# High Speed ECG Image Compression Using Modified SPIHT

Kazi Rafiqul Islam[1], Md. Anwarul Abedin[1], Masuma Akter[1], and Rupam Deb[2]

***Abstract*—In this paper a new technique of Electrocardiogram (ECG) image compression using Modified Set Partitioning In Hierarchical Tree (M-SPIHT) algorithm have been presented. Image quality is compared objectively using mean square error (MSE) and peak signal to noise ratio (PSNR) along with visual appearance. As a result, an efficient high speed ECG image compression at low bit-rate is achieved. The simulation results have shown that this new method is suitable for mobile communication due to achieving good visual quality of reconstructed ECG image at low bit rate compare to original SPIHT and other compression system.**

***Index Terms*—ECG, MSE, PSNR, SPIHT, M-SPIHT, Wavelet.**

## I. Introduction

The main purpose of compression is to represent an ECG image with the smallest possible number of bits. It can assist the transmission and processing of image. Among many medical signal sources, the compression of electrocardiogram (ECG) is in great demand [1]-[3]. Many types of ECG recordings generate a vast amount of data. These include up to 48 hour Holter recordings, telemetry recordings, continuous ECG performed in intensive care units and stress test ECG. With the growing use of these ECG signals to detect and diagnose heart disorders, ECG compression becomes mandatory to efficiently store and retrieve this data from medical database. Other practical importance includes transmitting real time ECG over the mobile communication network and storing patient data in a medical smart card. In this paper we treated ECG signal as image recorded on an ECG paper. ECG paper is traditionally divided into 1mm squares [4]. Vertically, ten blocks usually correspond to 1 mV, and on the horizontal axis, the paper speed is usually 25mm/s, which makes one block 0.04s (or 40ms). We also have "big blocks" which are 5mm on their side. Knowing the paper speed, its easy to work out heart rate. If the number of big block is 1, the rate is 300, if it is 2, the rate is 150 and so on. Rates in between these numbers are easy to interpolate. In recent years, wavelet based embedded image coder is quite attractive in modern applications. Wavelet transform, bit plane coding and other techniques make embedded image coders practically, which not only provide efficient compression
performance, distortion scalability, resolution scalability, but also other attractive features such as region of interest and random access.

The efficiency of a wavelet-based compression scheme relies on the efficiency of specifying to the decoder which coefficients to quantize before which others, and of the corresponding bit allocation. Said and Pearlman [7] developed an algorithm, called set partitioning in hierarchical trees (SPIHT) based on the same basic concepts. It was more effective in transmission of significance information to the decoder. Both the schemes relied on partial magnitude ordering of the wavelet coefficients, followed by progressive refinement, and produced embedded bit streams. The transmission of ordering information is achieved by a subset partitioning approach that is duplicated at the decoder. The refinement is based on ordered bit plane transmission of the magnitudes of the coefficients previously ascertained as significant.

In this work, a low bit rate image coder of modified SPIHT algorithm without arithmetic coder has been demonstrated for High speed ECG compression. The modifications of the *SPIHT* compressor have been presented combining the sorting and refinement phase. With the elimination of List of Singnificant Pixels (LSP) and List of insignificant pixels (LIP) lists, the memory requirement has been reduced tremendously. This paper is organized as follows. Section II discusses the original SPIHT algorithm and its modification. Section III shows the experimental verification of the M-SPIHT algorithm. Section IV concludes this paper.

## II. Coding Methodology

### A. Wavelet Transform

Wavelet transform (WT) represents an image as a sum of Wavelet functions (wavelets) with different locations and scales [5]. Any decomposition of an image into wavelets involves a pair of waveforms: one to represent the high frequencies corresponding to the detailed parts of an image (Wavelet function) and one for the low frequencies or smooth parts of an image (scaling function). The result of wavelet transform is a set of wavelet coefficients, which measure the contribution of the wavelets at these locations and scales. While embedded zero tree like algorithms are applied, a wavelet transform is performed on the image. This result is a multiscale representation. The transform reduces the correlation between neighboring pixels. The energy of the original image is concentrated in the lowest frequency band of the transformed image. Additionally, self similarities between different scales which result from the recursive application of the wavelet transform step to the low frequency band can be observed. Consequently, based upon these facts good compression performance can be achieved if

Manuscript received January 17; revised May 20, 2011.

