inverse wavelet transform is taken to decoded coefficients and the decompressed image will be obtained and performance parameters were determined to analyze the algorithm efficiency.

II. BACKGROUND

With the advances in imaging technology, diagnostic imaging has become an indispensable tool in medicine today. X-ray angiography (XRA), magnetic resonance angiography (MRA), magnetic resonance imaging (MRI), computed tomography (CT), and other imaging modalities are heavily used in clinical practice. Such images provide complementary information about the patient. While increased size and volume in medical images required the automation of the diagnosis process, the latest advances in computer technology and reduced costs have made it possible to develop such systems.

Brain tumor detection on medical images forms an essential step in solving several practical applications such as diagnosis of the tumors and registration of patient images obtained at different times. Segmentation algorithms form the essence of medical image applications such as radiological diagnostic systems, multimodal image registration, creating Anatomical atlases, visualization, and computer-aided surgery

Tumor segmentation algorithms are the key components of automated radiological diagnostic systems. Segmentation methods vary depending on the imaging modality, application domain, method being automatic or semi-automatic, and other specific factors. There is no single segmentation method that can extract vasculature from every medical image modality. While some methods employ pure intensity-based pattern recognition techniques such as thresholding followed by connected component analysis, some other methods apply explicit tumor models to extract the tumor contours. Depending on the image quality and the general image artifacts such as noise, some segmentation methods may require image preprocessing prior to the segmentation algorithm. On the other hand, some methods apply post-processing to overcome the problems arising from over segmentation.

Medical image segmentation algorithms and techniques can be divided into six main categories, pattern recognition techniques, model-based approaches, tracking-based approaches, artificial intelligence-based approaches, neural network-based approaches, and miscellaneous tube-like object detection approaches.

Pattern recognition techniques are further divided into seven categories, multi-scale approaches, skeleton-based approaches, region growing approaches, ridge-based approaches, differential geometry-based approaches, matching filters approaches, and mathematical morphology schemes.

Clustering analysis plays an important role in scientific research and commercial application. This thesis provides a survey of current tumor segmentation methods using clustering approach and provides both early and recent literature related to tumor segmentation algorithms and techniques.

III. METHODOLOGIES

The MRI brain tumor image is considered for this region based image compression. The primary tumor region is selected using clustering model to segment it. The second roi will be extracted through mapping the primary region on input image. Primary region will be compressed by using multilevel wavelet transform and Huffman Coding. The image (secondary region) to be compressed is transformed into frequency domain using lifting wavelet transform. In wavelet transform the images are divided into odd and even components and finally the image is divided into four levels of frequency components. The four frequency components are LL, LH, HL, HH, and then the image is encoded using SPECK coding for both roi and non roi part. Then the bit streams are obtained. The bitstreams which are generated by huffman coder is decoded to reconstruct the primary roi. The obtained bitstreams are decoded using SPECK decoding to reconstruct the secondary roi. Finally inverse wavelet transform is taken to decoded coefficients and the decompressed image will be obtained and performance parameters were determined to analyze the algorithm efficiency.

A. *Spatial Fuzzy Clustering Model*

Fuzzy clustering plays an important role in solving problems in the areas of pattern recognition and fuzzy model identification. A variety of fuzzy clustering methods have been proposed and most of them are based upon distance criteria. One widely used algorithm is the fuzzy c-means (FCM) algorithm. It uses reciprocal distance to compute fuzzy weights. A more efficient algorithm is the new FCFM. It computes the cluster center using Gaussian weights, uses large initial prototypes, and adds processes of eliminating, clustering and merging. In the following sections we discuss and compare the FCM algorithm and FCFM algorithm.

Spatial Fuzzy C Means method incorporates spatial information, and the membership weighting of each cluster is altered after the cluster distribution in the neighborhood is considered. The first pass is the same as that in standard FCM to calculate the membership function in the spectral domain. In the second pass, the membership information of each pixel is mapped to the spatial domain and the spatial function is computed from that. The FCM iteration proceeds with the new membership that is incorporated with the spatial function. The iteration is stopped when the maximum difference between cluster centers or membership functions at two successive iterations is less than a least threshold value.

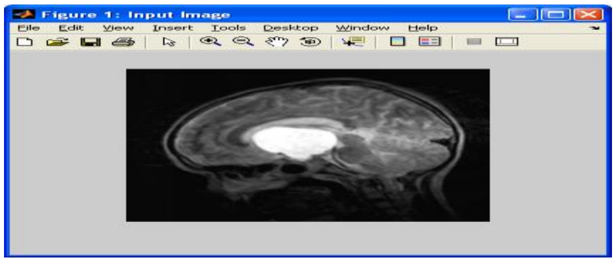


Fig 3.2.2: Restored Image without Noise

The fuzzy c-means (FCM) algorithm was introduced by J. C. Bezdek [2]. The idea of FCM is using the weights that minimize the total weighted mean-square error:

J(wqk, z(k)) = Σ (k=1,K) Σ (k=1,K) (wqk)|| x(q)-z(k)||2 Σ (k=1,K) (wqk) = 1

wqk = (1/(Dqk)2)1/(p-1) / Σ (k=1,K) (1/(Dqk)2)1/(p-1) , p > 1

B. *Image Segmentation*

Segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as super pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain visual characteristics.

