Image Compression using Different Vector Quantization Algorithms and Its Comparison

Gayatri Mohanta, Harish Chandra Mohanta

Abstract*: Image compression techniques are presented in this paper which can be used for storage and transmission of digital lossy images. It is mostly important in both multimedia and medical field to store huge database and data transfer. Medical images are used for diagnosis purposes. So, vector quantization is a novel method for lossy image compression that includes codebook design, encoding and decoding stages. Here, we have applied different lossy compression techniques like VQ-LBG (Vector quantization- Linde, Buzo and Gray algorithm), DWT-MSVQ (Discrete wavelet transform-Multistage Vector quantization), FCM (Fuzzy c-means clustering) and GIFP-FCM (Generalized improved fuzzy partitions-FCM) methods on different medical images to measure the qualities of compression. GIFP-FCM is an extension of classical FCM and IFP-FCM (Improved fuzzy partitions FCM) algorithm with a purpose to reward hard membership degree. The presentation is assessed based on the effectiveness of grouping output. In this method, a new objective function is reformulated and minimized so that there is a smooth transition from fuzzy to crisp mode. It is fast, easy to implement and has rapid convergence. Thus, the obtained results show that GIFP-FCM algorithm gives better PSNR performance, high CR (compression ratio), less MSE (Mean square error) and less distortion as compared to other above used methods indicating better image compression.

***Index Terms*: Vector quantization, Codebook design, DWT, MSVQ, Fuzzy clustering, GIFP-FCM.**

I. INTRODUCTION

Image compression techniques are needed to store large digital images for transmission which requires high bit-rates in low bandwidth channel. digital computer requires large storage to store image, audio and video files of different format. Similarly, computer network requires large bandwidth to transmit the data over the network. So, this data compression is important in different fields such as multimedia, medical, defense etc. Here we are focusing on compression of medical images. Image compression is required go get optimum diagnostic performance in medical images along with high diagnostic quality to give good Peak signal to noise ratio (PSNR) as well as good compression ratio (CR) of the reconstructed medical images.

Compression refers to reducing the redundancies of data representing image, audio or video without reducing its quality. Compress means to reduce the byte size of the original data. There are different lossless and lossy compression techniques for image compression. Lossy image compression is generally performed by utilizing vector quantization (VQ). The different lossy compression techniques used are Vector quantization (VQ_LBG), DWT-VQ (Discrete wavelet transform VQ), DWT-MSVQ (DWT Multistage Vector quantization), Fuzzy c-means (FCM), Generalized improved fuzzy partitions FCM (GIFP-FCM) methods. Vector quantization has been proven to be one of the most efficient procedures to generate appropriate codebooks. Vector quantization breaks down the image into number of rectangular squares. Each of these squares form a vector, called as training data vector. All the training vectors are assigned into number of clusters to produce a codebook. Similarly, image is recreated by replacing each training vector by its nearest codebook vector. The fuzzy logic methods such as FCM and GIFP-FCM techniques used in image compression are based on fuzzy clustering process. The proposed GIFP-FCM algorithm combines the benefits of VQ_LBG, DWT_VQ, and DWT_MSVQ & FCM to give good PSNR value, high compression ratio and maintaining the good quality of the reconstructed image after compression.

Image compression scheme for different 256 gray level bitmap benchmark and medical images with 512x512 pixels have been studied using the different lossy VQ and also MSVQ based on 3-decomposition levels of 2D-DWT.We have used medical images for image compression in our project. Reconstructed image qualities have been assessed dispassionately and abstractly. Target assessment has been used to assess the image in terms of PSNR. The different block sizes 2x2, 4x4 and 8x8 of the image have been taken with different codebook sizes like 16, 32, 64…256 [1].

In this paper, image compression is done by 2D-DWT, 2 stages of VQ followed by working out of RBF neural network. Compression effectiveness is analyzed based on bit rate and CR whereas contortion measure is dependent based on MSE and PSNR values. The RBF neural network training has increased the quality of compression image [4].

In this paper extended hybrid system, Hybrid DWT-VQ technique is applied on the image. The result shows the quality of reconstructed image depends on the size of the codebook. If CS (Codebook size) is increased, the CR decreases because the no. of bits that represent the block is increased and MSE decreases. Also, we can decrease MSE by increasing the correlation of pixels of image by computing differential matrix. But this will decrease a little CR. In extended hybrid system, differential matrix is computed before applying VQ to images. It is identical to original image and there is no any artifact in compressed image [3].

Revised Manuscript Received on July 05, 2019.

Gayatri Mohanta, ECE, College of Engineering and Technology, Bhubaneswar, Odisha, India

Harish Chandra Mohanta, ECE, Centurion University of Technology and Management, Bhubaneswar, Odisha, India

The PSNR can be improved by using the system error compensation (SEC) method. The SEC design is based on quantization and curvelet transform (CT) that decomposes the system error (E) to six scales. The results have improved 40.48%, 9.69% both in terms of bit rate and PSNR when compared to conventional method [6].

Here, VQ based image compression uses an improved partition based fuzzy clustering algorithm that is a modification of the classical fuzzy C-means method. It prizes fresh membership degrees. Here transition from fuzzy to crisp mode is done and GIFP-FCM method shows rapid convergence of the codebook design, easy to implement and faster [2]. Quality of the reconstructed image has been used using the average distortion measure and PSNR by varying the parameters α, m and codebook vector size in the number of iterations taken to converge [5].

An improved fuzzy partition (IFP-FCM) and generalized algorithm called GIFP-FCM for more effective clustering process have been introduced here. The robustness and convergence of both the algorithms are analyzed. GIFP-FCM outperforms FCM and IFP-FCM in clustering results and robustness. Also, experimental results are done on noisy image textures for segmentation [8].

VQ is one of the commonly used method for image compression. A novel fuzzy learning VQ (FLVQ) for image compression is used here. A modified objective function of the FCM algorithm is reformulated and is minimized by means of gradient method so that there would be a smooth transition from fuzzy mode (where each training vector is assigned to more than one codebook vectors) to crisp mode (where each training vector is assigned to only codebook vector) [9].

Neuro- fuzzy model technique combines the advantages of fuzzy VQ with neural network and wavelet and transform. Here the emphasis is on the usefulness of FVQ (fuzzy vector quantization) when combined with other conventional image coding techniques [7].

II. LOSSY COMPRESSION TECHNIQUES FOR IMAGE COMPRESSION

Vector quantization: It is the mapping of an input value x into an infinite or indefinite number of output values.

Advantages of VQ over SQ:

- Lower distortion is provided by vector quantization.
- Generation of codebook.

A.2.1 1D and 2D ANALYSIS OF VECTOR QUANTIZATION

A VQ is an approximator. The idea is rounding-off (say to the nearest integer). One-dimensional (1D) and 2-D vector quantization (VQ) is shown in fig. 2.1. Here every real number less than 2, 4, 6 are approximated by number 1, 3, and 5 respectively. The approximate values are uniquely represented by 2 bits. This is a 1-dimensional, 2-bit VQ. It has a rate of 2 bits/dimension.

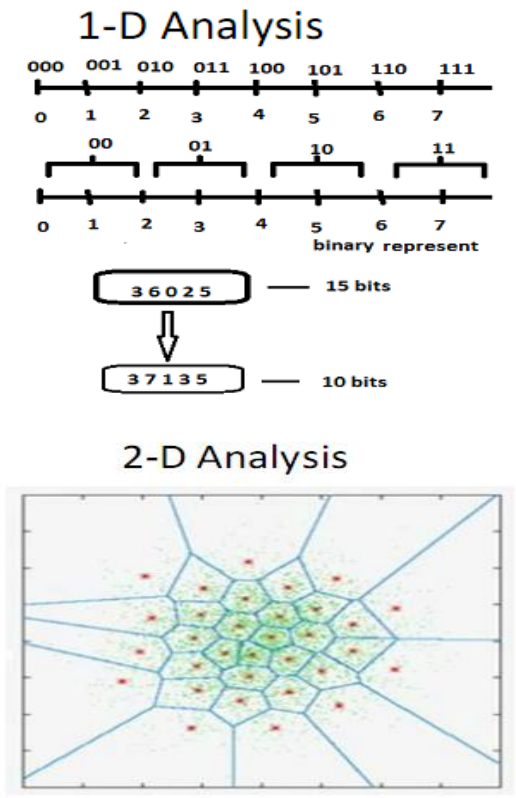

Fig. 2.1 1D and 2D analysis of VQ

In the above two examples, the red stars are called code vectors and the regions defined by the blue borders are called encoding regions. The set of all code vectors is called the codebook and the set of all encoding regions is called the partition of the space.

A.2.1.1 PRINCIPLE OF VECTOR QUANTIZATION

a. Vector quantization first partitions the input space X into K non-overlapping regions. A code vector is assigned to each cluster. The code vector is commonly chosen as the centroid of the vectors of the partition space.

b. Ki= (X_1, X_2…, Xm), where m represents number of training vectors in partition space Ki. e. g Size of training vector x is 2*2 or 4*4

1. A mapping process is carried out between the input vectors (Xi) and the centroid vector Yi. This introduces an error called distortion measure

$$d(X,Y) = \sqrt{\sum_{i=1}^{M}(X_i - Y_i)^2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.1)$$

where X and Y are two M-dimensional vectors.

a. The codebook of vector quantization consists of all the code words. The image is then divided into fixed size blocks and replaced with the best match found in the codebook based on the minimum distortion.

The basis for codebook construction is training vector space, block size and codebook size. The codebook is constructed with the constraint that the average distortion is minimized with respect to training space.

A.2.1.2 LBG ALGORITHM (GENERALIZED LYOLD ALGORITHM)

ALGORITHM CODEBOOK_DESIGN_LBG

Input: Take an initial codebook generated randomly as input.

Input image size=512*512, Block size= Training Vector size=2*2

Codebook size=16=No. of clusters or cells formed

No. of training vectors formed= (512*512)/(2*2)=256*256=65536

Initial codebook size=16*16, code vector=1*4=2*2

Output: Find the optimal codebook

Step1. Prepare the training vector space X of size= 65536*4 i.e. divide input image size (512*512) / 2*2 (Block size)

Step2. Apply the nearest neighbor condition by mapping each training vector Xi to its nearest code vector Yi using the Euclidean distance to obtain the cluster database.

Here 16 such clusters are formed.

Step3.The centroid of each partition region is calculated and updated with the old centroids.

Step4.Thus at every iteration, the codebook become progressively better. This process is continued till there is no change in the new codebook compared to previous codebook.

A.2.1.2.1 INITIAL CODEBOOK GENERATION

1. Randomly generate 16 indices in the range 1 to 65536.

53394 59362 8323 59860 41443 6393 18252
35841 62752 63235 10330 63609 62729 31810
52448 9299

2. Initial codebook is formed from the training vector space on the generated indices.

A.2.1.3 DECODING PROCESS

Perform VQ encoding on the input image using the produced new codebook and look-up/ index-table. The look-up table contains the indices of the codebook. At that point we store and transmit the codebook and the record table as a compacted document of the input image. VQ interpreting is performed utilizing the list table and the codebook to reproduce the approximate image of the original image.

ALGORITHM DECODING_PROCESS

INPUT: Codebook (size=16*4), Look-up table (size=256*256)

OUTPUT: Reconstructed image (size=512*512)

1. Initialize a reconstruction image of size 512*512
2. Code vectors (size 2*2) of look-up table (size 256*256) indices are used to form the reconstructed image of size 512*512.

EXAMPLE A.2.1 ENCODING AND DECODING PROCESS

Image of 6x6 size taken

241	192	212	76	10	220
21	156	123	36	108	233
165	108	109	52	112	45
155	41	19	247	67	78
117	57	98	126	234	43
156	52	67	23	107	65

*TABLE 2.1 CODEBOOK DESIGN FOR AN IMAGE OF SIZE 6*6*

Training vectors from image of size 2x2

241	192	212	76	10	220
21	156	123	36	108	233
165	108	109	52	112	45
155	41	19	247	67	78
117	57	98	126	234	43
156	52	67	23	107	65

Vectors to train a codebook

	x 1	**x 2**	**x 3**	**x 4**
X 1	241	192	2 1	156
X 2	212	7 6	123	3 6
X 3	1 0	220	108	233
X 4	165	108	155	4 1
X 5	109	5 2	1 9	247

Random codebook taken CB0 for N=3

	c1	**c2**	**c3**	**c4**
C1	32	177	143	210
C2	196	16	43	24
C3	180	130	212	101

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

I	X	Nearest Codeword	Euclidean distance			CBi CB0				
			C1	C2	C3					
0	X1	C3	248.41	225.942	216.959	C1	32	177	143	210
	X2	C2	270.7	99.64	126.831	C2	196	16	46	24
	X3	C1	63.93	351.763	255.421	C3	180	130	212	101
	X4	C3	226.17	146.952	86.9368	CB1				
	X5	C1	195.7	243.563	263.989	C1	59.5	136	63.5	240
1	X1	C3	211.99	197.74	104.896	C2	212	76	123	36
	X2	C2	268.35	0	103.384	C3	203	150	88	98.5
	X3	C1	107.4	317.134	246.25	CB2				
	X4	C2	244.72	65.437	104.896	C1	59.5	136	63.5	240
	X5	C1	107.4	257.919	212.728	C2	188.5	92	139	38.5
2	X1	C3	211.99	134.142	0	C3	241	192	21	156
	X2	C2	268.35	68.9227	197.74	CB3				
	X3	C1	107.4	267.536	260.083	C1	59.5	136	63.5	240
	X4	C2	244.72	79.9229	209.793	C2	188.5	92	139	38.5
	X5	C1	107.4	223.534	212.859	C3	241	192	21	156

2.2 LOOK-UP TABLE /INDEX TABLE

2	2	1
3	3	2
3	2	1

DECODING

188.5	139	188.5	139	59.5	63.5
192	38.5	192	38.5	136	240
241	21	241	21	188.5	139
192	156	192	156	192	38.5
241	21	188.5	139	59.5	63.5
192	156	192	38.5	136	240

RECONSTRUCTED IMAGE

A.2.1.4 WORKING OF VECTOR QUANIZATION

Representation of this encoding and decoding process is shown in Fig. 2.2

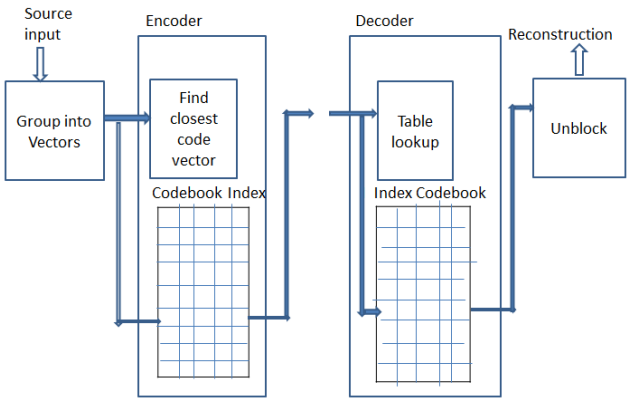

Fig. 2.2 Encoding and Decoding process of vector quantization

A.2.1.5 LIMITATIONS OF VQ_LBG

1. The time complexity (operations required to search for the best code-word i.e. computational cost) for the vector quantization is more comparative to other algorithms.
2. Storage needed for codebook, called as memory cost is high. So VQ need a high implementation costs in terms of space and time which is a major drawback.