Kazi Rafiqul Islam is with the Department of EEE, Dhaka University of Engineering & Technology.

Gazipur, Bangladesh Rupam De, Department of CSE Dhaka University of Engineering. & Technology, Gazipur, Bangladesh (E-mail: rafiqul_duet@yahoo.com).

those coefficients are first transmitted which represent most of the image energy.

### B. SPIHT Coder

The SPIHT algorithm is very efficient in transmission of ordering information, essentially involves a scalar quantization operation. The essence of the set portioning is to first classify the elemental coding units based on their magnitude and then to quantize them in a successive refinement framework. The elemental coding unites are scalar wavelet coefficients.

For completeness, we briefly introduce the SPIHT coding algorithm in this section. More details of SPIHT can be referred to [7]. Fig. 1 illustrates the wavelet tree structure of a typical three-scale pyramidal decomposition of an image. The image is generated by three stages of two-dimensional (2-D) DWT. The notations $LL_i$ , $HL_i$ ,$LH_i$ and $HH_i$ denote the output channels from the *i*th stage. The parent-offspring dependency for tree structures is also demonstrated. Each node has either four offspring or no offspring. The nodes has no offspring are located on $Layer_1$ (i.e., the bands $HL_1$, LH1 and $HH_1$) and some of them are located on the highest layer (one of them indicated by the " * " in Fig. 1).

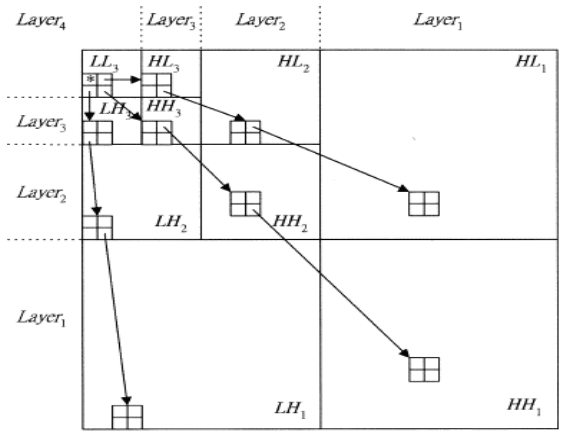



Fig. 1 Examples of the SPIHT tree structure in a typical three-scale pyramidal decomposition of an image. The arrows are oriented from the parent node to its offspring.

The main idea is based on partitioning of sets, which consists of coefficients or representatives of whole subtrees. The coefficients of a wavelet transformed image are classified in three sets:

1. The list LIP of **i**nsignificant pixels which contains the coordinates of those coefficients which are insignificant with respect to the current threshold th.

2. The list of **s**ignificant pixels (LSP) which contains the coordinates of those coefficients which are significant with respect to threshold, and

3. The list of insignificant **s**ets (LIS) which contains the coordinates of the roots of insignificant subtrees. During the compression procedure, the sets of coefficients in LIS are refined and if coefficients become significant they are moved from LIP to LSP.

We call a node (i.e., transformed coefficient) C (i, j) at a coarse scale a parent. All nodes at the next finer scale with the same spatial location, and of similar orientation are called children, this set denoted O (i, j). More precisely O (i, j) = {C (2i, 2j), C (2i, 2j+1), C (2i+1, 2j), C (2i+1, 2j+1)} except the nodes at the highest layer ($Layer_4$ in Fig. 1) and the lowest layer ($Layer_1$ in Fig. 1). All nodes at all finer scales with the same spatial location, and of similar orientation are called descendant, denoted D (i, j). A set L(i, j) is defined as L(i,j)=D(i,j)-O(i, j), and the set H is the group of coordinates of all the tree roots (nodes in the highest layer).We also refer to a node or a set as significant if the result of the significant test by (1)

$$S_n(X(i,j))=\begin{cases}1 & if \ \max_{C(k,l)\in X(i,j)}\{|C(k,l)|\}\geq 2^n \\ 0 & otherwise\end{cases} \quad (1)$$

where X(i, j) represents C(i, j) , D(i, j) , or L(i, j). This equation indicates that if the coefficient with maximum magnitude in a set is significant, then the significant test results in 1.