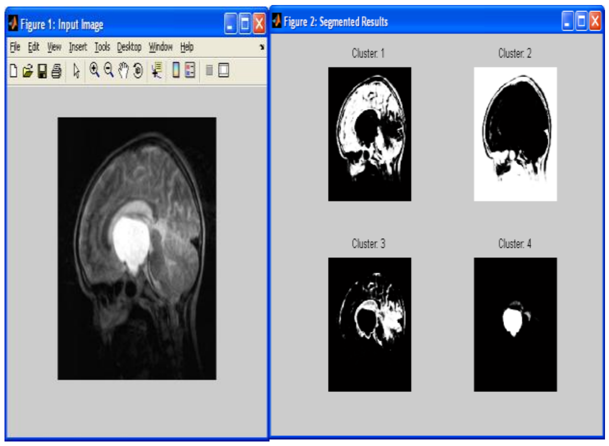


Figure 1: Input Image
File Edit View Insert Tools Desktop Window Help
Figure 2: Segmented Results
Cluster: 1
Cluster: 2
Cluster: 3
Cluster: 4

Fig 3.2.1: Segmentation of an Image Using Spatial Fuzzy C Means

The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image (see edge detection). Each of the pixels in a region is similar with respect to some characteristic or computed property, such as color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic(s) when applied to a stack of images, typical in medical imaging, the resulting contours after image segmentation can be used to create 3D reconstructions with the help of interpolation algorithms like Marching cubes.

C. *Morphological Process*

Morphological image processing is a collection of non-linear operations related to the shape or morphology of features in an image. Morphological operations rely only on the relative ordering of pixel values, not on their numerical values, and therefore are especially suited to the processing of binary images. Morphological operations can also be applied to grey scale images such that their light transfer functions are unknown and therefore their absolute pixel values are of no or minor interest.

Morphological techniques probe an image with a small shape or template called a structuring element. The structuring element is positioned at all possible locations in the image and it is compared with the correspondingneighbourhood of pixels. Some operations test whether the element "fits" within the neighbourhood, while others test whether it "hits" or intersects the neighbourhood:

A morphological operation on a binary image creates a new binary image in which the pixel has a non-zero value only if the test is successful at that location in the input image.

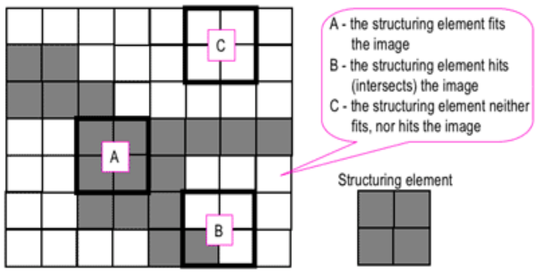

C
A
B
A - the structuring element fits the image
B - the structuring element hits (intersects) the image
C - the structuring element neither fits, nor hits the image
Structuring element

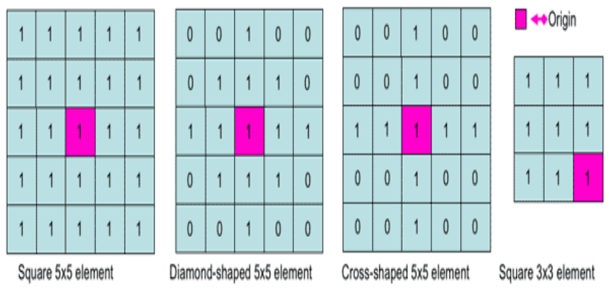
Fig 3.2.3: Probing of an Image with a Structuring Element

TheStructuring elementis a small binary image, i.e. a small matrix of pixels, each with a value of zero or one:

The matrix dimensions specify the *size* of the structuring element.

The pattern of ones and zeros specifies the shape of the structuring element.

An origin of the structuring element is usually one of its pixels, although generally the origin can be outside the structuring element.



Examples of Simple Structuring Elements

D. *Wavelet Transform*

Over the past several years, the wavelet transform has gained widespread acceptance in signal processing in general and in image compression research in particular. In applications such as still image compression, discrete wavelets transform (DWT) based schemes have outperformed other coding schemes like the ones based on DCT. Since there is no need to divide the input image into non-overlapping 2-D blocks and its basis functions have variable length, wavelet-coding schemes at higher compression ratios avoid blocking artifacts. Because of their inherent multi -resolution nature, wavelet-coding schemes are especially suitable for applications where scalability and tolerable degradation are important. Recently the JPEG committee has released its new image coding standard, JPEG-2000, which has been based upon DWT.