A.2.2 DWT_VQ

This procedure is one-dimensional (1-D) DWT and Fig. below shows the schematics of this method.

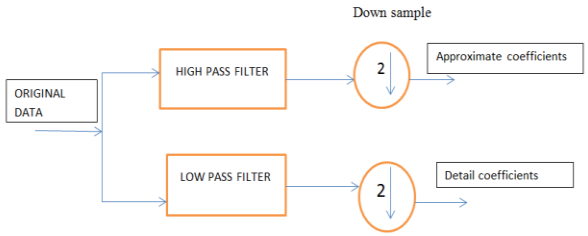

Fig. 2.3 Block Diagram of 1 -D forward DWT

If there should be an occurrence of 2-D DWT, the info information is gone through arrangement of both low pass and high pass channel in both two ways for example lines and segments. The yields are then down sampled by 2 in each direction as if there should be an occurrence of 1-D DWT. The total procedure is shown in Fig. 2.4 and Fig. 2.5. The yield is acquired in set of four coefficients LL, HL, LH and HH. The first alphabet set speaks to the change in row while the second alphabet set speaks to change in column. The alphabet L means low pass signal and H implies high pass signal. LH sign is a low pass signal in row and a high go column. Consequently, LH sign contain even components. Essentially, HL and HH contains vertical and corner to corner components, separately.

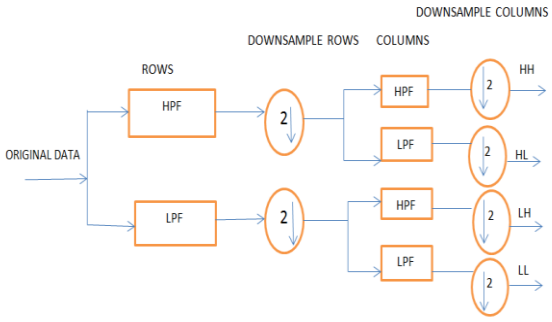

Fig. 2.4 Block diagram of 2-D forward DWT

In DWT reconstruction, input information can be accomplished in various goals by deteriorating the LL coefficient further for various dimensions. So as to reproduce the yield information, the compacted information is up-inspected by a factor of 2. The signal is additionally gone through a similar arrangement of high pass and low pass filter in the two rows and columns. The whole reconstruction method is shown in Fig. 2.5.

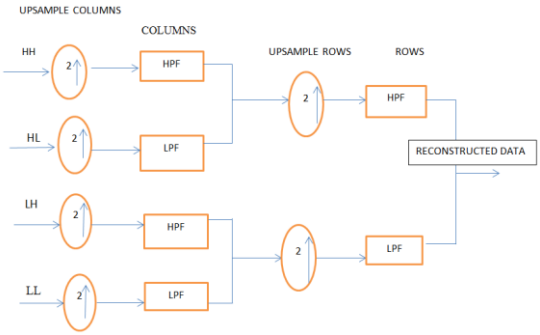

Fig. 2.5 Block diagram of 2D Inverse DWT

In this paper, an image is first separated into four sub-groups utilizing discrete wavelet transform. The principal level disintegration is performed to deteriorate the info information into a guess and the detail coefficients. In the wake of acquiring the changed grid, the detail and rough coefficients are isolated as LL, HL, LH, and HH coefficients. Every one of the coefficients are disposed of, with the exception of the LL coefficients as the significant data lives just in LL band and all other detail sub-groups contain less significant data. The LL coefficients are additionally changed into the second dimension as appeared in Fig. 2.6. The procedure proceeds for up to third level decomposition.

In the wavelet decomposition of the image, most of the information often resides in the lowest frequency sub-band, so LL sub band is only considered for further decompositions.

Fig. 2.6 Decomposition structure of an image "Cameraman" using DWT

A.2.2.1 ALGORITHM COMPUTATION_ 2D-WAVELET

INPUT: Image of (size=512*512)

OUTPUT: Approximate coefficient(LL sub-band)
Detail coefficients(HL, LH, HH sub-bands)

1. Input image is partitioned into two halves in horizontal direction using HPF & LPF.
2. The result is downsampled by a factor 2.
3. The downsampled output is partitioned in the vertical direction by the same filters HPF & LPF.
4. The output of filters are downsampled by a factor 2 and results four sub-bands(coefficients LL, LH, HL, HH).

A.2.2.2ALGORITHM

ENCODING_BITSTREAM_GENERATION

INPUT: LL, LH, HL, HH sub-bands

OUTPUT: LL-bitstream, LH-bitstream, HL-bitstream, HH-bitstream

1. LL BITSTREAM
   LL sub-band is encoded using Huffman dictionary.
2. LH BITSTREAM
   VQ encoding on LH sub-band and result codebook & look-up table.
   Generate LH bitstream using look-up table and Huffman dictionary.
3. HL BITSTREAM
   VQ encoding on HL sub-band and result codebook & look-up table.
   Generate HL bitstream using look-up table and Huffman dictionary.
4. HH BITSTREAM
   VQ encoding on HH sub-band and result codebook & look-up table.
   Generate HH bitstream using look-up table and Huffman dictionary.
5. Form a 1-D vector by combining LL, LH, HL, HH bitstreams.
   LL bitstream vector (size 442652*1)
   LH bitstream vector (size 41177*1)
   HL bitstream vector (size 41103 *1)
   HH bitstream vector (size 42948*1)

A. 2.2.3 ALGORITHM DECODING_FROM_BITSTREAMS

INPUT: All sub-band bitstream

OUTPUT: Reconstructed image

1. Extract all the sub-band bitstreams.
2. Apply Huffman decoding on each sub-band bitstream to generate LL sub-band coefficients and look-up table of other sub-bands.
3. Apply VQ decoding process to extract each sub-band channels using the above look-up tables and codebooks generated at encoding process.
4. Then Inverse DWT is applied to all the four sub-bands to reconstruct the image

ALGORITHM COMPRESSION-RATIO

1. Input image size 512*512*8
2. Total length of all bitstreams generated at encoding process:

(Length (A1_bitStream) +length (H1_bitStream) +length (V1_bitStream)
+length (D1_bitStream))= 567880

3. Compression ratio=(Image size* Bit depth) / Total length of all bit streams

CR=3.692

A.2.3 DWT_MSVQ

Multistage vector quantization (MSVQ) is a methodology that decreases both the encoding intricacy and the memory necessities for vector quantization, particularly at high rates. In this methodology, the info is quantized in various stages. The contribution to the nth-organize vector quantizer is the distinction between the first info and the recreation acquired from the yields of the former (n-l) stages. The contrast between the contribution to a quantizer and the reconstruction value is called the residual, thus MSVQ is otherwise called Residual vector quantizer (RVQ). The reconstruction vector is the sum of the outputs points of each stage. This coding plan is for grey scale images of dimension 512*512 pixels. The compression scheme performs two phases of vector quantization on discrete wavelet change (DWT) coefficients of high frequency(details) sub- groups and the low frequency (estimate) sub-band is kept without vector quantization and entropy coding (Huffman coding) is connected on the guess sub-band and the detail's sub-groups files. A joined DWT and MSVQ use to take points of interest given by the two to get a high-compression ratio with acceptable recovered image quality in term of PSNR as opposed to utilizing every technique exclusively.

The fundamental thought of multistage vector quantizer (MSVQ) is to separate the encoding task into progressive stages, where the first stage plays out a moderate quantization of the information vector utilizing a little codebook. At that point, a moment organize quantizer works on the error vector between the first and quantized first stage output. The residual vector in this manner gives second approximation of the original input vector, consequently prompting an accurate representation of the input vector. The input vector is quantized by the underlying or first stage vector quantizer meant by VQ1 whose codebook is C1 = c10, c11 ...c1 (N1−1) with codebook measure N1. The quantized guess ˆx1 is then subtracted from x creating the error vector. This error vector is then connected to a second vector quantizer VQ2 whose codebook is C2 = c20, c21... c2 (N2 −1) with size N2 yielding the quantized.

The encoder transmits a couple of lists indicating the chose code word for each stage and the undertaking of the decoder is to perform two table lookups to produce and afterward aggregate the two codewords. In fact, the overall code word or indices is the concatenation of code words or indices chosen from each of two codebooks. Subsequently, the equivalent product codebook can be created from the Cartesian product C1xC2 where C1 and C2 are the two codebooks separately. Contrasted with the full-search VQ with the item codebook C, the two-phase VQ can reduce the complexity from N = N1 × N2 to N1 + N2 where N1 and N2 are the two different codebook sizes taken individually. The multistage vector quantization framework for two phases appears in Fig. 2.7. In the Fig., 'X' speaks to the info vector, LUT represents lookup table and i1, i2, represents lookup table indices from two different stages. The overall index is the concatenation of each indices of the two codebooks. From the Fig 2.8, it is obvious that the input vector is given only to the first stage, though the input to the second stages is the error vectors from the previous stage which is denoted by error vector Xˆ is

the reconstructed signal at the decoder end.

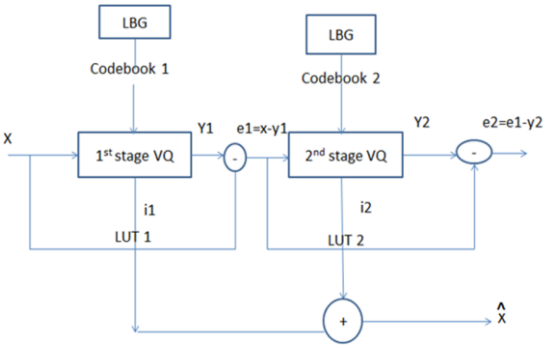

Fig. 2.7 Two stage multi stage vector quantization system

A.2.3.1 MSVQ ENCODER

In MSVQ Encoder as appeared in Fig. A.2.8, the input vector 'X' is quantized with the first stage codebook creating the first stage code vector Q0(X), a remaining vector y0 is framed by subtracting Q0(X) from 'X'. At that point, y0 is quantized utilizing the second stage codebook, with precisely a similar strategy as in the first stage, yet with 'y0' rather than 'X' as the input to be quantized. Subsequently, in each stage except from the last stage, are residual vector is created and go to the following stage to be quantized autonomously of different stages. MSVQ is an error refinement process where input to a stage are residual vectors from the last stage and they will, in general, be less and less corresponded as the procedure proceeds.

A.2.3.2 MSVQ DECODER

The decoder as shown in Fig. 2.8 receives an index identifying the stage code vector selected for each stage and forms the reproduction X by summing the identified vectors. The general quantization error is equivalent to the quantization residual from the last stage. Consecutive looking of the stage codebooks prompts the encoding complexity to the storage complexity.

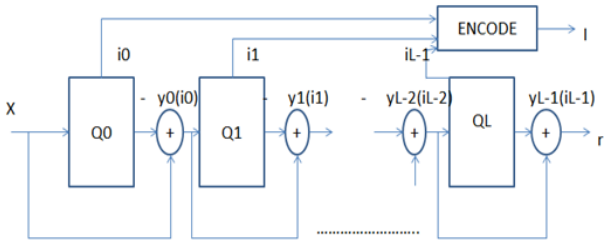

Fig. 2.8 Encoder block diagram of vector quantization for L stages

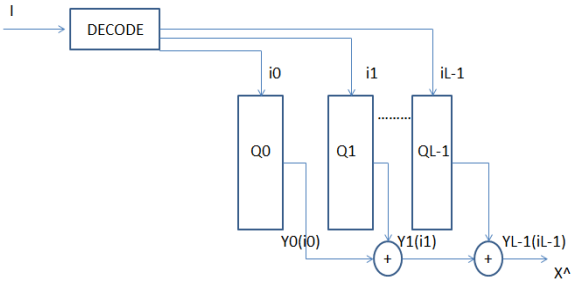

Fig. 2.9 Decoder block diagram of vector quantization for L stages

A.2.3.3 ENCODING & DECODING PROCESS OF IMAGE COMPRESSION USING DWT-MSVQ

A DWT-MSVQ and Huffman coding is a compression scheme that reduces the encoding complexity and storage requirement at a high rate. It builds the compression ratio keeping the reconstructed image quality not to be degraded. In the wavelet decomposition of the image, most of the data are in lowest frequency sub-band. Subsequently, Huffman coding (lossless compression) is applied to significant low frequency coefficients to improve imge quality in term of PSNR. MSVQ is used to quantize the coefficients of other high frequency sub-groups utilizing two phases then Huffman coding is applied to the indices of these sub- groups to build the compression ratio. The flowchart of MSVQ utilized in the proposed scheme is shown in detail in Fig. 2.10. In the proposed scheme many wavelet filters, for example, Haar, Daubechies wavelet channels, for example, db2, db3, db4, db5, coiflet filters like coif l, coif2 and the bi-orthogonal 9/7 (bior4.4) and so forth. The consequences of utilizing the bi-orthogonal 9/7 wavelet filter and Haar channels to decompose and reconstruct the image in the proposed scheme are better than all other wavelet filters output referenced previously. Fig. 2.10 and 2.11 shows the block diagram of the encoder and of the MSVQ scheme for first level decomposition of the image respectively.