If D (i, j) is significant, then it is partitioned into O (i, j) and L (i, j) if L (i, j) exists. If not, it is a zerotree of type A. If L (i, j) is significant, then it is partitioned into {D (2i, 2j), D (2i, 2j+1), D (2i+1, 2j), D (2i+1, 2j+1)} except the coordinates in the highest layer. Otherwise, it is a zerotree of type B. If we encounter a zerotree, we code such tree as a zerotree symbol, and avoid coding all its nodes. The nodes are scanned by the order of importance. It is performed so that no child is scanned before its parent. Therefore, one starts scanning the nodes C(i, j) for (i, j) $\in$ H and the sets D(i, j) for (i, j) $\in$ H . The result of significant test for a node or for a set is coded. In addition, for each node C (i, j), if it is significant, its sign bit is also coded.

The process begins with setting LSP as an empty list, adding the coordinates (i, j) $\in$ H to the LIP, adding those with descendants to the LIS as type A entries, and outputting the maximal value of n. The value can be obtained by using (2)

$$n=\left\lfloor \log_2 \max_{(i,j)}\{|C(i,j)|\}\right\rfloor \quad (2)$$

Then the following two passes, the sorting pass and refinement pass, are used for every n value. In the sorting pass, we scan each C (i, j) in the LIP and each D (i, j) or L (i, j) in the LIS, extract significant nodes, and put them into the end of the LSP. In the refinement pass, however, another bit of precision is added to the magnitudes of nodes in the LSP. We decrease by one, i.e., cut the threshold in half, and use these two passes for each in the order of the sorting pass first until the bit budget is exhausted.

The algorithm addressed above does not consider the statistical dependence between adjacent nodes and between adjacent sets. To increase the coding efficiency, the significance values of 2×2 adjacent nodes (the nodes with the same parent) were grouped and coded as a single symbol by the arithmetic coding algorithm. In general, the decoder duplicates the execution path of the encoder as it was also the case in Shapiro's algorithm. To ensure this behavior, the coder sends the result of a binary decision to the decoder before a branch is taken in the algorithm. Thus, all decisions of the decoder are based on the received bits. The name of the algorithm is composed of the words *set* and *partitioning*. The sets O (i, j), D (i, j) and L (i, j) were already mentioned. Another advantage of *SPIHT* in comparison to EZW algorithm is that the complete outcome of the encoder is binary. Therefore, the compression rate can be achieved

exactly and arithmetic encoders for binary alphabets can be applied to further increase the compression efficiency.

### C. Modified SPIHT Coder

In SPIHT, the usage of three temporary lists is a powerful way to improve the codec's efficiency. But they are quite memory consuming. It is a major drawback for SPIHT algorithm. In addition, during coding we often insert or delete the elements in the lists. These frequent operations will greatly increase the coding time with the expanding of the lists. In order to realize the implementation of SPIHT algorithm, a successful low-memory solution must be provided. In this algorithm, the sorting and refinement phase are combined as one scan pass. Below we present two new concepts [10], called *number of error bits* and *absolute zerotree*, to modify the original SPIHT algorithm.

*1) Number of Error Bits:* During SPIHT coding, only the most significant bits in *transform coefficients* are outputted for later decoding. Thus, the last several bits (i.e., the least significant bits) will be omitted. We call this omitted part of bits as *truncating error*.

We can define *number of error bits* (denoted by $\mu_e$) before encoding to indicate the number of bits that would be omitted finally. In practical implementation, as soon as C (i,j) is found as significant coefficient or insignificant coefficient, its first ( $n+1-\mu_e$) bits are immediately outputted. And its coordinates are no longer stored into the LSP or the LIP. These two lists can be thrown away.

*2) Absolute Zerotree:* After wavelet decomposition, most of the significant coefficients are concentrated in low-pass subbands. And the magnitudes of *transform coefficients* decrease rapidly with the decline of the pyramid level. Through extensive experiments, we found that the coefficients in many sets are so small that these trees will always be zerotrees before the expected compression ratio is reached. In SPIHT coding, the coordinates of these zerotree roots are stored in the LIS and will never be removed. It results in the rapid expansion of the LIS.