Basically we use Wavelet Transform (WT) to analyze non-stationary signals, i.e., signals whose frequency response varies in time, as Fourier Transform (FT) is not suitable for such signals.

To overcome the limitation of FT, Short Time Fourier Transform (STFT) was proposed. There is only a minor difference between STFT and FT. In STFT, the signal is divided into small segments, where these segments (portions) of the signal can be assumed to be stationary. For this purpose, a window function "w" is chosen. The width of this window in time must be equal to the segment of the signal where it is still be considered stationary. By STFT, one can get time-frequency response of a signal simultaneously, which can't be obtained by FT. The short time Fourier transform for a real continuous signal is defined as:

$$X(f,t)= \int_{-\infty}^{\infty}[x(t)w\ (t-\tau)^{*}]e^{-2j\pi j\tau}dt \text{ -------(3.1)}$$

Where the length of the window is (t-τ) in time such that we can shift the window by changing value of t and by varying the value τ we get different frequency response of the signal segments.

The Heisenberg uncertainty principle explains the problem with STFT. This principle states that one cannot know the exact time-frequency representation of a signal, i.e., one cannot know what spectral components exist at what instances of times. What one can know are the time intervals in which certain band of frequencies exists and is called resolution problem. This problem has to do with the width of the window function that is used, known as the support of the window. If the window function is narrow, then it is known as compactly supported. The narrower we make the window, the better the time resolution, and better the assumption of the signal to be stationary, but poorer the frequency resolution:

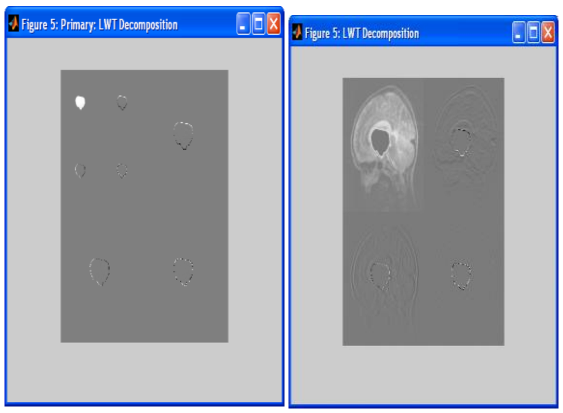


Figure 5: Primary: LWT Decomposition
Figure 5: LWT Decomposition

Fig 3.2.4: Wavelet Decomposition

Narrow window ===> good time resolution, poor frequency resolution

Wide window ===> good frequency resolution, poor time resolution

The wavelet transform (WT) has been developed as an alternate approach to STFT to overcome the resolution problem. The wavelet analysis is done such that the signal is multiplied with the wavelet function, similar to the window function in the STFT, and the transform is computed separately for different segments of the time-domain signal at different frequencies. This approach is called Multi-resolution Analysis (MRA) [4], as it analyzes the signal at different frequencies giving different resolutions.

For now, assume the loose requirement that Ψ(t) has compact temporal and spectral support (limited by the uncertainty principle of course), upon which set of basis functions can be defined.

The basis set of wavelets is generated from the mother or basic wavelet is defined as:

$$\Psi a,b(t) = \frac{1}{\sqrt{a}}\psi\left(\frac{t-b}{a}\right) \text{; a, b } \in \Re \text{ and a>0 ---- (3.2)}$$

The variable 'a' (inverse of frequency) reflects the scale (width) of a particular basis function such that its large value gives low frequencies and small valuegives high frequencies. The variable 'b' specifies its translation along x-axis in time. The term 1/ $\sqrt{a}$ is used for normalization.

a. *Forward Transform*

Step1: Column wise processing to get H and L

H = (Co-Ce) (1)

L = (Ce- _H/2_) (2)

Where Co and Ce is the odd column and even column wise pixel values

Step 2: Row wise processing to get LL,LH,HL and HH, Separate odd and even rows of H and L,

b. Reverse Lifting Scheme In IWT

Inverse Integer wavelet transform is formed by Reverse lifting scheme. Procedure is similar to the forward lifting scheme.

Hodd – odd row of H, Lodd- odd row of L, Heven- even row of H, Leven - even row of L

LH = Lodd-Leven

LL = Leven – (LH / 2)

HL = Hodd – Heven

HH = Heven – (HL / 2)

IV. CODING TECHNIQUES

A. Huffman Coding

Huffman coding is an entropy encoding algorithm used for lossless data compression. The term refers to the use of a variable-length code table for encoding a source symbol (such as a character in a file) where the variable-length code table has been derived in a particular way based on the estimated probability of occurrence for each possible value of the source symbol.

Huffman coding uses a source symbols to unique strings of bits will produce a smaller specific method for choosing the representation for each symbol, resulting in a prefix code (sometimes called "prefix-free codes", that is, the bit string representing some particular symbol is never a prefix of the bit string representing any other symbol) that expresses the most common source symbols using shorter strings of bits than are used for less common source symbols. Huffman was able to design the most efficient compression method of this type: no other mapping of individual average output size when the actual symbol frequencies agree with those used to create the code. A method was later found to design a Huffman code in linear time if input probabilities (also known as weights) are sorted.

