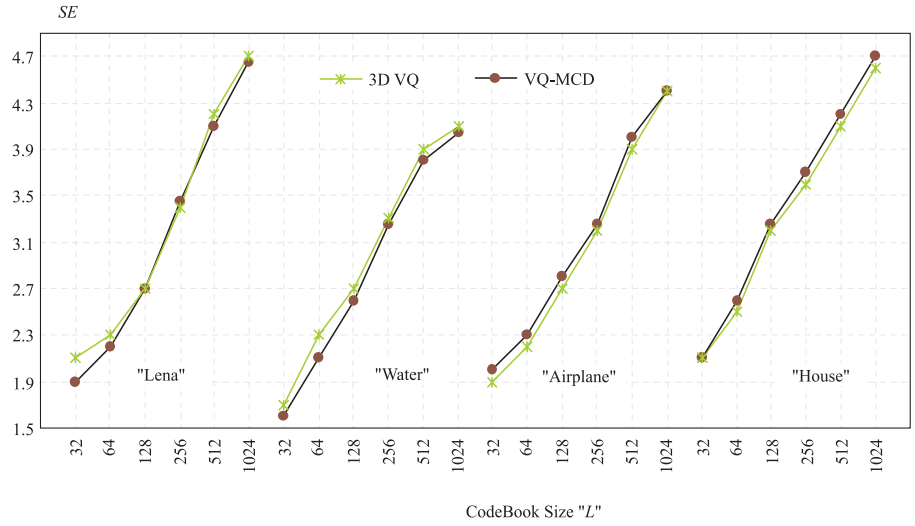


Fig. 5. A plot of $\overline{SE}$ vs. L in case of implementing conventional 3D VQ and VQ-MCD with the test images "Lena", "Water", "Airplane" and "House".

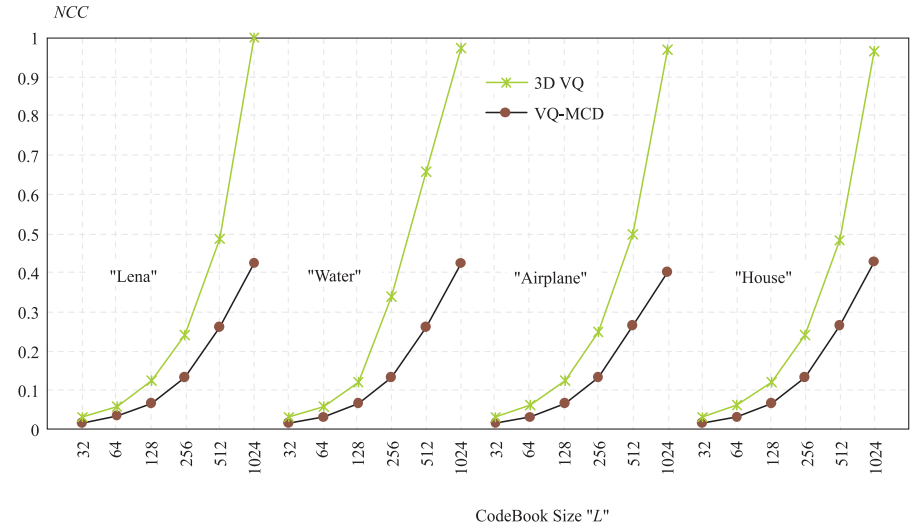


Fig. 6. A plot of NCC vs. L in case of implementing conventional 3D VQ and VQ-MCD with the test images "Lena ", "Water", "Airplane" and "House"