The introduction of *absolute zerotree* is a simple solution to this problem. We have defined to indicate the number of truncating error bits. For a zerotree, if the magnitudes of all its descendants are lower than $2^{\mu e}$, it becomes an *absolute zerotree* and will never be significant in the last *scan passes*. Their coordinates need not be stored in the LIS. Obviously, the length of the LIS is shortened. And because we do not scan among *absolute zerotrees* any longer, the coding time is also greatly reduced.

The new notation Zero indicates whether a set is an absolutes zerotree. As shown in (3), the value 0 means that the sets is an absolute zerotree and vice versa.

$$Zero(X(i,j))=\begin{cases}0 & if \ \max_{C(k,l)\in X(i,j)}\{|C(k,l)|\}\geq 2^{\mu_e}\\ 1 & otherwise\end{cases} \quad (3)$$

where X(i, j) represents C(i, j) , D(i, j) , or L(i, j).

The *Number of error bits* defined before encoding will indicate the number of bits that will be omitted finally. During implementation, when a wavelet coefficient will be found as significant or insignificant, its last error bits will be omitted and rest of the bits will be outputted. In addition, the co-ordinates of the coefficient will not be stored in LSP and LIP for further processing. Therefore, M-SPIHT is the low memory solution of SPIHT algorithm by eliminating the temporary list LSP and LIP. The entries inside LIS are split up into two parts. One contains the pixel coordinates (i,j) for which all the elements of set D(i,j) are insignificant, and the other contains the pixel coordinates (i,j) for which all the elements of the set L(i,j) are insignificant. These are referred to as *Type A* and *Type B*.

In MSPIHT, the sorting and refinement pass are combined together to reduce the execution time. Before coding, MSPIHT initializes the number of error bits. The initial value of *n* is set to be the index of the MSB of the largest magnitude of wavelet coefficients. The coding process starts with adding those with descendants to the LIS as type A, for each $(i,j)\in H$, the last error bits of C(i,j) will be omitted and rest of the bits of that pixel and its sign bit will be outputted. In the first coding pass, the magnitude, and sign information of coefficients are coded.

For simplicity, we assume, herein, that each coefficient has four offspring. The adjacent coefficients with the same parent are grouped and coded together to remove redundancy. The same scheme is used to code the significance of adjacent sets. MSPIHT begins coding the coefficients in the highest layer (i.e. the $LL_3$ in fig. 2). For each coordinate (i,j), it codes the individual coefficient C(i,j) and the descendant set D(i,j). Whenever coding C (i,j), it checks whether C(i,j) has been significant from previous coding pass.

After coding C(i,j), MSPIHT checks the significance of D(i,j) .if D(i,j) is insignificant in the previous coding pass, it checks the results of $S_n(D(i,j))$. If $S_n(D(i,j))=1$, the set D(i,j) will be partitioned into O(i,j) and L(i,j), if L(i,j) exits, else the entry (i,j) will be removed from the LIS list. For each entry of O (i,j), it will take the significant test with respect to the threshold value of each coding pass. If Zero (k,l)=1 according to equation(4), it will output a 1bit and its sign bit. If not, it will consider the whole branch as an absolute zerotree and will finish the whole branch. If L(i,j) exits and is significant, then (i,j) will be moved to the end of LIS as an entry of Type B, else the entry (i,j) will be removed from LIS list.

If L(i,j) is insignificant, the entry is of Type B. Hereafter, it will check whether Sn(L(i,j)) is significant, if yes, then partition into set D(k,l). For each entry (k,l), it will out put Zero(D(k,l)). If Zero (D (k,l))=1, then the entry (k,l) will be added as an entry of type A. otherwise the entry (i,j) will be removed from the list LIS. After that, for next pass, the value of n is decreased by one. The process continues until the bit budget is exhausted.