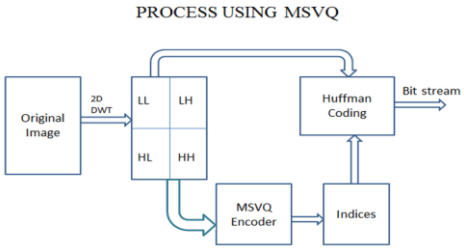

Fig. 2.10 Encoder block diagram of the MSVQ scheme for one level decomposition using 2D DWT

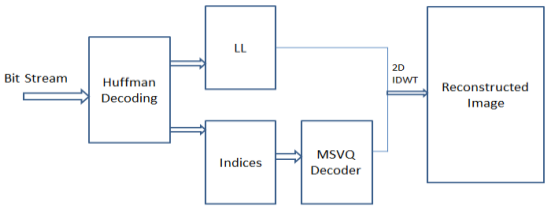

Fig. 2.11 Decoder block diagram of the MSVQ scheme for one level decomposition using 2D DWT

A.2.3.4 ALGORITHM DWT_MSVQ_ENCODING

The LL sub-band is Huffman encoded to generate the LL bit stream.

The HL sub-bands of size 256*256 is VQ encoded to generate the look-up table I1H1 of (size 64*64).

After 1st stage VQ encoding, we found the image Y1 of size 256*256 and the error vector e1 (size=256*256) is calculated by subtracting it from the original image i.e. e1=X-Y1.

The error vector is given as input to the 2nd quantizer stage for VQ encoding to generate look up table I2H2 (size 128*128).

After 2nd stage VQ encoding, image Y2 of size 256*256 is found.

These sub-bands are Huffman encoded to get the HL bitstream.

Similar process is carried out for the HL and HH sub-band to generate bitstream.

A.2.4 FUZZY C-MEANS CLUSTERING (FCM)

Fuzzy C-implies (FCM) calculation is one of the grouping systems utilized broadly that depends on the idea of fuzzy C-segment. The FCM calculation has wide applications in the region of information mining, image segmentation, and pattern recognition and image classification and so on.

FCM is an unsupervised grouping algorithm that has been effectively applied to various issues. FCM adopts fuzzy allotments to make each given estimation of input contribution somewhere in the range of 0 and 1 so as to decide the level of it having a place with a gathering. FCM is a fuzzy grouping strategy enabling a bit of information to have a place with at least two bunches. The fuzzy set has been applied to deal with uncertainty

In Fuzzy grouping, each point has a level of having a place with bunches, as in fluffy rationale, as opposed to having a place totally with only one group. Thus, points on the edge of a cluster may be in the cluster to a lesser degree than points in the center of cluster.

The FCM algorithm attempts to partition a finite collection of elements X= {X1, X2,…Xj} into a collection of c fuzzy clusters with respect to some given criterion.

The algorithm is based on minimization of the following objective function:

$$J_m(U,V) = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m} d^2(X_j, V_i) \quad \text{..... (2.2)}$$

Subject to constraint $\sum_{i=1}^{c} u_{ij} = 1, \forall j$ and

$$d^2(X_j, V_i) = \|X_j - V_i\|_A^2 \quad \text{..... (2.3)}$$

Degree of fuzzification, m≥1

and A-norm is defined by

$$\|X\|_A = \sqrt{\langle X, X\rangle_A} = \sqrt{X^T A X} \quad \text{..... (2.4)}$$

where *m* (the Fuzziness Exponent) is any real number greater than 1,

n is the number of data, *C* is the number of clusters,

U_{ij} is the degree of membership of X_j in the cluster i,

X_j is the jth of d-dimensional measured data, Vi is the d-dimension center of the cluster,

and ||*|| is any norm expressing the similarity between any measured data and the center.

By zeroing the gradient of J_m with respect to V, we get the following membership equation as

$$U_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{d^2(X_j,V_i)}{d^2(X_j,V_k)}\right)^{\frac{2}{m-1}}}, \forall i \quad \text{..... (2.5)}$$

Similarly, by zeroing the gradient of J_m with respect to U, then we get the centroid equation as

$$V_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} X_j}{\sum_{j=1}^{n} u_{ij}^{m}}, \forall i \quad \text{..... (2.6)}$$

A.2.4.1 ALGORITHM **Fuzzy**_cmeans_clustering

INPUT: Image of size 512*512, Block size=4*4, No. of clusters=c=64

OUTPUT: Reconstructed FCM image

1. Prepare training vectors of size 16384*16, cluster_n=64, data_n=16384
2. Initialize fuzzy partitioning matrix
   U=Random (cluster_n, data_n)
3. [Main loop]
   1. Update fuzzy partitioning matrix

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{d^2(X_j,V_i)}{d^2(X_j,V_k)}\right)^{\frac{2}{m-1}}}$$

   $\|X_j - V_k\|$ is the distance from point *j* to other cluster centers *k*. (as we have 2D data we use euclidean distance here).

   Where Xj=Training set={X1,X2,…….Xn},
   Vi=codebook vector,
   n is the number of training vectors,
   m is the fuzziness index
   c is the no.of clusters formed

   2. Update centroid

$$V_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} X_j}{\sum_{j=1}^{n} u_{ij}^{m}}$$

4. Compute Objective function

$$J_{FCM} = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m} d^2(X_j, V_i)$$

5. Check the termination condition
   If abs (obj_fcn (i)-obj_fcn (i-1)) < minimum amount of improvement
   break the main loop
   end if

The FCM algorithm consists of various steps. In the first step, we are dividing the image of size 512*512 into blocks of size (say 4*4).

Number of clusters is equal to c, 2<c<n, n is the no of training vectors. This results 16384 16-dimensional training vectors .In the 2nd step, we are initializing the fuzzy partition matrix or membership function U_{ij}. In the 3rd step we are updating the membership function U_{ij} and the center V_i. Also the value of objective function J-fcm is calculated. The iteration continues until the objective function value minimizes and reaches to a local minimum. Thus the codebook for fuzzy c-means is designed. Now, by using the produced new codebook and look-up table we are decoding the image for image compression using FCM.

Example:

Let x= [2 3 4 5 6 7 8 9 10 11] =Training vectors, m=2=Fuzziness parameter, Number of cluster C=2, c1=3, c_2=11.

Step 1: For first iteration calculate membership matrix using the equation

$$U_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{d^2(X_j,V_i)}{d^2(X_j,V_k)}\right)^{\frac{2}{m-1}}}, \forall i \quad \text{..... (2.7)}$$

For node 2 (1st element):

U11 = 98.78% =0.9878, membership of first node to first cluster

U12 =1.22% membership of first node to second cluster

For node 3 (2nd element):

U21 = 100% membership of second node to first cluster

U22 = 0% membership of second node to second cluster

For node 4 (3rd element):

U31 = 98% membership of first node to first cluster

U32 =2% membership of first node to second cluster

and so on until we complete the set and get U matrix and so on until we complete the set and get U matrix

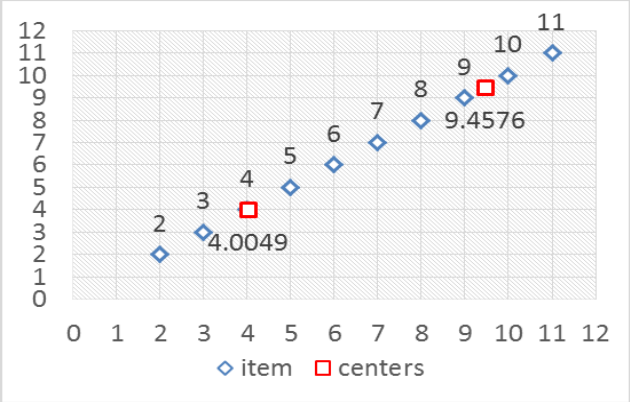

FIG. 3.13 Center calculations of data items for 1st iteration in fuzzy c-means clustering

Output for first iteration

X	cluster1	cluster2
2	0.9878	0.0122
3	1.0000	0
4	0.9800	0.0200
5	0.9000	0.1000
6	0.7353	0.2647
7	0.5000	0.5000
8	0.2647	0.7353
9	0.1000	0.9000
10	0.0200	0.9800
11	0	1.0000

Step2: Now we compute new centers

$$C_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}}, \forall\, i \quad \text{..... (2.8)}$$

c1= 4.0049 and c2=9.4576

Step 3: Repeat step until there is visible change.

Final iteration:

U =

X	cluster1	cluster2
2	0.9357	0.0643
3	0.9803	0.0197
4	0.9993	0.0007
5	0.9303	0.0697
6	0.6835	0.3165
7	0.3167	0.6833
8	0.0698	0.9302
9	0.0007	0.9993
10	0.0197	0.9803
11	0.0642	0.9358

So, c1 = 3.8688 c2 = 9.1314

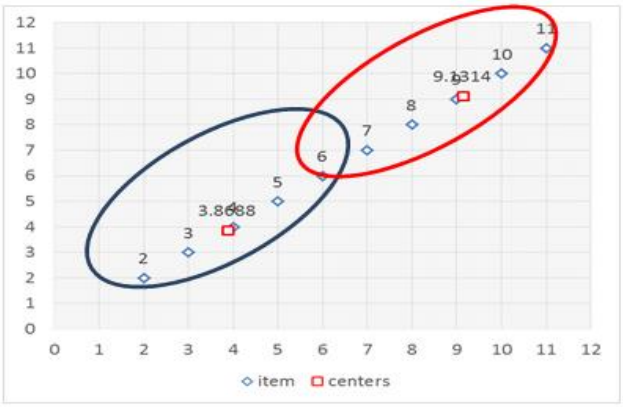

FIG. 3.14 Clusters formed with centers calculated at final iterations

A.2.5 GIFP-FCM

The proposed GIFP-FCM algorithm employs improved fuzzy partitions which is a modification of conventional FCM algorithm. The generalized improved fuzzy partitions FCM, another clustering technique is mainly used to reward crisp membership degree. Transition from fuzzy to crisp mode is efficient here. This method has fast convergence and easy to implement. It is independent of the initial codebook taken. So initially a random codebook is taken originally. This GIFP-FCM technique increases the flexibility of algorithm & clustering process by varying the fuzziness index value (i.e. M value can be 1.2, 1.5, 2, 3, 4 etc.). It uses a technique called RPCL (Rival Penalized Competitive learning). RPCL technique rewards the winning structure by modifying its weight to to adapt to particular input & simultaneously punishes the rival structure by a small learning rate. It is used in clustering process where the biggest membership function is reinforced and suppresses memberships of all other cluster center with the training vector. GIFP-FCM clustering process rewards crisp membership degrees.

For a single training vector X_j the membership constraint function is defined as,

$$f(u_{ij}) = \sum_{i=1}^{c} u_{ij}\,(1 - u_{ij}^{m-1}) \quad \text{..... (2.9)}$$

It reaches its minimum by assigning only certain u_{ij} with biggest membership value of 1, while other $u_{kj}(k \neq i)$ are suppressed. The following objective function is calculated by minimizing $f(u_{ij})$.

$$J_{GIFP-FCM} = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^{m} d^2(X_j, V_i) + \sum_{j=1}^{n} a_j \sum_{i=1}^{c} u_{ij(1-u_{ij}^{m-1})} \quad \text{..... (2.10)}$$

The minimization of $J_{GIFP-FCM}$ results the following center and membership update equations.

$$V_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}}, \forall\, i \quad \text{..... (2.11)}$$

and

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d^2(x_j, v_i) - a_j}{d^2(x_j, v_k) - a_j}\right)^{\frac{1}{m-1}}} \quad \text{.....(2.12)}$$

Here we are introducing a parameter aj called as maximal reward term given as

$a_j = \alpha \min \{ d^2(X_j, V_s) \mid s \in \{1,2\ldots..c\} \}$, $0 \leq \alpha \leq 1$ .....(2.13)

because the term $\{d^2(X_j, V_i) - a_j\}$ need to be positive, else if a_j exceeds $d^2(X_j, V_i)$ for certain 1≤i≤c, then there would be violation of the constraint $0 \le u_{ij} \le 1$.

Then the update membership equation is given by,

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left\{\frac{d^2(x_j,v_i)-\alpha \min_{1\le s\le c} d^2(x_j,v_s)}{d^2(x_j,v_k)-\alpha \min_{1\le s\le c}\left(d^2(x_j,v_s)\right)}\right\}^{\frac{1}{m-1}}} \quad \text{..... (2.14)}$$

Then we are updating equations 1 and 2 till the final codebook converges. At the end of the maximum no of iteration, when the tolerance level is reached, the codebook design is completed. This GIFP-FCM is one of the efficient methods for design of the codebook for image compression by vector quantization.