For a set of symbols with a uniform probability distribution and a number of members which is a power of two, Huffman coding is equivalent to simple binary block encoding, e.g., ASCII coding. Huffman coding is such a widespread method for creating prefix codes that the term "Huffman code" is widely used as a synonym for "prefix code" even when such a code is not produced by Huffman's algorithm.

Although Huffman's original algorithm is optimal for a symbol-by-symbol coding (i.e. a stream of unrelated symbols) with a known input probability distribution, it is not optimal when the symbol-by-symbol restriction is dropped, or when the probability mass functions are unknown, not identically distributed, or not independent (e.g., "cat" is more common than "cta"). Other methods such as arithmetic coding and LZW coding often have better compression capability: both of these methods can combine an arbitrary number of symbols for more efficient coding, and generally adapt to the actual input statistics, the latter of which is useful when input probabilities are not precisely known or vary significantly within the stream.

B. Basic Technique

A source generates 4 different symbols {a1,a2,a3,a4} with probability {0.4;0.34;0.2;0.04}. A binary tree is generated from left to right taking the two least probable symbols and putting them together to form another equivalent symbol having a probability that equals the sum of the two symbols. The process is repeated until there is just one symbol. The tree can then be read backwards, from right to left, assigning different bits to different branches.

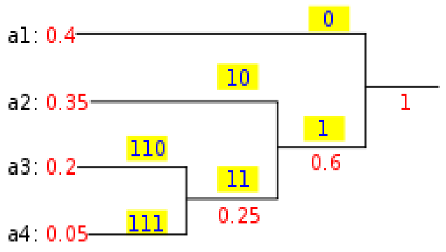


Fig 4.3: Huffman Code Flow

The final Huffman code is:

SYMBOL	CODE
A1	0
A2	10
A3	110
A4	111

Fig 4.3: Huffman Code

The standard way to represent a signal made of 4 symbols is by using 2 bits/symbol, but the entropy of the source is 1.73 bits/symbol. If this Huffman code is used to represent the signal, then the average length is lowered to 1.84 bits/symbol; it is still far from the theoretical limit because the probabilities of the symbols are different from negative powers of two.

The technique works by creating a binary tree of nodes. These can be stored in a regular array, the size of which depends on the number of symbols, n. A node can be either a leaf node or an internal node. Initially, all nodes are leaf nodes, which contain the symbol itself, the weight (frequency of appearance) of the symbol and optionally, a link to a parent node which makes it easy to read the code (in reverse) starting from a leaf node. Internal nodes contain symbol weight, links to two child nodes and the optional link to a parent node. As a common convention, bit '0' represents following the left child and bit '1' represents following the right child. A finished tree has up to n leaf nodes and n − 1 internal nodes. A Huffman tree that omits unused symbols produces the most optimal code lengths.

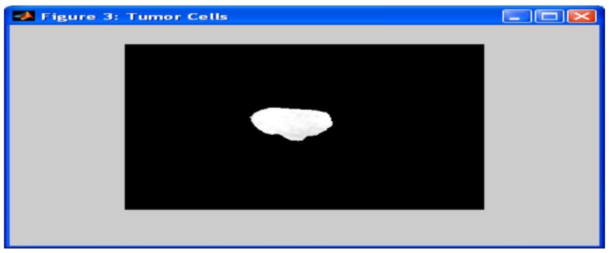


Fig 4.2: Selection of Image using Huffman Coding

The process essentially begins with the leaf nodes containing the probabilities of the symbol they represent, then a new node whose children are the 2 nodes with smallest probability is created, such that the new node's probability is equal to the sum of the children's probability. With the previous 2 nodes merged into one node (thus not considering them anymore), and with the new node being now considered, the procedure is repeated until only one node remains, the Huffman tree.

C. Data Decoding

The process of decompression is simply a matter of translating the stream of prefix codes to individual byte values; usually by traversing the Huffman tree node by node as each bit is read from the input stream (reaching a leaf node necessarily terminates the search for that particular byte value). Before this can take place, however, the Huffman tree must be somehow reconstructed. In the simplest case, where character frequencies are fairly predictable, the tree can be preconstructed (and even statistically adjusted on each compression cycle) and thus reused every time, at the expense of at least some measure of compression efficiency. Otherwise, the information to reconstruct the tree must be sent a priori. A naive approach might be to prepend the frequency count of each character to the compression stream. Unfortunately, the overhead in such a case could amount to several kilobytes, so this method has little practical use. If the data is compressed using canonical encoding, the compression model can be precisely reconstructed with just B2B bits of information (where B is the number of bits per symbol). Another method is to simply prepend the Huffman tree, bit by bit, to the output stream. For example, assuming that the value of 0 represents a parent node and 1 a leaf node, whenever the latter is encountered the tree building routine simply reads the next 8 bits to determine the character value of that particular leaf. The process continues recursively until the last leaf node is reached; at that point, the Huffman tree will thus be faithfully reconstructed. The overhead using such a method ranges from roughly 2 to 320 bytes (assuming an 8-

bit alphabet). Many other techniques are possible as well. In any case, since the compressed data can include unused "trailing bits" the decompresses must be able to determine when to stop producing output. This can be accomplished by either transmitting the length of the decompressed data along with the compression model or by defining a special code symbol to signify the end of input (the latter method can adversely affect code length optimality, however).