## III. Experimental Results and Image Quality Evaluation

The image quality can be evaluated objectively and subjectively. A standard objective measure of image quality is reconstructed error. Two of the error metrics used to compare the various image compression techniques are the Mean Square Error (MSE) and the Peak Signal to Noise Ratio (PSNR).The MSE is the cumulative squared error between the compressed and the original image, whereas PSNR is a measure of the peak error. The mathematical formulas for the two are

$$MSE = \frac{1}{MN}\sum_{Y=1}^{M}\sum_{X=1}^{N}\left[I(X,Y)-I'(X,Y)\right]^2 \quad (4)$$

$$PSNR = 10\log 10\left(\frac{255^2}{MSE}\right) \quad dB \quad (5)$$

In this set of experiment, the SPIHT is chosen as the zero-tree based image coder and the M-SPIHT applied without arithmetic coding. This algorithm shows excellent performance without arithmetic coding among the class of zerotree-based encoders. The descriptions of visual quality in the compressed ECG image are evaluated based on at the different bit rate obtained M-SPIHT algorithm. With M-SPIHT algorithm, SPIHT algorithm has been taken to compare the reconstructed image quality in terms of mean square error (MSE) and peak signal-to-noise ratio (PSNR).

The biorthogonal 4.4 filter bank and 512×512 gray scale ECG image with 8 bpp are used for the experiment. The Nine level decompositions are constructed by a symmetric extension at the image edge. It is important to observe that the bit rates are not entropy estimated calculated from the actual size of the compressed file. By using progressive transmission ability, the sets of distortions are obtained from the same file. The decoder reads the same file bytes of the file calculated the inverse subband transform and then compared the recovered image with the original. The distortion is measured by the PSNR as in (5).

In this work, A ECG images have been used to measure the performance of reconstructed image in terms of PSNR.The performance of proposed MSPIHT without arithmetic coding is compared to other popular algorithms such as arithmetic coded SPIHT. Table I compares the performance of proposed M-SPIHT algorithm versus the original SPIHT algorithms. While the results are at extremely low bit rates, the PSNR difference is almost equivalent at all the other rates.

TABLE I COMPARISON IN PSNR, MSE, ENCODING TIME AND DECODING TIME WITH BIT RATE DISTORTION PERFORMANCES OF ORIGINAL SPIHT, M-SPIHT IMAGE CODING OF TEST ECG IMAGES

| Bit Rate (bpp) | | 0.05 | 0.075 | 0.1 | 0.25 |
|---|---|---|---|---|---|
| MSE | MSPIHT | 155.184 | 95.025 | 49.156 | 13.324 |
| | SPIHT | 152.331 | 91.712 | 46.340 | 6.043 |
| PSNR | MSPIHT | 26.256 | 28.386 | 31.249 | 36.918 |
| | SPIHT | 26.337 | 28.541 | 31.505 | 40.352 |
| Encoding time(sec) | MSPIHT | 0.594 | 2.641 | 2.797 | 9.016 |
| | SPIHT | 48.281 | 48.281 | 48.281 | 48.281 |
| Decoding time(sec) | MSPIHT | 0.1719 | 1.2344 | 1.1094 | 5.3125 |
| | SPIHT | 38.594 | 38.594 | 38.594 | 38.594 |

We have presented the coding results from 0.05 bits/pixel up to 0.25 bits/pixel for these ECG images with varying the bit rate. This range of bit rate is quite low for image coding, but the reconstructed image quality is also impressive in terms of PSNR and MSE. PSNR comparisons are made with SPIHT and M-SPIHT to show the effectiveness of the M-SPIHT algorithm. This algorithm surpassed the binary encoded version of SPIHT for these ECG images. And the encoding and decoding time significantly reduces then SPIHT at low bit rate. The plots for ECG images have shown the efficiency of M-SPIHT algorithm.

In Fig. 2 is plotted the PSNR versus bit rate obtained of image for M-SPIHT, entropy-coded SPIHT using arithmetic code. The PSNR of M-SPIHT without arithmetic coding is about similar to the other algorithms at low bit rate. Also, in Fig. 3 is plotted the MSE versus bit rate for ECG image that is same at low bit rate compared to the SPIHT algorithms.