A.2.5.1 Algorithm GIFP_FCM

INPUT: Image of size 512*512, Block size=4*4, No. of clusters=k=64

OUTPUT: Reconstructed FCM image

1. Prepare training vectors of size 16384*16, cluster_n=64, data_n=16384
2. Initialize fuzzy partitioning matrix

U=Random (cluster_n, data_n)

3. [Main loop]

1. Update centroid

$$V_i = \frac{\sum_{j=1}^{n} u_{ij}^m x_j}{\sum_{j=1}^{n} u_{ij}^m}$$

2. Compute the distance matrix between each row in center and each row in data (training vectors)

3. Reward term

aj=α min$\{d^2(X_j, V_s)|$ s ∈ {1,…..c}}

4. Compute Objective function

$$J_{FCM} = \sum_{i=1}^{c}\sum_{j=1}^{n} u_{ij}^m d^2(X_j, V_i) + \sum_{j=1}^{n} a_j \sum_{i=1}^{c} u_{ij\,(1-u_{ij}^{m-1})}$$

5. Update fuzzy partitioning matrix

$$u_{ij} = \frac{1}{\sum_{k=1}^{c}\left(\frac{d^2(X_j, V_i) - aj}{d^2(X_j, V_k) - aj}\right)^{\frac{1}{m-1}}}$$

5. Check the termination condition

If abs (obj_fcn (i)-obj_fcn (i-1)) < minimum amount of improvement

break the main loop

end if

6. Exit main loop

III. SIMULATION RESULTS: A PERFORMANCE EVALUATION

A.3.1 OBJECTIVE EVALUATION PARAMETERS

Image compression technique presents some measure of bending in the reconstructed image. So, assessment of the image quality is a significant factor. The nature of reproduced images can be assessed regarding objective measure and subjective measure. In objective assessment, statistical properties are considered though, in subjective assessment, viewer see and examine image legitimately to decide the image quality. The objective assessment measures incorporate peak signal to noise ratio (PSNR), compression ratio (CR), distortion(D), relative repetition, bit rate, mean square error and so on. Image having same PSNR worth may have different perceptual quality. By changing the parameters, tradeoffs can be accomplished for compressed image against reconstructed image quality over wide a range. So as to have a visual observation, subjective assessment of a image has likewise been performed and is additionally portrayed in this paper.

The target quality measurement incorporates the statistical analysis of an input. There are different objective assessment parameters. The most ordinarily utilized parameters are peak signal to noise ratio (PSNR), compression proportion (CR), distortion (D), mean square error. Nature of the image compression is estimated through storage requirement and transmission time. They are estimated through parameters like, compression ratio, relative excess and bit rate. A few other quality estimation factors like MSE (mean square error), PSNR (peak signal to noise ration), and so forth can be apportioned to discover how well a image is imitated regarding the reference image.

A.3.1.1Mean Square Error (MSE)

MSE is another significant performance assessment parameter for estimating the nature of packed image commonly utilized alongside the PSNR investigation. It analyzes the first image and recreated image and results the dimension of distortion. The MSE between the original data and the reconstructed data is:

$$MSE = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}[f(x,y) - f'(x,y)]^2$$

……(3.1)

Where M = Original image size N= Reconstructed image size

f(x,y)=Compressed image f'(x,y)=Original image

A. 3.1.2 Peak Signal to Noise Ratio (PSNR)

Peak Signal to Noise Ratio (PSNR) has been the most popular tool for the objective quality measurement of the compressed image. It is simple to compute.

The PSNR in decibel (dB) is

$$psnr = 10\log_{10}\left(\frac{255^2}{mse}\right) = psnr = 10\log_{10}\left(\frac{I^2}{mse}\right) \quad \text{..... (3.2)}$$

where I is allowable image pixel intensity level.

For 8 bit per pixel image, $I = 2^8 - 1 = 255$ and MSE is the mean square error. Example 3.1 PSNR COMPARISION Fig. 3.1(a) shows the standard image 'MRI sectional image' of size 512*512. The image is compressed using VQ with two different compression levels to obtain different *PSNR* values. The reconstructed images are shown in Fig. 3.1(b), 3.1(c) and Fig. 3.2(b), 3.2(c) respectively. The PSNR for the image 4(b) is 34.44 dB, and (c) is 35.32 dB. It is observed that *PSNR* difference between Fig. A.3.1.1 (b) and (c) is 0.88 dB. However, when we visualize the reconstructed image, we can observe a reconstruction quality difference between the two reconstructed images.

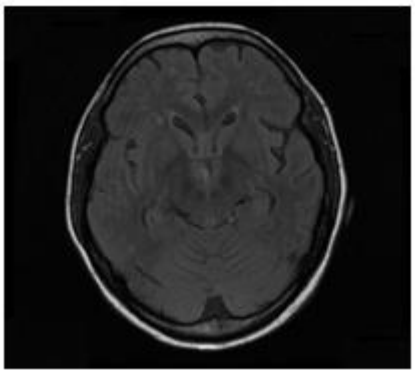

Fig. 3.1 (a) Original MRI sectional image

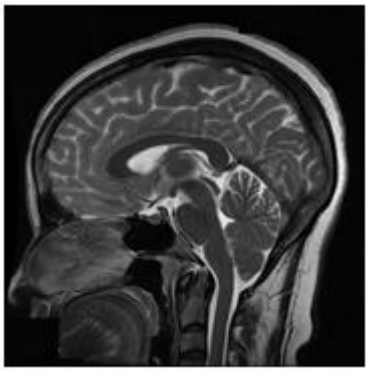

Fig. 3.1.1 (b) Original brain image

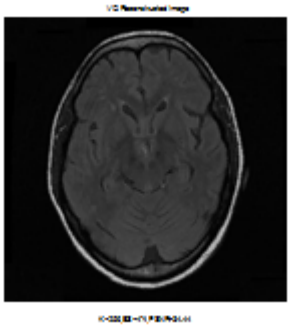

Fig. 3.1.1(c) PSNR=34.44, k=256, BS=4*4

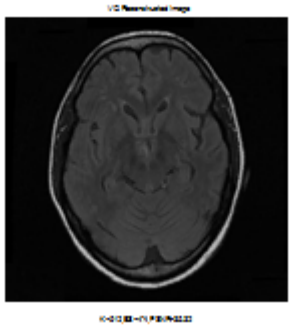

Fig. 3.2.1 (a) PSNR=35.32, BS=4*4
Reconstructed images after VQ

Fig. 3.2.1 PSNR performance comparison results for MRI sectional images

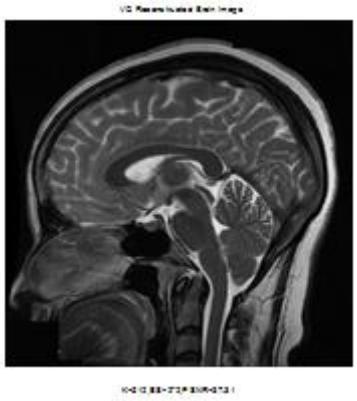

Fig. 3.2.1(b) PSNR=37.31, K=512, BS=2*2

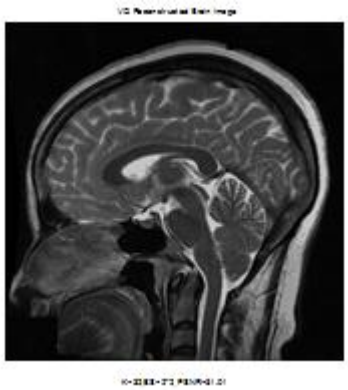

Fig. 3.2.1 (c) PSNR=31.01, K=32, BS=2*2
Reconstructed images VQ

Fig. 3.2 PSNR performance comparison results for brain images

This proves that, *PSNR* is not the sufficient measurement criteria as image is viewed as visual perception. Visual perception at the end level user has been a important issue and is growing demand especially in telemedicine, and teleconferencing areas. Hence, it is necessary to take account of both objective measure and subjective measure in order to evaluate the reconstruction quality of compressed data.

A.3.1.2 Compression Ratio (CR)

In the process of image compression, it is important to know how much detail coefficients one can discard from the input data in order to preserve critical information of the original data.

Compression ratio (*CR*) is a measure of the reduction of the detail coefficient of the data and is defined as,

$$CR = \frac{\text{Discarded data}}{\text{Original data}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.3)$$

The compression ratio can be varied to get different image quality. The more the details coefficients are discarded, the higher the *CR* can be achieved. Higher compression ratio means lower reconstruction quality of the image and vice versa.

Compression ratio depends on the bit rate that results from the Huffman coding. Therefore, the effect of the large or small code book size is less effect on PSNR and CR calculation.

Compression ratio:

Method 1

$$CR = \frac{\text{size of the input image in bits}}{\text{size of the output image in bits}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.4)$$

Method 2

$$CR = \frac{\text{Block size(BS)} * \text{BitDepth}}{\text{Block size (BS)} * \text{Bits per pixel(BPP)}} \quad \ldots\ldots\ldots\ldots\ldots (4.5)$$

where

$$BPP = \frac{\log_2^{\text{Codebook Size}}}{\text{Block Size}} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (4.6)$$

A compression ratio like 10 (or 10:1) indicates that the original image has 10 information carrying units (e.g. bits) for every 1 unit in the compressed data set.

Illustration 1. Storage requirement and transmission time considering bandwidth of 64kbps while an image is of size 1024*1024*3 of bit depth 8: The storage requirement is 1024*1024*3*8 bits and the transmission time is 1024 *1024 *3*8 seconds=64000 seconds.

Illustration 2. Compression ratio when bitmap image size of 256*256, block size 2x2

Codebook size CS=16. 8 bit image resolution.

Block size (BS) =8*8, Codebook Size (CS) =32

BPP=($\log_2$CS)/(BS)=5/64,

CR=(BS*BITDEPTH)/(BS*BPP)=(8*8*8)/(8*8*5/64)=512/5=102.4

5/64 bits required to represent 1 pixel. So (5/64)*8*8 bits are required to represent the input image.

A.3.2 RESULTS OF VQ ALGORITHM

Here image compression has been appilied on six diverse 256 grey level bitmap medicinal images with 512*512 pixels. Reconstructed image quality can be assessed objectively and subjectively. In this paper, objective assessment has been utilized to assess the compressed image in term of PSNR.

VQ result incorporates correlations between image quality when utilizing diverse information vector sizes i.e (square size) and codebook measure. The nature of the recreated image relies on the size of the input vector and codebook estimate. In the event that for a similar square size, the codebook size is expanded, we found that the compression proportion (which is the proportion of the uncompressed size to the compressed size) decreases the number of bits that represents the square is expanded and PSNR increments on the grounds that the input vectors are getting approximated or quantized by increasing number of code-words. Keeping the equivalent codebook estimate, if square size is expanded, we see that the compression ratio is increased on the grounds that the quantity of training set (total number of squares) is decreased and PSNR decreases because expanding the number of pixels that approximated by a similar number of code-words (codebook measure). Therefore, the codebook measure under 16, the PSNR become low. In this way, codebook measure which is more prominent or equivalent to 16 code-words (code- vector) has been utilized in this exploration for VQ.

Time investigation for codebook creation if there should be an occurrence of square size 64 and 256 are high contrasted with square size 16, 32, 128. Keeping square size fixed and expanded the code book measure, CR diminishes and PSNR increments.

TABLE A.3.1 RESULTS OF IMAGE COMPRESSION USING LBG_VQ FOR DIFFERENT IMAGES

Image name	Block size	Codebook size	CR	PSNR (dB)	MSE
Chest.bmp	2*2	16	8	31.45	46.55
		32	6.4	33.86	26.73
		64	5.3	36.73	13.79
		128	4.6	38.01	10.26
		256	4	39.57	7.17
	4*4	16	32	28.91	83.56
		32	25.6	31.26	48.66
		64	21.3	32.37	37.65
		128	18.3	33.61	28.26
		256	16	35.04	20.36
	8*8	16	128	26.81	135.39
		32	102	28.02	102.57
		64	85	29.05	80.85
		128	73	30.45	58.57
		256	64	31.47	46.34

Image name	Block size	Codebook size	CR	PSNR (dB)	MSE
Brain. bmp	2*2	16	8	29.83	67.50
		32	6.4	31.01	51.44
		64	5.3	33.00	32.57
		128	4.6	34.15	24.95
		256	4	35.46	18.48
	4*4	16	32	25.94	165.45
		32	25.6	26.79	135.97
		64	21.3	27.84	106.86
		128	18.3	29.27	76.77
		256	16	30.42	58.94
	8*8	16	128	23.13	315.80
		32	102	24.21	250.09
		64	85	24.83	213.85
		128	73	26.03	162.00
		256	64	26.57	142.95

Image name	Block size	Codebook size	CR (%)	PSNR (dB)	MSE
Spinal cord. bmp	2*2	16	8	29.68	69.97
		32	6.4	32.30	38.25
		64	5.3	35.03	20.38
		128	4.6	37.00	12.94
		256	4	38.36	9.46
	4*4	16	32	26.97	130.53
		32	25.6	29.80	68.04
		64	21.3	31.17	49.61

		128	18.3	32.78	34.24
		256	16	33.57	28.54
	8*8	16	128	24.65	222.65
		32	102	26.71	138.63
		64	85	27.87	105.96
		128	73	29.44	73.93
		256	64	30.17	62.47

Image name	Block size	Codebook size	CR (%)	PSNR (dB)	MSE
MRI section.bmp	2*2	16	8	33.19	31.18
		32	6.4	34.23	24.50
		64	5.3	36.19	15.62
		128	4.6	37.74	10.92
		256	4	38.89	8.38
	4*4	16	32	29.76	68.61
		32	25.6	31.41	46.97
		64	21.3	32.42	37.26
		128	18.3	33.63	28.17
		256	16	34.44	23.36
	8*8	16	128	26.44	147.26
		32	102	27.94	104.45
		64	85	28.92	83.26
		128	73	29.54	72.27
		256	64	31.05	51.03

Image name	Block size	Codebook size	CR (%)	PSNR (dB)	MSE
LENA.BMP	2*2	16	8	30.60	56.58
		32	6.4	32.55	36.11
		64	5.3	33.98	25.94
		128	4.6	35.49	18.33
		256	4	36.82	13.50
	4*4	16	32	27.30	121.07
		32	25.6	28.47	92.42
		64	21.3	29.65	70.32
		128	18.3	30.74	54.72
		256	16	31.62	44.74
	8*8	16	128	24.56	227.24
		32	102	25.55	180.92
		64	85	26.49	145.70
		128	73	27.36	119.28
		256	64	28.31	95.81