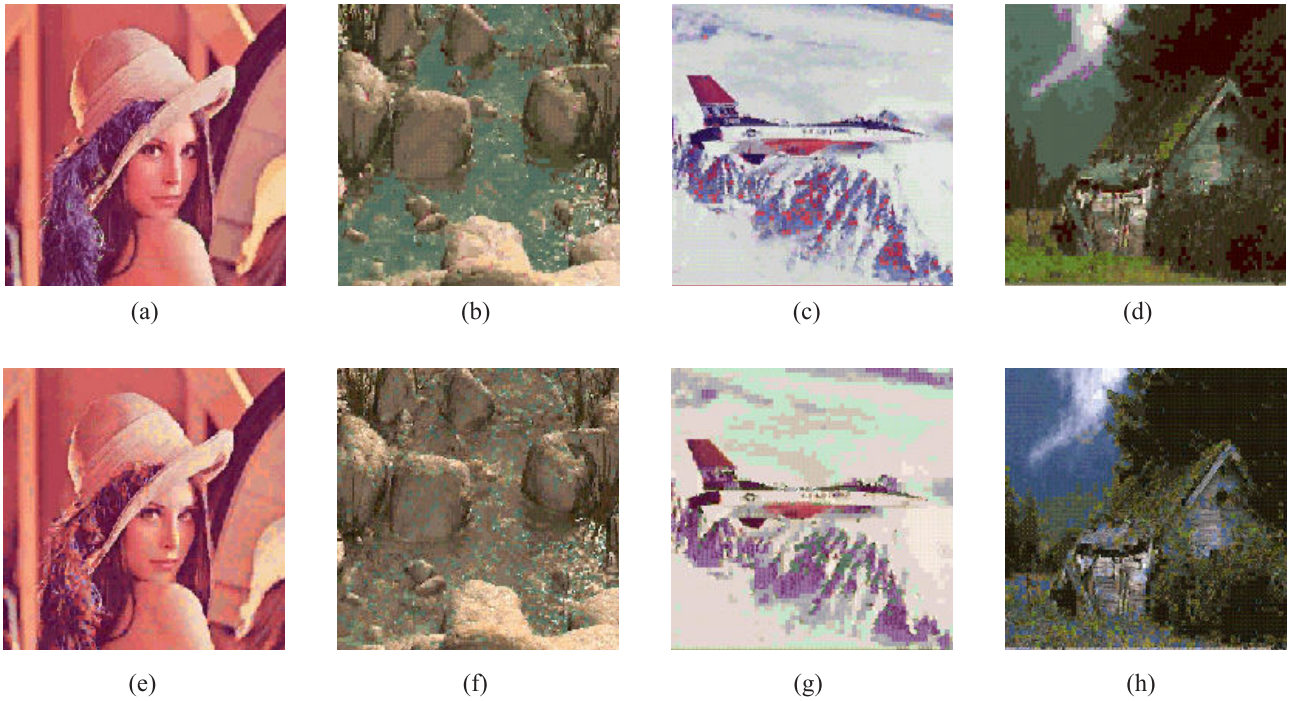


Fig. 7. Examples of implementing conventional 3D VQ (a,b,c and d) and the VQ-MCD (e,f,g and h) algorithms at a bit rate of 0.5 bpp.

is implemented:

$$PSNR = \frac{1}{3}\Big(20\log_{10}\frac{A^2}{MSE(R)} + 20\log_{10}\frac{A^2}{MSE(G)} + 20\log_{10}\frac{A^2}{MSE(B)}\Big) \quad (24)$$

Where $MSE(R), MSE(G)\&MSE(B)$ — the mean-squared error in the reconstructed R, B and G components of the image respectively.
A – The maximum available level of each layer (in our case $A = 255$). However, since PSNR is not an especially accurate measure of colour visual quality, a method for perceptual comparison is included and based on the one suggested in [16]: the original and reconstructed images were subjectively evaluated by observers with a background in image compression ($N = 25$). Here, we use the scales: 5-excellent quality, 4-very good, 3-good, 2-acceptable, 1-unacceptable. The display of the images is repeated 10 times ($M = 10$) per investigated image. The mean value for the subjective evaluation of the i-th reconstructed image is:

$$\overline{SE}(i) = \frac{1}{mn}\sum_{m}^{M}\sum_{n}^{N} SE_{nm}(i)\,. \quad (25)$$

Where:
$SE_{mn}(i)$ – The quality step given by the n-th observer for the i-th reconstructed image at the m-th displaying time.

Figure 4 depicts the PSNR values (versus the implemented codebook size) when applied to the test images for the conventional and the VQ-MCD schemes while Fig. 5 demonstrates $\overline{SE}$ vs. L.

To show the advantage of implementing VQ-MCD over the conventional VQ, Fig. 6 depicts a comparison of the NCC for both algorithms at different bit rates.

The analysis of the given data shows that:

- VQ-MCD exhibits a comparable output performance ($PSNR$ and $\overline{SE}$. In fact, the higher the codebook size, the closer $PSNR$ and $\overline{SE}$ to that obtained with conventional 3D VQ. Here, PSNR reduction is in the range 0.65–0.12 dB.
- At high bit rates VQ-MCD exhibits higher $\overline{SE}$ values than 3D VQ, something like 0.05–0.1, even there is a small reduction in $PSNR$.
- The proposed VQ-MCD encoder exhibits a great reduction in the coding time of about 58–67 % depending on the implemented codebook size.

For better illustration and visual comparison, Fig. 7 shows some examples of the reconstructed images at different bit rates.

5 CONCLUSION

In this paper, a new encoding technique is proposed and based on converting the 3D-Layered vectors into 2D-Layered vectors using the Mahalanobis Colour Distance (MCD). The proposed VQ-MCD exhibits a comparable output performance ($PSNR$ and $\overline{SE}$). In fact, the higher the codebook size, the closer $PSNR$ and $\overline{SE}$ to that obtained with conventional 3D VQ. Here, $PSNR$ reduction

is in the range 0.65–0.12 dB. At high bit rates VQ-MCD exhibits higher $\overline{SE}$ values than 3D VQ (0.05–0.1), even there is a small reduction in $PSNR$. Finally, the proposed VQ-MCD encoder exhibits a great reduction in the coding time of about 58–67 % depending on the implemented codebook size.

References

[1] SYMES, P. D.: Video Compression, McGraw Hill, New York, 1998.

[2] FRIGUI, H.—KRISHNAPURAM, R.: A Robust Competitive Clustering Algorithm with Applications in Computer Vision, IEEE Transactions on Pattern Analysis and Machine Intelligence **21** (May 1999), 450–465.