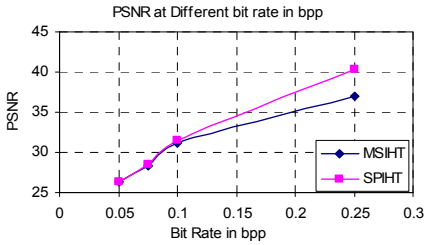



Fig. 2 Coding result for the ECG image

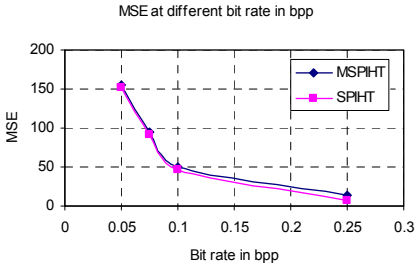



Fig. 3 Coding result for the ECG image

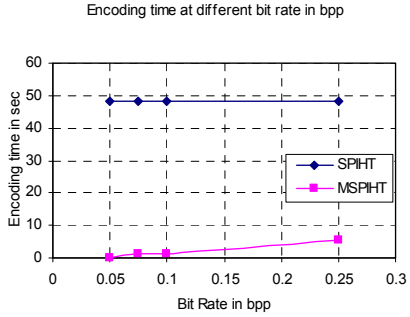



Fig: 4 Encoding time with respect to bit rate for the ECG image.

Fig.4 and Fig.5 shows the encoding and decoding time of ECG image using the modified SPIHT algorithm is almost zero at 0.05 bpp to 0.1 bpp. Also the visual quality of the reconstructed ECG image at very low bit rate is quite impressive for M-SPIHT algorithm. We should note that the experimental image is 512×512 in size, which is quite large among all usual image/video formats, particularly for mobile communication. For smaller images, the coding efficiency will be further increased and more exciting requirements will be met. By various analyses, we can say that the ECG image compression by M-SPIHT algorithm is quite good then other ECG image compression technique.

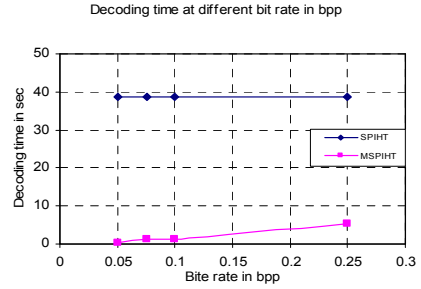
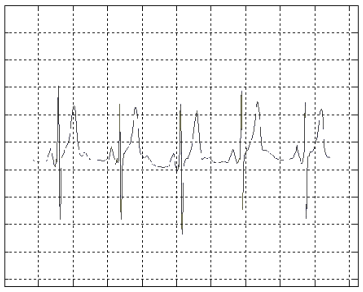
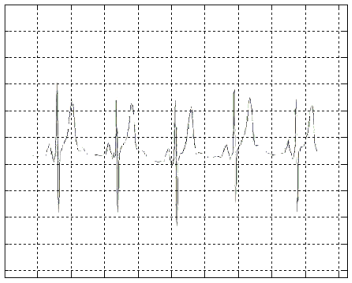


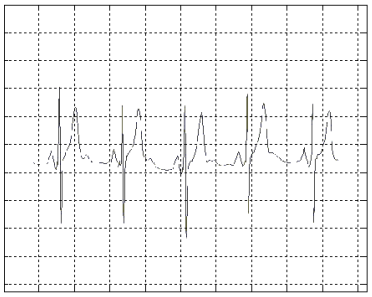

Fig. 5 Decoding time with respect to bit rate for the ECG image.



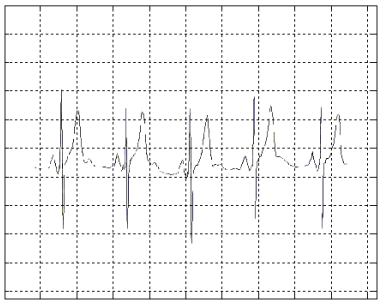
(a)



(b)



(C)



(D)

Fig. 6 Images obtained using modified SPIHT a) Original Image (b)Bit Rate = 0.05 bpp, PSNR= 26.25 dB (c)Bit Rate=0.075 bpp, PSNR =28.38dB (d)Bit Rate=0.1, PSNR =31.24dB.