Image name	Block size	Codebook size	CR (%)	PSNR (dB)	MSE
CAMERAMAN.BMP	2*2	16	8	31.48	46.16
		32	6.4	33.54	28.71
		64	5.3	35.14	19.88
		128	4.6	37.18	12.42
		256	4	38.70	8.76
	4*4	16	32	26.80	135.57
		32	25.6	28.76	86.46
		64	21.3	29.96	65.48
		128	18.3	31.20	49.23
		256	16	32.53	36.28
	8*8	16	128	23.80	270.97
		32	102	24.83	213.70
		64	85	25.89	167.21
		128	73	26.97	130.62
		256	64	28.16	99.27

Fig. A.3.3 Graphical comparison of CR, PSNR, MSE for vector size [2 2] of spinal cord image using VQ_LBG algorithm

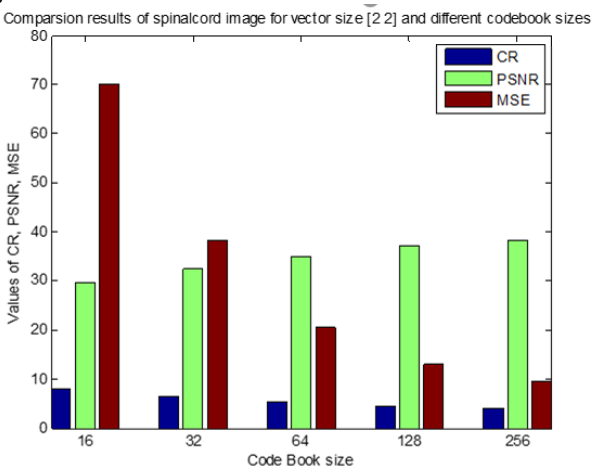

Fig. A.3.4 Graphical comparison of CR, PSNR, MSE for vector size [4 4] of spinal cord image using VQ_LBG algorithm

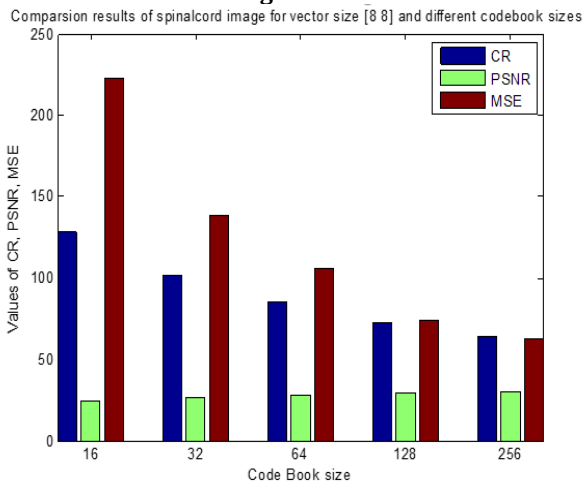

Fig. A.3.5 Graphical comparison of CR, PSNR, MSE for vector size [8 8] of spinal cord image using VQ_LBG algorithm.

A.3.3 DIFFERENT VQ_LBG RECONSTRUCTED IMAGES

(For different block sizes (BS) and codebook sizes (K) taken

Fig. 3.6 (a) k=64, BS=2*2, PSNR=33

Fig. 3.6(b) k=256, BS=2*2, PSNR=35.46

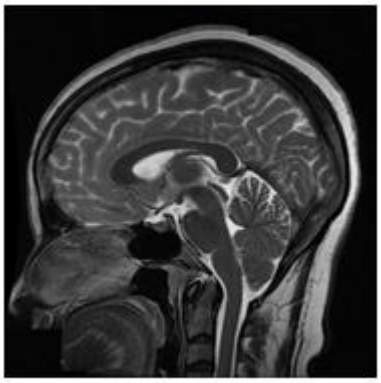

Fig. 3.6(c) k=512,BS=2*2,PSNR=37.31

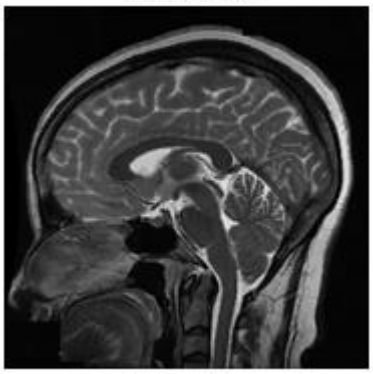

Fig. 3.6(d) k=512,BS=4*4,PSNR=31.65

Fig. 3.6(e) k=64,BS=4*4,PSNR=32.37

Fig. 3.6(f) k=256,BS=4*4,PSNR=35.04

Fig. 3.6(g) k=256,BS=4*4,PSNR=31.62

Fig. 3.6(h) k=64,BS=8*8,PSNR=26.49

Fig. 3.6(i) k=64, BS=2*2, PSNR=35.14

Fig. 3.6(j) k=256, BS=8*8, PSNR=28.16

Fig. 3.6(k) k=512BS=8*8,PSNR=31.11

Fig. 3.6(l) k=512,BS=4*4,PSNR=34

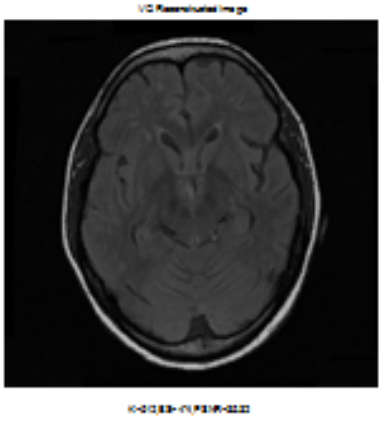

Fig. 3.6(m) k=256,BS=2*2,PSNR=37.74

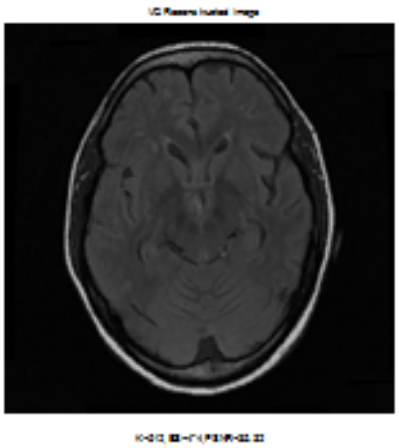

Fig. 3.6(n) k=512,BS=4*4,PSNR=35.32

Fig. 3.6 Different VQ_LBG reconstructed images with different block & codebook sizes.

A.3.4 RESULT ANALYSIS OF DWT_VQ & DWT_MSVQ USING HAAR TRANSFORM

In the proposed scheme, the first image deteriorated into various sub-groups as indicated by the decomposition level. In this task, most extreme decomposition level can become to is three levels, since when it is more prominent than three, the PSNR turns out to be low, which means high distortion in the nature of the recreated (compressed) image. The approximation sub-band (low frequency sub-band) stays without quantization in light of the fact that a large portion of the data regularly resides in it and for a similar reason Huffman coding connected on this sub-band then RVQ connected on the detail sub-groups (high frequency sub-groups) of the dimension n, (n=l, 2 or 3). The detail sub-groups contain data that is less significant than estimate sub-band; subsequently, just the detail sub- groups of last level are vector quantized by two stages, other sub-groups are set to zeros to expand the compression ratio and after that Huffman coding applied on the lists of these sub-groups.

At the point when the square size expands the compression, the ratio is likewise expanded because the number of training set decreases and this prompt decrease the indices that the Huffman coding will be applied on and the other way around. Since the estimate sub-band stays without quantization and just the detail sub-groups quantized, and the compression proportion relies upon the bit rate that outcomes from the Huffman coding; in this way, the impact of the codebook measure on the PSNR and on the image compression is little. This implies the huge or the little size of codebook has less affecting on the PSNR and on the compression ratio than unconstrained VQ. The CR and PSNR are likewise affected by the decomposing level when it increases the CR increment because expanding the number of sub-groups that set to zeros and the PSNR decrease on the grounds that the size of estimation sub-band become smaller. Table 3.2 demonstrates the impact of utilizing distinctive square size and codebook estimate on the detail sub-groups of various dimensions utilizing Haar wavelet channel.

In the proposed scheme, we utilized just two stages in MSVQ because each stage is streamlined than improving all stages at once, the codebook structure

unpredictability and memory prerequisites are diminished however at the expense of reduced PSNR. Also wavelet filter 9/7 (bior4.4) are better from the once that use the Haar (db1) wavelet filter from the view of compression ratio with keeping acceptable value of PSNR, for example; the results of MRI image (when using the block size 16*16, codebook size 4, for the two stages and third decomposition level) have CR=77.7%, PSNR=25.74 for the wavelet filter 9/7 (bior4.4), while CR=89.84%, PSNR=23.56 for the wavelet filter Haar (dbl).

RESULTS OF DWT_VQ

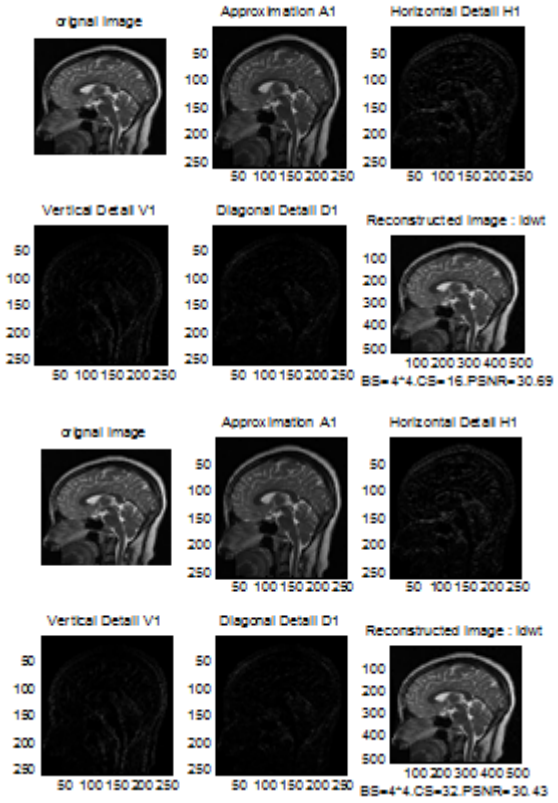

Fig. 3.7 Different DWT_VQ single stage reconstructed images

TABLE A. 3.2 RESULTS OF HYBRID DWT-MSVQ USING HAAR WAVELET FILTER

	STAGE2	IMAGE NAME	LEVEL 1 DECOMPOSITION			LEVEL 2 DECOMPOSITION			LEVEL 3 DECOMPOSITION		
BSCS	BSCS		CR	PSNR (dB)	MSE	CR	PSNR (dB)	MSE	CR	PSNR (dB	MSE
4*416	2*232	Brain.bmp	2.82	26.34	150.87	11.04	21.39	471.74	44.59	18.32	956.98
		Chest.bmp	2.58	26.38	149.40	10.10	24.15	249.81	41.02	21.32	479.42
		Spinal cord.bmp	2.51	23.64	281.19	10.29	19.87	669.02	40.79	20.85	534.57
		MRI section2.bmp	3.06	29.01	81.62	12.14	23.51	289.15	47.23	19.61	710.86
		Lena.bmp	2.52	22.73	346.11	9.97	20.46	583.68	41.91	17.9	750.05
		Cameraman.bmp	2.69	26.41	148.35	10.46	20.45	585.31	42.36	17.6	756.4
16*164	4*44	Brain.bmp	4.49	31.00	51.63	17.69	25.43	186.03	71.51	21.23	489.16
		Chest.bmp	4.03	28.11	100.46	16.06	27.62	112.42	63.64	24.64	223.08
		Spinal cord.bmp	3.98	24.73	218.53	15.89	21.42	468.66	63.33	23.22	309.36
		MRI section2.bmp	5.74	33.72	27.58	21.92	27.62	112.35	88.29	23.11	317.42
		Lena.bmp	3.91	24.23	245.40	15.68	22.26	385.59	62.62	19.55	720.13
		Cameraman.bmp	4.29	30.55	57.27	17.10	24.27	242.81	67.45	20.49	579.87
16*164	16*164	Brain.bmp	4.71	31.78	43.14	18.72	25.67	176.12	75.65	20.49	579.71
		Chest.bmp	4.21	28.48	92.10	16.72	27.81	107.47	66.97	23.73	275.13
		Spinal cord.bmp	4.14	25.32	190.92	16.61	21.96	413.69	66.39	22.77	343.27
		MRI section2.bmp	6.02	34.03	25.66	23.28	27.03	128.69	94.17	21.05	509.84
		Lena.bmp	4.11	24.29	241.81	16.41	22.21	390.52	65.59	19.30	762.46
		Cameraman.bmp	4.48	31.12	50.17	17.97	24.48	231.70	71.34	19.33	758.45