D. Coding Algorithm

- Obtain the probability of each symbol.
- Arrange the probabilities in descending order.
- The first source reduction is obtained by combining the least two probabilities to form a compound symbol.
- This process is repeated until the reduced source with two symbols is reached.
- Each of the reduced sources is coded starting from the smallest source and working back to the original source.

Huffman procedure creates an optimal code for asset of symbols and probabilities subject to the constrain that the symbol decoded one at a time. After the code has been created coding and / or error free decoding is accomplished. The code itself instantaneous uniquely decodable block code. It is called a block code because each source symbol is mapped into a fixed sequence of code symbols. It is instantaneous because each code word is a string of code symbols that can be decoded without referring succeeding symbols. It is uniquely decodable because any string of code symbols can be decoded in only one way.

E. Speck Coding

The SPECK coder is a powerful image compression algorithm that produces an embedded bit stream from which the best reconstructed images in the mean square error sense can be extracted at various bit rates. The perceptual image quality, however, is not guaranteed to be optimal since the coder is not designed to explicitly consider the human visual system (HVS) characteristics. Extensive HVS research has shown that there are three perceptually significant activity regions in an image: smooth, edge, and textured or detailed regions. By incorporating the differing sensitivity of the HVS to these regions in image compression schemes such as SPECK, the perceptual quality of the images can be improved at all bit rates.

Previous work to improve the visual quality of embedded coders has applied just noticeable distortion thresholds for uniform noise in different sub bands to weight the transform coefficients but no distinction made between coefficients belonging to different activity regions inside a sub band. In this paper, the differing activity regions are used to assign perceptual weights to the transform coefficients prior to SPIHT encoding.

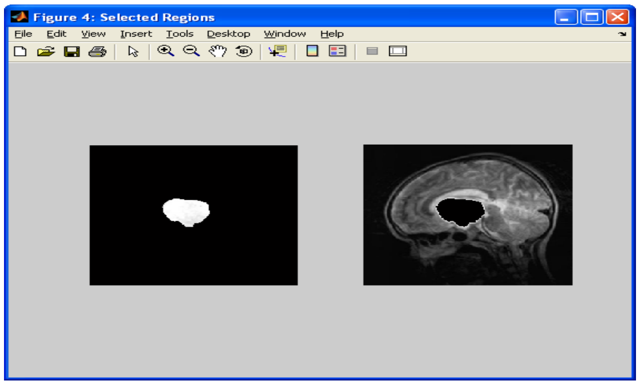


Figure 4: Selected Regions
File Edit View Insert Tools Desktop Window Help

Fig 4.6: Primary and Secondary Regions Segmentation

a. Progressive Image Transmission

After converting the image pixels into wavelet coefficient SPECK is applied. We assume, the original image is defined by a set of pixel values $p_{i,j}$, where (i, j) the pixel coordinates. The wavelet transform is actually done to the array given by,

$$c(i,j) = DWT\{p(i,j)\}$$

Where c (i, j) is the wavelet coefficients.

In SPIHT, initially, the decoder sets the reconstruction vector $\widehat{c}$ to zero and updates its components according to the coded message. After receiving the value (approximate or exact) of some coefficients, the decoder can obtain a reconstructed image by taking inverse wavelet transform, called as “progressive transmission”.

$$\hat{p}(i, j) = IDWT\{c(i, j)\}$$

A major objective in a progressive transmission scheme is to select the most important information-which yields the largest distortion reduction-to be transmitted first. For this selection, we use the mean squared-error (MSE) distortion measure

$$D_{MSE}(p-\hat{p}) = \frac{1}{N}\|p-\hat{p}\ \|^2 = \frac{1}{N}\sum_i\sum_j\ (p_{i,j}-\hat{p}_{i,j})^2$$

Where Nis the number of image pixels. $p_{i,j}$ is the Original pixel value and $\hat{p}_{i,j}$ is the reconstructed pixel value. Furthermore, we use the fact that the

Euclidean norm is invariant to the unitary transformation

$$D_{MSE}(p-\hat{p}) = D_{MSE}(c-\hat{c}) = \frac{1}{N}\sum_i\sum_j\ (c_{i,j}-\hat{c}_{i,j})^2$$