[3] CARPINETO, C.—ROMANO, G.: A Lattice Conceptual Clustering System and its Application to Browsing Retrieval, Machine Learning **24** No. 2 (1996), 95–122.

[4] PANOS NASIOPOULUS—RABAB K. WARD: A High-Quality Fixed Length Compression Scheme for Colour Images, IEEE Trans. Commun. **43** No. 11 (Nov. 1995), 2672–2677.

[5] LINDE, Y.—BUZO, A.—GRAY, R.: An Algorithm for Vector Quantizer Design, IEEE Mag. on Commun. **28** No. 1 (1980), 84–95.

[6] TAI, S. C.—LAI, C. C.—LIN, Y. C.: Tow Fast Nearest Neighbor Searching Algorithms for Image Vector Quantization, IEEE Trans. on Commun. **44** No. 12 (Dec. 1996), 1623–1628.

[7] SITARAM, V. S.—HUANG, C-M—ISRAELEN, P. D.: Efficient Codebooks for Vector Quantization Image Compression with an Adaptive Tree Search Algorithm, IEEE Trans. on Commun. **42** No. 11 (Nov. 1994), 2177–2182.

[8] MOHAMED, S.—FAHMY, M.: Image Compression Using VQ-BTC, IEEE Trans. on Commun. **43** No. 7 (July 1995), 2177–2182.

[9] HSUEH-MING HANG—HASKELL, B. G.: Interpolative Vector Quantization of Colour Images, IEEE Trans. on Commun. **36** No. 4 (April 1988), 465–470.

[10] PALIWAL, K.—RAMASUBRAMANIAN, V.: Comments on Modified K-Means Algorithm for Vector Quantizer Design, IEEE Transactions on Iimage Processing **9** No. 11 (Nov 2000), 1964–1967.

[11] HUI, D.—LYONS, D.—NEUHOFF, D.: Reduced Storage VQ via Secondary Quantization, IEEE Transactions on Image Processing **7** No. 11 (Apr. 1998), 477–495.

[12] XIAPLIN WU: YIQ Vector Quantization in a New Colour Palette Architecture, IEEE Transactions on Image Processing **5** No. 2 (Feb 1996), 321–329.

[13] YANG, C. C.—KWOK, S. H.: Efficient Gamut Clipping for Colour Image Processing using LHS and YIQ, Opt. Eng. J. **42** No. 3 (2003), 701–7111.

[14] IMAI, F.—TSUMURA, N.—MIYAKE, Y.: Perceptual Colour Difference Metric for Complex Images Based on Mahalanobis Distance, J. Elec. Imaging **8** No. 2 (2001), 385–393.

[15] Al-OTUM, H. M.: Morphological Operators for Colour Image Processing based on Mahalanobis Distance Measure, Opt. Eng. J. **42** No. 9 (Sep. 2003).

[16] Al-OTUM, H. A.: Qualitative and Quantitative Image Quality Assessment of Vector Quantization, JPEG, and JPEG2000 Compressed Images, J. Electronic Imaging, **12** No. 3 (July 2003), 511-521.

Received 10 april 2005

Hazem Al-Otum, PhD, Associate Prof IEEE Member, received his BASc and MASc in radio engineering from the Georgian Technical University (GTU) Republic of Georgia in 1996. He received his PhD degree in Digital Communications and Image Processing in 1996 also from GTU / Republic of Georgia in 1996. From February 1997 to February 2000 he was an assistant professor in the EE Department at the private Philadelphia University, Jordan. Since February 2000 he was with the Electrical Engineering Department of the Jordan University of Science and Technology (JUST). From February 2006, He was an Associate Professor with EE Department at JUST. He is member of the Jordanian Association of Engineers (JAE) and IEEE. His current interests include image coding, color morphology, video tracking, digital watermarking and pattern recognition.

Walid Shahab, Associate Professor of Electrical and Electronic Engineering. BSc, (1977), MSc, (1978) and PhD, (1981) from Universite des Sciences et Technique du Languedoc, Montpellier, France. Chairman of Biomedical Engineering Department, Ex-chairman of Electrical Engineering Department and Ex-associate dean of Engineering Faculty at Jordan University of Science and Technology. Research Interests: Surface Acoustic Waves, Digital and Analog Electronics an digital signal Processing.

Ma'moun Abdullah Ali Al-Smadii, born in Ajloun, Jordan in 1974, received the BS degree in Electrical Engineering/computer and MSc Degree in electrical engineering/control and power both from Jordan university of science and technology, Irbid, Jordan in 1998, and 2003, respectively. Since 1999, he was a teacher assistant in the department of electrical and electronics engineering, Al-huson university college, Balqa' Applied University. His research interests are in the areas of digital image processing, computer vision, microprocessor architecture and design, control, robotics, and machine learning. The master thesis subject was "Dynamic tracking of moving objects based on digital image segmentation and motion compensation techniques". His course work includes Bachelor courses in Digital Logic Design, Microprocessor Systems, and Computer Architecture. Diploma courses in microprocessor Basics and applications.