A.3.5 RESULTS OF DWT_MSVQ USING HAAR WAVELET

Stage 1: Block size=16*16, Codebook size=4

Stage 2: Block size=4*4, Codebook size=4

Level 1 reconstructed images after DWT_MSVQ

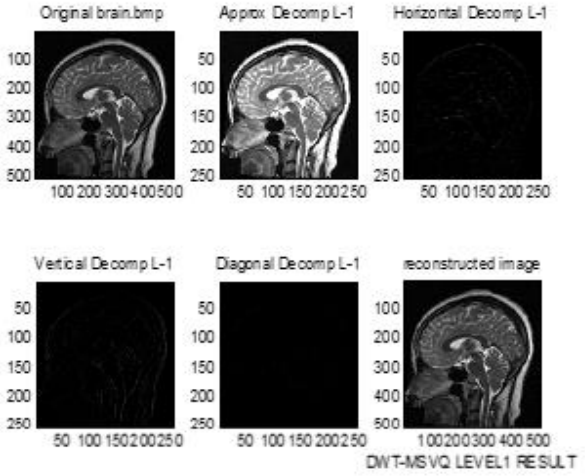

Fig. 3.8(a) CR=1.12,PSNR=30.87,MSE=53.17

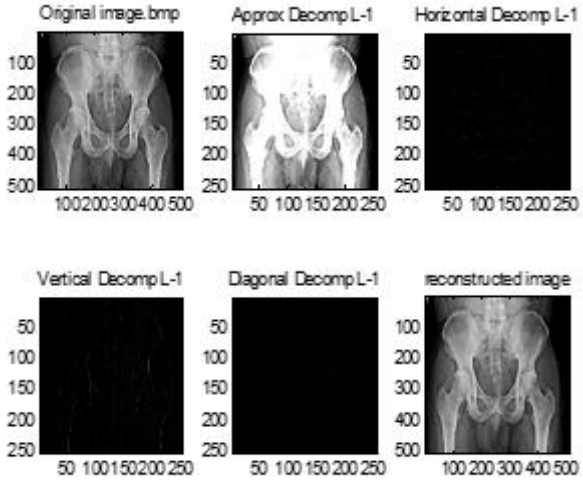

Fig. 3.8(b) CR=1.10,PSNR=28.09,MSE=100.86

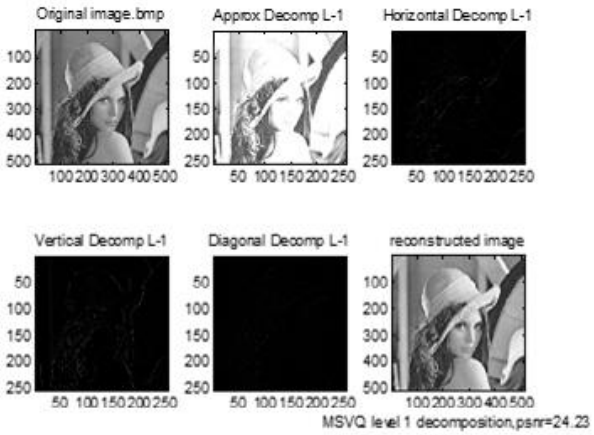

Fig. 3.8(c) CR=3.91,PSNR=24.23,MSE=245.40

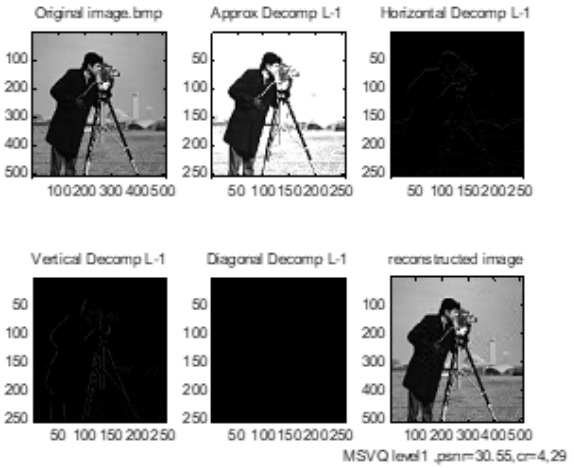

Fig. A.3.8(d) CR=4.29,PSNR=30.55,MSE=57.27

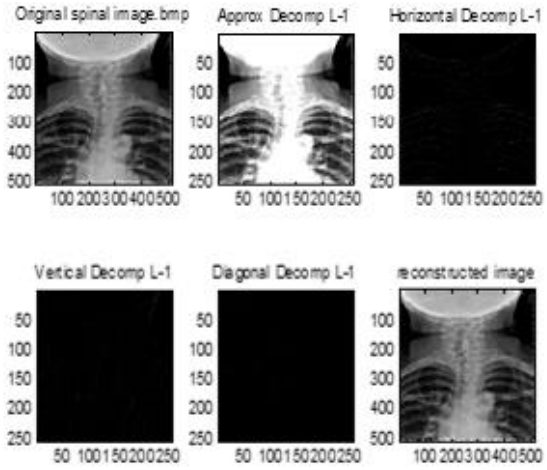

Fig. 3.8(e) CR=1.98,PSNR=25.88,MSE=167.55

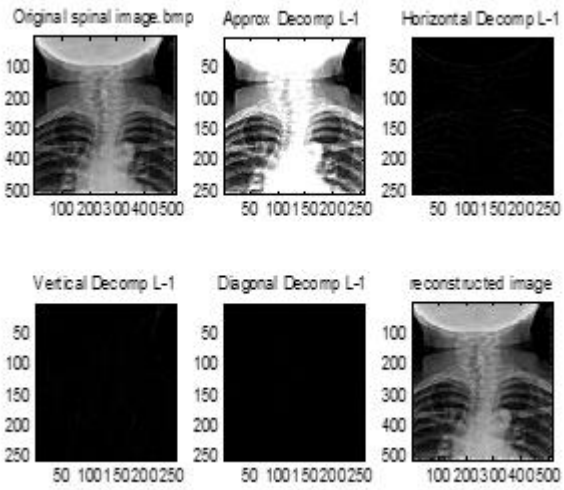

Fig. 3.8(f) CR=1.43,PSNR=33.72,MSE=27.60

Level 2 reconstructed images after DWT_MSVQ

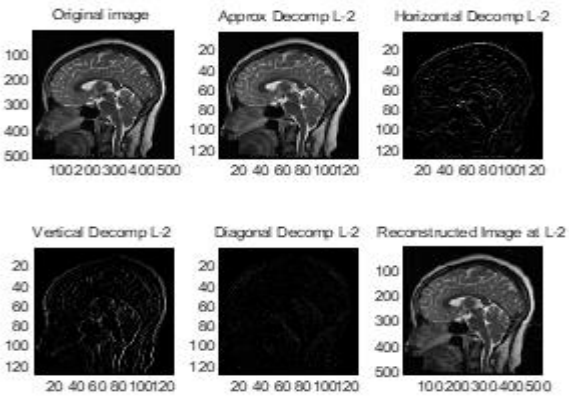

Fig. 3.8(g) CR=4.44,PSNR=25.41,MSE=1186.76

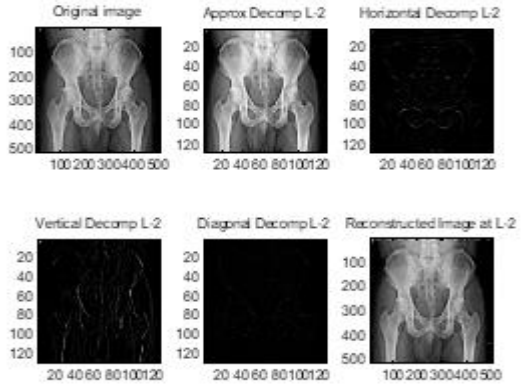

Fig. 3.8(h) CR=3.96,PSNR=27.45MSE=116.80

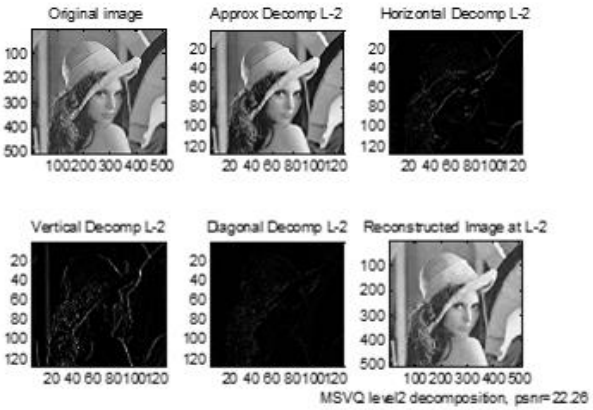

Fig. 3.8(i) CR=15.68,PSNR=22.26,MSE=385.59

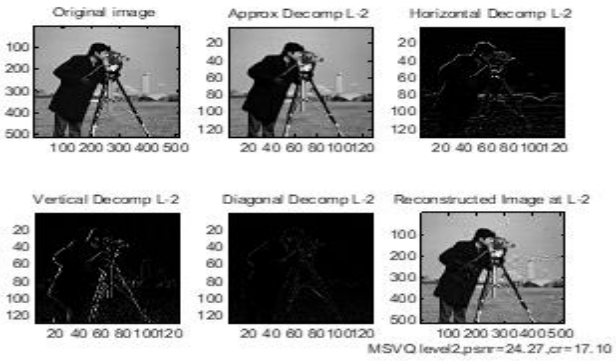

Fig. 3.8(j) CR=17.10,PSNR=24.27,MSE=242.81

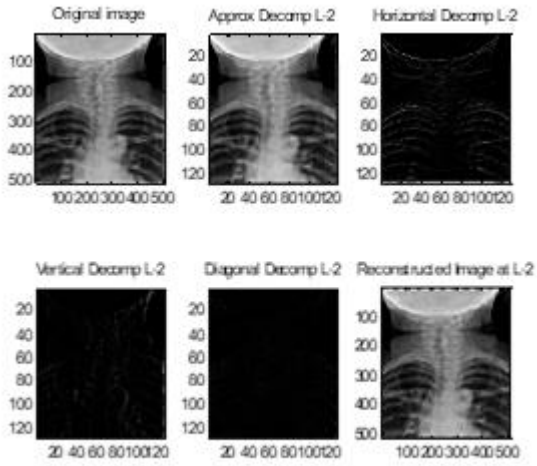

Fig. 3.8(k) CR=3.97,PSNR=23.24,MSE=308.5

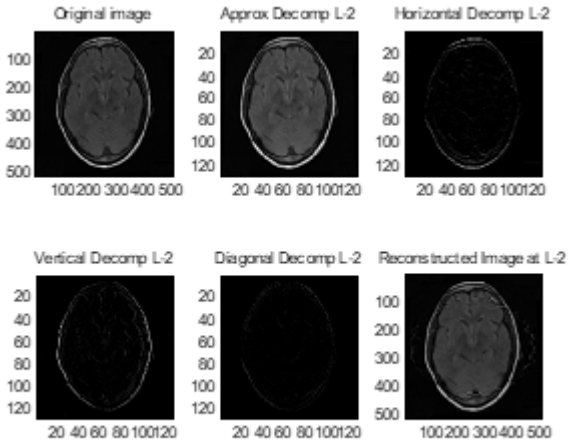

Fig. 3.8(l) CR=5.47,PSNR=27.69,MSE=110.52

Level 3 reconstructed images after DWT_MSVQ

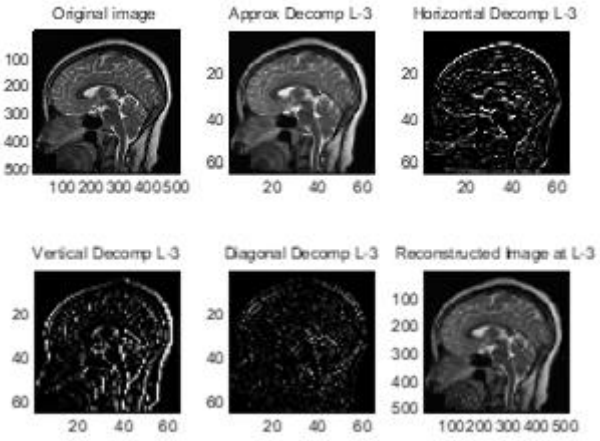

Fig. 3.8(m) CR=18.01,PSNR=21.37,MSE=473.71

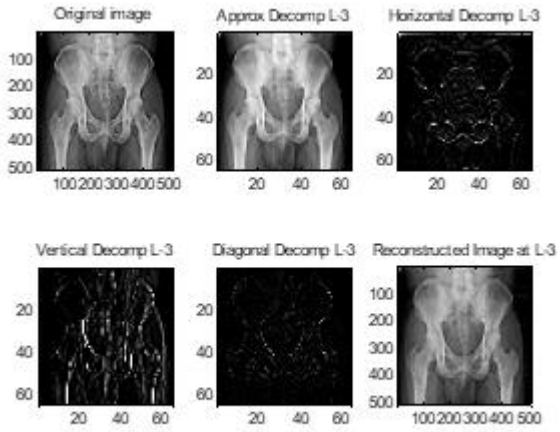

Fig. 3.8(n) CR=15.91,PSNR=24.41,MSE=235.02

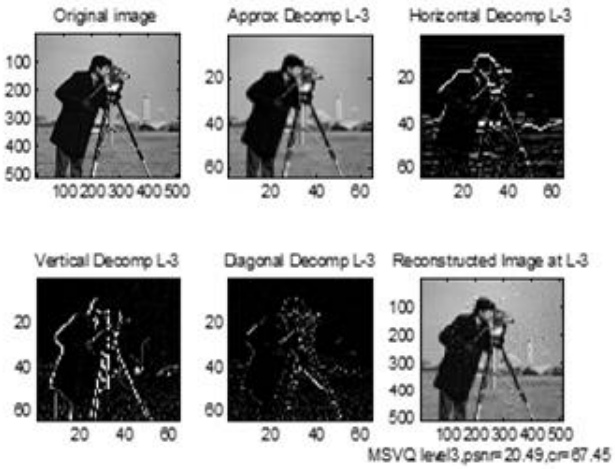

Fig. 3.8(o) CR=67.45,PSNR=20.49,MSE=579.87

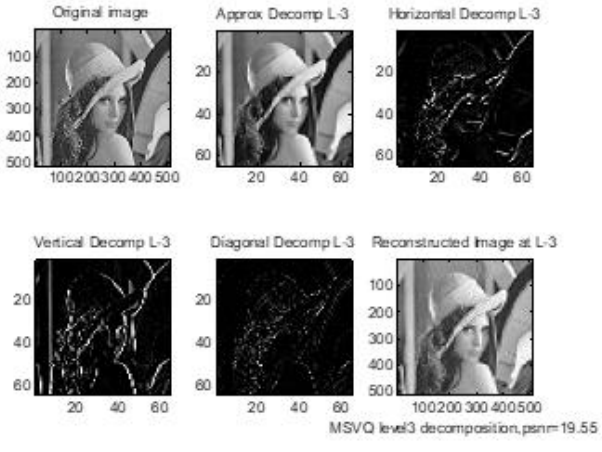

Fig. 3.8(p)CR=62.62,PSNR=19.55,MSE=720.13

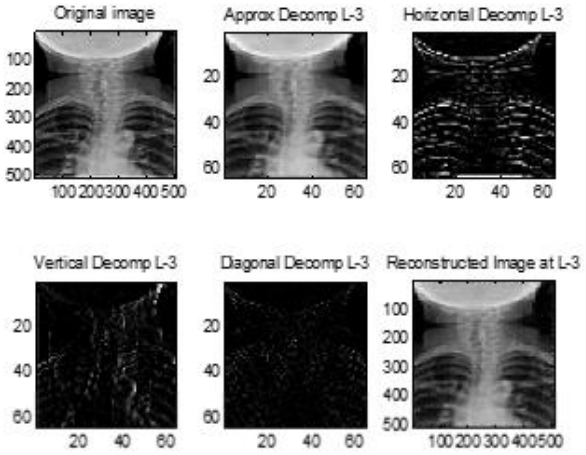

Fig. 3.8(q) CR=15.87,PSNR=23.06,MSE=320.94

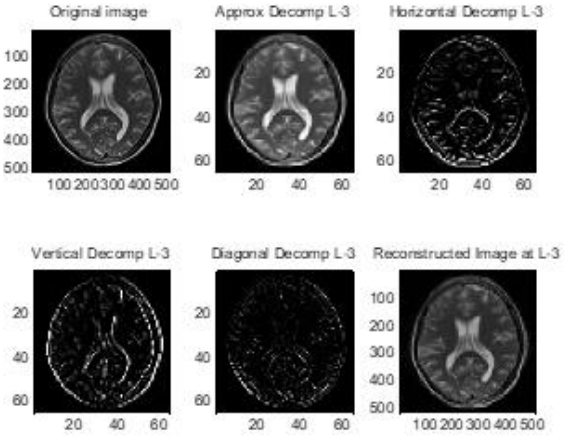

Fig. 3.8(r) CR=22.30,PSNR=22.84,MSE=338.09

Fig. A.3.8 Different DWT_MSVQ reconstructed images using Haar wavelet filter

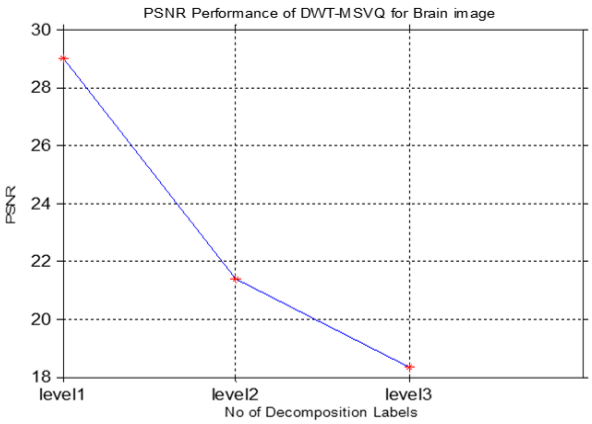

Fig. 3.9 Graphical representation of PSNR performance of DWT-MSVQ for 3 levels decomposition of Brain image

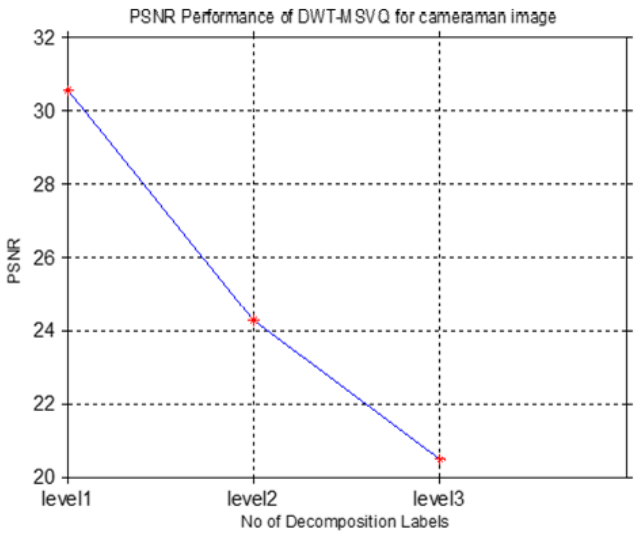

Fig. 3.10 Graphical representation of PSNR performance of DWT-MSVQ for 3 levels decomposition of Cameraman image

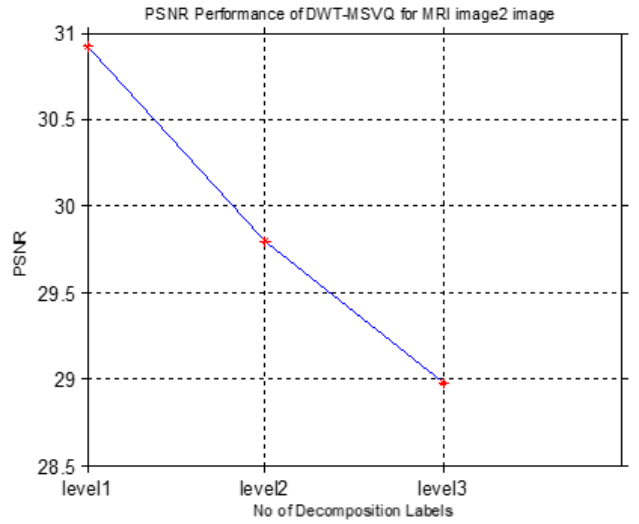

Fig. 3.11 Graphical representation of PSNR performan of DWT-MSVQ for 3 levels decomposition of MRI image2 image.

A.3.6 RESULTS OF IMAGES USING FUZZY C-MEANS METHOD

TABLE 3.3 Tabulation results of different images using fuzzy c-means method

IMAGE NAME	CODEBOOK SIZE=K	PSNR	MSE
Brain.bmp	16	26.07	160.55
	32	26.32	151.71
	64	26.47	146.36
	128	26.52	144.57
	256	26.54	144.33
	512	26.60	142.19
Chest image.bmp	16	28.63	89.16
	32	29.12	79.58
	64	29.33	75.70
	128	30.09	63.58
	256	30.15	62.77
	512	30.18	62.40
IMAGE NAME	**CODEBOOK SIZE K**	**PSNR**	**MSE**
Lena image.bmp	16	26.58	142.73
	32	26.85	134.07
	64	27.00	129.45
	128	27.05	128.11
	256	27.08	127.80
	512	28.04	124.28
Cameraman.bmp	16	25.43	185.94
	32	25.80	170.71
	64	26.25	154.17
	128	26.51	145.23
	256	26.54	144.43
	512	26.70	138.92
Spinalcord.bmp	16	25.95	164.97
	32	26.26	153.71
	64	26.39	149.32
	128	26.46	146.65
	256	26.48	146.41
	512	27.01	140.25
MRI sectional image.bmp	16	29.15	78.93
	32	29.59	71.38
	64	30.16	62.68
	128	30.26	61.16
	256	30.48	58.18
	512	30.60	56.56

3.6.1 RECONSTRUCTED IMAGES USING FCM

For Vector size or Block size=4*4 and different codebook sizes=k taken

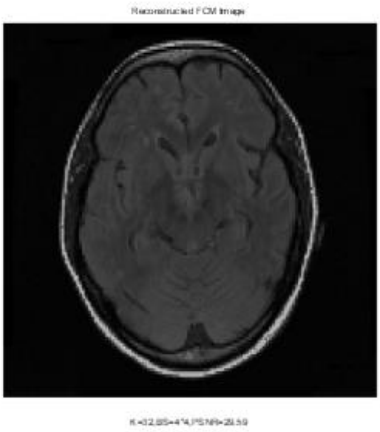

Fig. 3.12(a) k=32,PSNR=30.00,MSE=71.38