From the above the equation, it is clear that if the exact value of the transform coefficient $c_{i,j}$ is sent to the decoder, then the MSE decreases by $\left|c_{i,j}\right|^2 / N$. This means that the coefficients with larger magnitude should be transmitted first because they have a larger content of information. This is the progressive transmission method. Extending this approach, we can see that the information in the value of $\left|c_{i,j}\right|$ can also be ranked according to its binary representation, and the most significant bits should be transmitted first. This idea is used, for example, in the bit-plane method for progressive transmission. Following, we present a progressive transmission scheme that incorporates these two concepts: ordering the coefficients by magnitude and transmitting the most significant bits first. To simplify the exposition, we first assume that the ordering information is explicitly transmitted to the decoder. Later, we show a much more efficient method to code the ordering information.

b. Transmission of the Coefficient Values

Let us assume that the coefficients are ordered according to the minimum number of bits required for its magnitude binary representation, that is, ordered according to a one-to-one mapping, Now, let us assume that, besides the ordering information, the decoder also receives the numbers μ_n corresponding to the number of coefficients such that $2^n \le \left|c_{i,j}\right| < 2^{n+1}$. In the example of Fig. 1 we have $\mu_5 = 2$, $\mu_4 = 2$, $\mu_3 = 4$, etc. Since the transformation DWT is unitary, all bits in a row have the same content of information, and the most effective order for progressive transmission is to sequentially send the bits in each row, as indicated by the arrows in Fig. 1. Note that, because the coefficients are in decreasing order of magnitude, the Ieading“0”bits and the first “1” of any column do not need to be transmitted, since they can be inferred from μ_n and the ordering.

The progressive transmission method outlined above can be implemented with the following algorithm to be used by the encoder.

Algorithm I

1) Output n = $\lfloor \log_2(\max_{(i,j)}\{\left|c_{i,j}\right|\})\rfloor$ to the decoder

2) Output μ_n, followed by the pixel coordinates $\eta(k)$ and sign of each of the μ_n coefficients such that $2^n \le \left|c_{i,j}\right| < 2^{n+1}$ (sorting pass).

3) Output the nth most significant bit of all the coefficients with $\left|c_{i,j}\right| \ge 2^{n+1}$ (i.e., those that had their coordinates transmitted in previous sorting passes), in the same order used to send the coordinates (refinement pass);

4) Decrement n by one, and go to Step 2).

c. Set Partitioning Sorting Algorithm

One of the main features of the proposed coding method is that the ordering data is not explicitly transmitted. Instead, it is based on the fact that the execution path of any algorithm is defined by the results of the comparisons on its branching points. So, if the encoder and decoder have the same sorting algorithm, then the decoder can duplicate the encoder's execution path if it receives the results of the magnitude comparisons, and the ordering information can be recovered from the execution path.

One important fact used in the design of the sorting algorithm is that we do not need to sort all coefficients. Actually, we need an algorithm that simply selects the coefficients such that $2^n \le |c_{i,j}| < 2^{n+1}$, with n decremented in each pass. Given n, if $|c_{i,j}| \ge 2^{n+1}$ then we say that a coefficient is significant; otherwise it is called insignificant.

The sorting algorithm divides the set of pixels into partitioning subsets T_m and performs the magnitude test

$$\underset{(i,j)\in T_m}{Max}\{|c_{i,j}|\} \ge 2^n$$

If the decoder receives a "no'' to that answer (the subsetis insignificant), then it knows that all coefficients in T_m, are insignificant. If the answer is "yes" (the subset is significant), then a certain rule shared by the encoder and the decoder is used to partition T_m, into new subsets $T_{m,l}$ and the significance test is then applied to the new subsets. This set division process continues until the magnitude test is done to all single coordinate significant subsets in order to identify each significant coefficient.

To reduce the number of magnitude comparisons (message bits) we define a set partitioning rule that uses an expected ordering in the hierarchy defined by the subband pyramid. The objective is to create new partitions such that subsets expected to be insignificant contain a large number of elements, and subsets expected to be significant contain only one element.

To make clear the relationship between magnitude comparisons and message bits, we use the function

$$S_n(T) = \quad 1; \quad \underset{(i,j)\in T_m}{Max}\{|c_{i,j}|\} \ge 2^n \; 0;$$

otherwise.

To indicate the significance of a set of coordinates T .To simplify the notation of single pixel sets, we write $S_n(\{(i,j)\})$ as $S_n(i,j)$.

d. Spatial Orientation Trees

Normally, most of an image's energy is concentrated in the low frequency components. Consequently, the variance decreases as we move from the highest to the lowest levels of the subband pyramid. Furthermore, it has been observed that there is a spatial self-similarity between subbands, and the coefficients are expected to be better magnitude-ordered if we move downward in the pyramid following the same spatial orientation. For instance, large low-activity areas are expected to be identified in the highest levels of the pyramid, and they are replicated in the lower levels at the same spatial locations.