## IV. Conclusion

An efficient high speed ECG image compression at low bit rate has been achieved by M-SPIHT algorithm. At 0.05, 0.075, 0.1, 0.25 bit rate the encoding time is 0.594, 2.641, 2.797, 9.016 second and decoding time is 0.1719, 1.2344, 1.1094, and 5.3125 second respectively. The experimental results have shown remarkable improvement of this embedded image coder that provides very efficient ECG image compression at very low bit rate. Very good visual performance has been achieved compare to arithmetic coded SPIHT and other. In this sense, M-SPIHT provides an essential algorithm for ECG image compression and makes these zerotree-based encoders more competitive than other wavelet ones. This algorithm can be used for other medical signal to coding-decoding for mobile communication because of having good PSNR at very low bit rate.

## References


[1] S. M. S Jalaleddin, C. G. Hutchens, R. D. Strattan and W. A. Coberly, "ECG data compression techniques – A unified approach," IEEE Trans. Biomed. Eng., vol. 37, no. 4, pp. 329-343, April 1990.

[2] G. Nave and A. Cohen, "ECG compression using longterm-prediction," IEEE Trans. Biomed. Eng., vol. 39, no. 4, pp. 330-337, April 1992.

[3] M. L. Hilton, "Wavelet and wavelet packet compression of electrocardiogram," IEEE Trans. Biomed. Eng., vol. 44, no. 5, pp.394-402, May 1997.

[4] Leo Schamroth, an Introduction to Electrocardiography, Blackwell Scientific, 7th Ed., 1990.

[5] Daubechies, Ten Lectures on Wavelets. Philadelphia, PA: SIAM, 1992.

[6] M.Shapiro, "Embedded image coding using zerotrees of wavelet coefficients," IEEE Transaction on Signal Processing, vol. 41, pp 3445-3462, December 1993.

[7] Amir Said, and William A. Pearlman, "A New, Fast, and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees" IEEE transactions on circuits and systems for video technology, vol. 6, issue 3, pp. 243-250, June 1996

[8] Z.Xiong, C.Herley, K.Ramchandran, and M. T.Orchard, "Space-frequency quantization for a space-varying wavelet packet coder," IEEE International Conference on Image Processing, pp. 614-617, Washington DC, October 1995.

[9] J.Knipe, X.Li, and B.Han, "An improved lattice vector quantization scheme for wavelet compression," IEEE Transaction on Signal Processing, vol. 46, issue 1, pp. 239-243, January 1998.

[10] Yong Sun, Hui Zhang, Guangshu Hu, "*Real-time implementation of a new low-memory SPIHT image coding algorithm using DSP chip*", IEEE Transactions on Image Processing, vol 11, Issue 9, pp 1112 1116, Sept. 2002 .

[11] Tareq Aziz, M. A. Haque, "*Wavelet based ECG image compression"* Signal Processing Systems Design and Implementation, 2005. IEEE Workshop on. pp *731*-734.

[12] M.B.I. Reaz, M. Akter, and F. Mohd-Yasin, "Image Compression System for Mobile Communication: Advancement in the Recent Years", presented at Journal of Circuits, Systems, and Computers, 2006, pp.777-815.

[13] M. B. I. Reaz, M. Akter, F. Mohd-Yasin, " An Efficient Image Coder using M*o*dified SPIHT for Wireless Communication" WSEAS Transactions on Communications. Issue 6, Volume 5, June 2006,pp 1146-1150.

[14] M.B.I. Reaz, M. Akter, F. Mohd-Yasin, F. Choong, M.I. Ibrahimy and M.Kamada, "Design and Implementation of a Modified SPIHT Algorithm for image compression."Proceeding on Intelligent Systems and Control – 2007, track-592-020.

[15] Shiv Kumar, Member IACSIT, Vijay K Chaudhari, Member IACSIT, Dr. R. K. Singh, Dr. Dinesh Varshney, "A New Algorithm for Voice Signal Compression (VSC) and Analysis Suitable for Limited Storage Devices Using MatLab." IJCEE 2009 Vol.1 (5): 656-665 ISSN: 1793-8163.