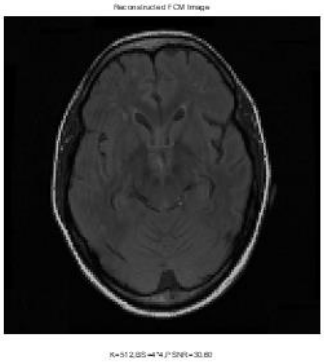

Fig. 3.12(b) k=512,PSNR=30.60,MSE=56.56

Fig. 3.12(c) k=256,PSNR=26.48,MSE=146.41

Fig. 3.12(d) k=512,PSNR=27.01,MSE=140.25

Fig. 3.12(e) k=256,PSNR=27.08,MSE=127.80

Fig. 3.12(f) k=128, PSNR=27.05, MSE=128.11

Fig. 3.12(f) k=512,PSNR=26.70,MSE=138.92

Fig. 3.12(g) k=256,PSNR=26.54,MSE=144.43

Fig. 3.12(h) k=256, PSNR=30.15, MSE=62.77

Fig. 3.12(i) k=32, PSNR=29.12, MSE=79.58

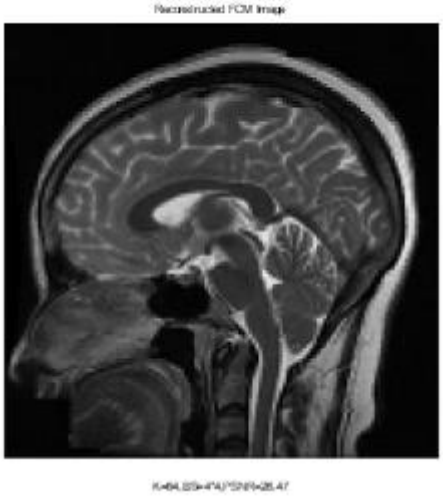

Fig. 3.12(j) k=64, PSNR=26.47, MSE=146.36

Fig. 3.12(k) k=32, PSNR=26.32, MSE=151.71

Fig. 3.12 Fuzzy c-means reconstructed images with different block and codebook sizes.

A.3.7 RESULTS OF IMAGES USING GIFP-FCM (GENERALIZED IMPROVED FUZZY PARTITIONING FCM) METHOD

The performance of GIFP-FCM based vector quantization algorithm is measured by taking six different 512*512 pixels medical images of 256 gray levels. Each image is divided into block size of 4*4 sub-images to generate 16384 16-dimensional training vectors. Table 3.4 shows the effects of variation of parameters α (reward term), M (fuzziness parameter) and different codebook sizes on the PSNR and distortion i.e. mean square value taken. The results show that for a given α value, higher the values of M result in higher distortion, lower PSNR and lesser number of iterations to converge for designing of codebook. Also, when the α and codebook size value increases, PSNR is increased with less distortion measured. But to get an optimum image quality, we must have a tradeoff between the choice of M and α value. Table 3.5 shows the Peak Signal to noise, mean square value and distortion measured for different medical images with vector size=4*4 and different codebook sizes taken.

TABLE 3.4 Comparison of performance of GIFP-FCM for brain image with different α, K and M values

Image name	Codebook size=k	Variation of α	M	PSNR (in dB)	MSE
Brain.bmp	64	0.5	1.2	30.09	68.35
			1.5	29.78	70.03
			2	28.03	102.17
	128		1.2	31.00	51.77
			1.5	30.92	69.58
			2	28.92	53.24
	64	0.7	1.2	30.46	65.00
			1.5	29.75	69.16
			2	29.09	80
	128		1.2	30.93	62.58
			1.5	30.87	63.94
			2	29.93	68.53
	64	0.9	1.2	30.85	62.05
			1.5	29.89	74.96
			2	29.50	79.54
	128		1.2	30.96	59.63
			1.5	30.74	60.12
			2	29.57	72.58

TABLE 3.5 PSNR, MSE, Distortion performance of GIFP-FCM algorithm for different images with different codebook sizes

IMAGE NAME	CODEBOOK SIZE=K	PSNR	MSE	DISTORTION
Brain.bmp	16	27.00	129.58	2073.30
	32	28.33	95.48	1527.70
	64	29.73	69.12	1106.00
	128	30.97	51.99	831.95
	256	32.12	39.86	637.86
	512	33.26	30.63	490.18
Chest image.bmp	16	29.34	75.60	1209.7
	32	31.33	47.79	764.71
	64	32.72	34.75	556.03
	128	34.04	25.59	409.49
	256	35.11	20.01	320.23
	512	36.46	14.66	234.70
Lena image.bmp	16	27.38	118.62	1898.0
	32	28.54	90.89	1454.3
	64	29.67	70.15	1122.5
	128	30.76	54.48	871.82
	256	31.68	44.13	706.14
	512	32.50	36.54	584.78

Cameraman.bmp	16	26.99	130.01	130.01
	32	28.81	85.51	85.51
	64	30.22	61.75	61.75
	128	31.59	45.08	45.08
	256	32.66	35.21	35.21
	512	34.01	25.82	25.82
Spinalcord.bmp	16	28.65	88.55	1416.8
	32	30.74	54.74	875.88
	64	32.22	38.95	623.34
	128	33.82	26.97	431.56
	256	34.90	21.01	336.25
	512	36.27	15.34	245.43
MRI sectional image.bmp	16	30.71	55.17	882.76
	32	32.35	37.84	605.58
	64	33.60	28.34	453.55
	128	34.90	21.03	336.61
	256	37.20	12.37	198.04
	512	37.74	10.93	174.98

A.3.7.1 RECONSTRUCTED IMAGES USING GIFP-FCM METHOD

Fig. 3.13(a) k=256,PSNR=32.12,MSE=39.86

Fig. 3.13(b) k=512, PSNR=36.46, MSE=14.