A tree structure, called spatial orientation tree, naturally defines the spatial relationship on the hierarchical pyramid. Fig.3.6 shows how our spatial orientation tree is defined in a pyramid constructed with recursive four-subband splitting. Each node of the tree corresponds to a pixel and is identified by the pixel coordinate. Its direct descendants (offspring) correspond to the pixels of the same spatial orientation in the next finer level of the pyramid. The tree is defined in such a way that each node has either no offspring (the leaves) or four offspring, which always form a group of 2 x 2adjacent pixels.

For instance, except at the highest and lowest pyramid levels, we have

O(i,j) = {(2i,2j),(2i,2j+1),(2i+1,2j),(2i+1,2j+1)}

The following sets of coordinates are used to present the new coding method:

O (i,j): set of coordinates of all offspring of node (i, j);
D (i, j) : set of coordinates of all descendants of the node
H: set of coordinates of all spatial orientation tree roots (nodes in the highest pyramid level);
L(i, j) = D(i, j) - O(i, j) .

We use parts of the spatial orientation trees as the partitioning subsets in the sorting algorithm. The set partition rules are simply the following.

I)The initial partition is formed with the sets ((i, j)) and D(i,j), for all (i,j)∈ H.
2) If D(i,j) is significant, then it is partitioned into L(i,j) plus the four single-element sets with (k , I)∈ O(i,j) .
3)If L(i,j) is significant, then it is partitioned into the four sets D(k,I), with (k,I) ∈ O(i,j)

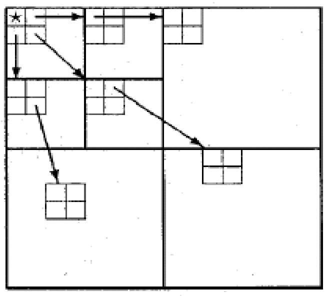


Fig 4.5.4: Spatial-Orientation Tree

e. Coding Algorithm

Since the order in which the subsets are tested for significance is important, in a practical implementation the significance information is stored in three ordered lists, called list of insignificant sets (LIS), list of insignificant pixels (LIP), and list of significant pixels (LSP). In all lists each entry is identified by a coordinate (i,j) , which in the LIP and LSP represents individual pixels, and in the LIS represents either the set D(i,j) or L(i, j) . To differentiate between them, we say that a LIS entry is of type A if it represents D (i, j), and of type B if it represents L (i, j).

During the sorting pass (see Algorithm I), the pixels in the LIP-which were insignificant in the previous pass-are tested, and those that become significant are moved to the LSP. Similarly, sets are sequentially evaluated following the LIS order, and when a set is found to be significant it is removed from the list and partitioned. The new subsets with more than one element are added back to the LIS, while the single-coordinate sets are added to the end of the LIP or the LSP, depending whether they are insignificant or significant, respectively. The LSP contains the coordinates of the pixels that are visited in the refinement pass.

Algorithm II

```
1) Initialization: output n = $\lfloor \log_2(\max_{(i,j)}\{|c_{i,j}|\}) \rfloor$ ; set the LSP as an empty list, and add the coordinates (i, j) ∈Hto the LIP, and only those with descendants also to the LIS, as type Aentries.
2) Sorting Pass:
2.1) for each entry (i,j) in the LIP do:
2.1.1) output $S_n(i, j)$ ;
2.1.2) if $S_n(i, j)$ = 1 then move (i, j )to the LSP and output the sign of $c_{i,j}$;
2.2) for each entry (i, j) in the LIS do:
2.2.1) if the entry is of type A then
•   output $S_n(D(i,j))$ ;
•   if $S_n(D(z, 1))$ = 1 then
•   for each (k , I )∈ O(i,j ) do:
•   output $S_n(k, l)$;
•   if $S_n(k, I)$ = 1 then add ( k ,l ) to the LSB and sign of $c_{i,j}$
•   ifS,(k, 1) = 0 then add ( k ,l ) to the end of LIP.
•   if l(i, j ) # 0 then move (i, j ) to the end of the LIS, as an entry of type B, and go to Step 2.2.2); otherwise, removeentry(i, j ) from the LIS;
2.2.2) if the entry is of type B then
•   output Sn(L(2, J ) ) ;
•   if $S_n(L(I, j))$ = 1 then
•   add each ( k , I ) ∈ O(z, j ) to the end of the LIS as an entry of type A;
•   remove(i, j) from the LIS.
3) Refinement Pass: for each entry (i, j )in the LSP, except those included in the last sorting pass (i.e., with same n), output the nth most significant bit of $|c_{i,j}|$ ;
4) Quantization-Step Update: decrement n by 1 and go to Step 2.
```

One important characteristic of the algorithm is that the entries added to the end of the LIS in Step 2.2) are evaluated before that same sorting pass ends. So, when we say “for each entry in the LIS” we also mean those that are being added to its end. With Algorithm II, the rate can be precisely controlled because the transmitted information is formed of single bits. The encoder can also use the property, to estimate the progressive distortion reduction and stop at a desired distortion value.

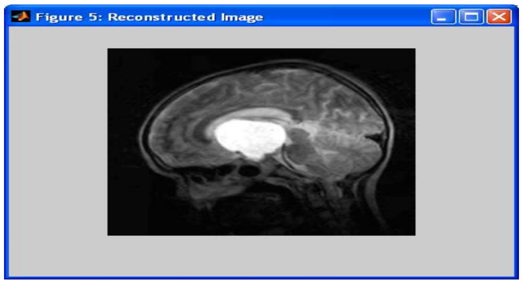
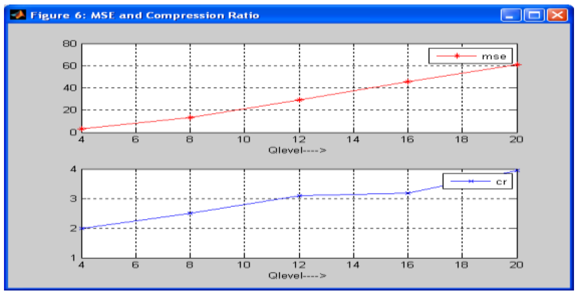
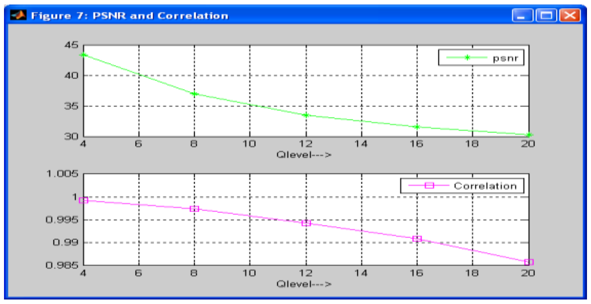


Fig 4.5.4: Reconstructed Image at the Destination End

F. Performance Analysis

The Compression Ratio, Error, Correlation, Encode and Decode Time are evaluated at the Destination end to achieve the Best result for the Reconstructed Image.In order to provide objective assessments of segmentation performance, there is a need for an objective 3D ground truth with associated MR images that exhibit the same major segmentation challenges as that of common, realistic scans of a tumor patient. A database of real brain tumor MR images, along with their segmentations, may provide the means to measure the performance of an algorithm by comparing the results against the variability of the expert raters' judgments.





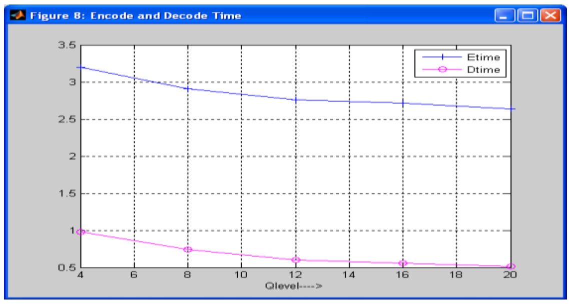


Fig 4.6: Performance Analysis

V. Conclusion

This project presented that medical image compression using lossy and lossless coding for analysis of compression effect based on wavelet based set partitioning embedded block coding and entropy Huffman coding. Wavelet based speck coding was used to compress image by considering significant coefficients based on priority. Lossless Huffman coding provided better encoding performance with multi-level wavelet transform. It represented an image interms of detailed coefficients in all directions. Here, the parametric such as compression ratio, peak signal to noise ratio, correlation coefficient and elapsed time will be evaluated from these used coders. The simulated results shows that, Combination of Speck and Huffman coding were provided better compression ratio and considerable error with different bitrate values.

References

[1] Maglogiannis, I., Kormentzas, G.: 'Wavelet-based compression with ROI coding support for mobile access to DICOM images over heterogeneous radio networks', Trans. Inf. Technol. Biomed., 2009, 13, (4), pp. 448–466

[2] Doukas, C., Maglogiannis, I.: 'Region of interest coding techniques for medical image compression', IEEE Eng. Med. Biol. Mag., 2007, 26, (4), pp. 29–34

[3] Zhang, Q., Xiaio, X.: 'Extracting regions of interest in biomedical images'. Int. Seminar on Future BioMedical Information Engineering, 2008, pp. 3–6

[4] Yu, Y., Wang, B.: 'Saliency based compressive sampling for image signals', IEEE Signal Process. Lett., 2010, 17, (11), pp. 973–976

[5] Calderbank, A.R., Daubechies, I., Sweldens, W., Yeo, B.L.: 'Wavelet transforms that map integers to integers', Appl. Comput. Harmon. Anal., 1998, 4, pp. 332–369

[6] Ardizzone.E, Pirrone.R, and Orazio.O.G, "Fuzzy C-Means Segmentation on Brain MR Slices Corrupted by RF-Inhomogeneity," In Proc. The 7th international workshop on Fuzzy Logic and Applications: Applications of Fuzzy Sets Theory, WILF '07, Springer- Verlag, pp: 378-384, 2007.