Fig. 3.13(c) k=256, PSNR=34.90, MSE=15.34

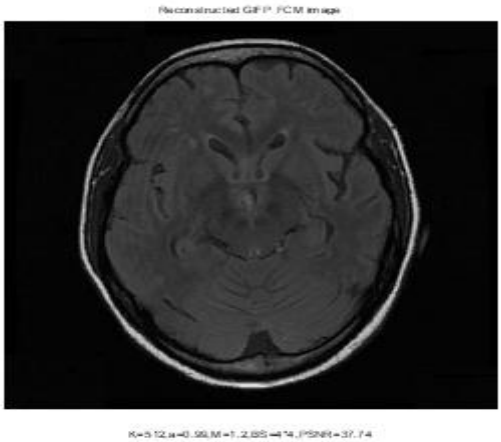

Fig. 3.13(d) k=512,PSNR=37.74,MSE=10.93

Fig. 3.13(e) k=512, PSNR=32.50, MSE=36.54

Fig. 3.13(f) k=512, PSNR=34.01, MSE=25.82

Fig. A.4.13 Different reconstructed images using GIFP-FCM METHOD

TABLE 3.6 PSNR PERFORMANCE OF MRI IMAGE2 FOR VARIOUS ALGORITHMS WITH VECTOR SIZE= [4 4]

ALGORITHMS	Codebook size K=32	Codebook size K=64	Codebook size K=128	Codebook size K=256	Codebook size K=512
1.**LBG_VQ**	31.41	32.42	33.63	34.44	35.32
2.**DWT_MSVQ** Stage1: Vec Size=[4 4],k=16 Stage1: Vec Size=[2 2],k=32	Level 1 Psnr=30.92 Mse: 81.62	Level 2 Psnr=29.80 Mse: 110.05	Level 3 Psnr=28.98 Mse: 140.82		
3.**FCM**	29.59	30.16	30.26	30.48	30.60
4.**GIFP_FCM** (α=0.99,M=1.2 used)	**32.35**	**33.60**	**34.90**	**37.20**	**37.74**

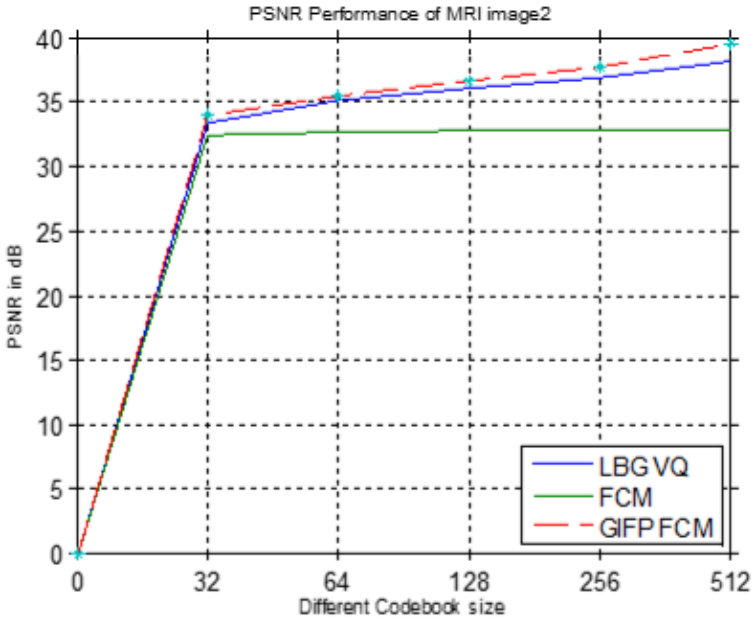

Fig. 3.14 Graphical representation of PSNR performance for MRI image2 using LBG_VQ, FCM, GIFP-FCM algorithms

TABLE 3.7 PSNR PERFORMANCE OF CAMERAMAN IMAGE FOR VARIOUS ALGORITHMS, VECTOR SIZE= [4 4]

	Codebook size K=32	Codebook size K=64	Codebook size K=128	Codebook size K=256	Codebook size K=512
1.**LBG_VQ**	28.76	29.96	31.20	32.53	33.40
3.**DWT_MSVQ** Stage1: Vec Size=[4 4],k=16 Stage1: Vec Size=[2 2],k=32	Level 1 Psnr=30.55 Mse=57.27 CR=4.29	Level 2 Psnr=24.27 Mse=242.81 CR=17.10	Level 3 Psnr=20.49 Mse=579.87 CR=67.45		
4.**FCM**	25.80	26.25	26.51	26.54	26.70
5.**GIFP_FCM** (α=0.99,M=1.2 used)	**28.81**	**30.22**	**31.59**	**32.66**	**34.01**

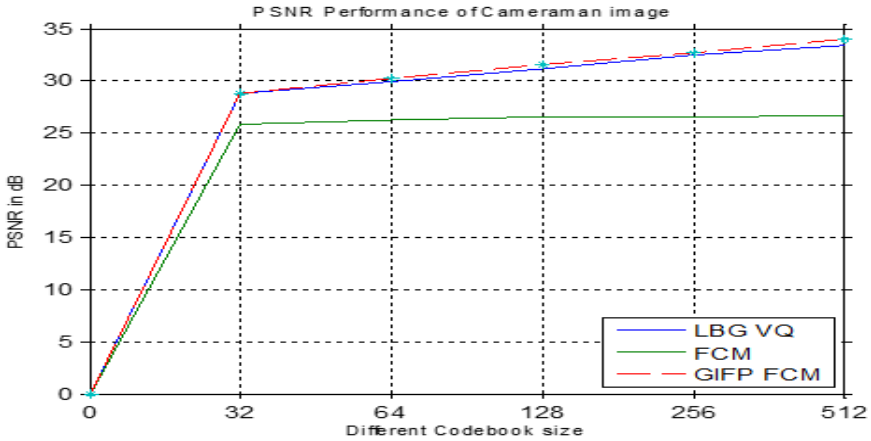

Fig. 3.15 Graphical representation of PSNR performance for cameraman image using LBG_VQ, FCM, GIFP-FCM algorithms

TABLE 3.8 PSNR PERFORMANCE OF BRAIN IMAGE FOR VARIOUS ALGORITHMS, VECTOR SIZE= [4 4]

ALGORITHMS	Codebook size K=32	Codebook size K=64	Codebook size K=128	Codebook size K=256	Codebook size K=512
1.**LBG_VQ**	26.79	27.84	29.27	30.42	31.68
2.**DWT_MSVQ** Stage1: Vec Size=[4 4],k=4 Stage1: Vec Size=[4 4],k=4	Level 1 Psnr=29.00 Mse=81.71	Level 2 Psnr=21.39 Mse=471.74	Level 3 Psnr=18.32 Mse=956.98		
3.**FCM**	26.32	26.47	26.52	26.54	26.60
4.**GIFP_FCM** (α=0.99,M=1.2 sed)	**29.33**	**29.73**	**30.97**	**32.12**	**33.26**

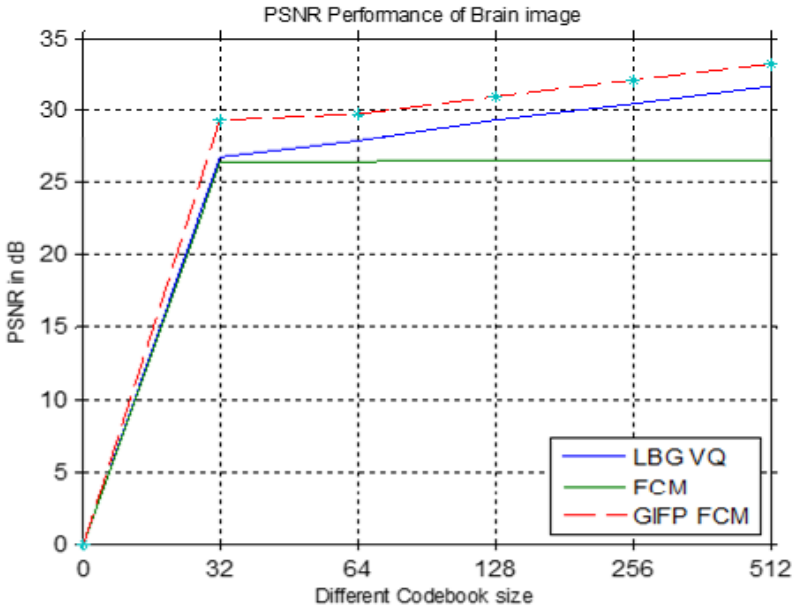

Fig. 3.16 Graphical representation of PSNR performance for Brain image using LBG_VQ, FCM, GIFP-FCM algorithms

TABLE 3.9 PSNR PERFORMANCE OF CHEST IMAGE FOR VARIOUS ALGORITHMS, VECTOR SIZE= [4 4]

ALGORITHMS	Codebook size K=32	Codebook size K=64	Codebook size K=128	Codebook size K=256	Codebook size K=512
1.**LBG_VQ**	30.26	31.37	32.61	33.04	34.01
2.**DWT_MSVQ** Stage1: Vec Size=[4 4],k=4 Stage1: Vec Size=[4 4],k=4	Level 1 Psnr=31.00 Mse=149.40	Level 2 Psnr=29.15 Mse=241.81	Level 3 Psnr=26.32 Mse=479.42		
3.**FCM**	30..6	31.12	31.33	31.90	32.05
4.**GIFP_FCM** (α=0.99, M=1.2 used)	**31.33**	**32.72**	**34.04**	**35.11**	**36.46**

Table 3.6, 3.7, 3.8, 3.9 represents the PSNR comparison among different algorithms i.e. LBG_VQ, DWT_MSVQ, FCM and the proposed GIFP-FCM based image compression algorithm. The proposed method shows better results in term of Peak Signal to Noise Ratio for the medical images taken with all the codebook sizes.

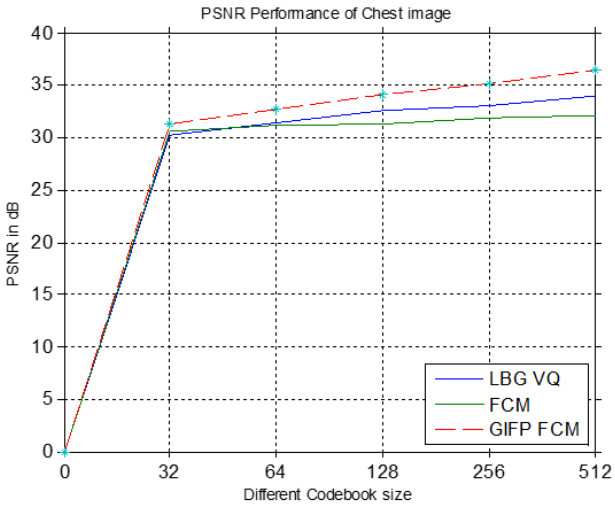

Fig. 3.17 Graphical representation of PSNR performance for chest image using LBG_VQ, FCM, GIFP-FCM algorithms

IV. CONCLUSIONS

This paper mainly concentrates on the concept of image compression combining different vector quantization algorithms like LBG, DWT-VQ, DWT-MSVQ, FCM and GIFP-FCM. These algorithms are tested on several type of gray scale medical and benchmark images. The simulation results show predictable improved execution for the GIFP-FCM vector quantization calculation. This strategy does not rely upon codebook introduction, simple to actualize, quick intermingling and does not require substantial calculation. When compressed with different calculations, the proposed GIFP-FCM technique can accomplish high CR with adequate reconstructed image quality regarding benchmark performance indices like PSNR and normal distortion and mean square qualities. Along these lines this GIFP-FCM based vector quantization can be utilized for image compression process with high reliability and robustness. The image compression work can be used for compressing the images using different Neuro wavelet models that uses Radial basis function neural network for training the indices generated after vector quantization process to increase the quality of compressed images. Also, we can use other different transform techniques like curvelet transform for decomposition of images into scales. Self-Organizing Feature mapping algorithm can be used in vector quantization for shortening of encoding and decoding time.

REFERENCES

1. A. Mohsin, A. Brifcani at el. "Image Compression Analysis Using Multistage Vector Quantization Based on Discrete Wavelet Transform," IEEE International Conference Methods and Models in Computer Science (ICM2CS), New-Delhi, pp (46-53), Dec 13-14 2010.
2. P. Bhattacharya, A. Mitra, A. Chatterjee, "Vector Quantization based Image Compression using Generalized Improved Fuzzy Clustering," IEEE International Conference on Control, Instrumentation, Energy & Communication (CIEC), pp (662-666) Feb 2014.
3. Tejas S. Patel at el. "Image Compression Using DWT and Vector Quantization," International Journal of Innovative Research in Computer and Communication Engineering Vol. 4, Issue 3, ISSN 2320-9801, IJIRCCE, May 2013.
4. Vinit D. Rout, S. Dholay, "Analyzing Image compression with efficient transforms and Multi Stage Vector Quantization using Radial Basis Function Neural Network," IEEE International Conference on Engineering & Technology (ICETECH), 20th March 2015.
5. L. Zhu, F. Chung and S. Wang, "Generalized Fuzzy C-Means Clustering Algorithm with Improved Fuzzy Partitions," IEEE Transactions on Systems, Man and Cybernetics-Part Cybernetics Vol.39, No.3, June 2009.
6. T. Phanprasit, "Compression of Medical Image using Vector Quantization," IEEE Biomedical International Conference (BMEiCON) 6th October 2013.
7. V. Singh, N. Rajpal, K. Murthy, "An Algorithm Approach for Efficient Image Compression using Neuro-Wavelet Model and Fuzzy Vector Quantization Technique," International Journal on Computer Science and Engineering (IJCSE) Vol.02, No.07, 2010.
8. G. Tsekouras, M. Antonios at el., "Image Compression Based on a Novel Fuzzy Learning Vector Quantization Algorithm,"
9. G. Tsekouras, "A fuzzy vector quantization approach to image compression," Applied Mathematics and Computation, Vol.167, Issue 1, pp (539-560) August 2005.

AUTHORS PROFILE

Gayatri Mohanta received B.Sc. degree in Electronics and Telecommunication from B. J. B. Autonomous College, Bhubaneswar, Odisha, India in 2014, the M. Sc. Degree from B. J. B. Autonomous College, Bhubaneswar, Odisha, India in 2016 and the M. Tech degree in Signal Processing from College of Engineering and Technology, Bhubaneswar in 2017. Her current research interests include image compression techniques, microwave filters design and antenna design.

Harish Chandra Mohanta received the M.Tech degree in Electronics and Communication Engineering from IIT Kharagpur in 2015. He is currently an Assistant Professor with the School of Engineering and Technology, Centurion University. His current research interests include image processing, development of electronic devices, embedded and antennas.