Effective PCG Signals Compression Technique Using an Enhanced 1-D EZW

Nabil Boukhennoufa[1], Khier Benmahammed[2] and Redha Benzid[1]

[1]*Department of Electronics, Batna University, Algeria*
[2]*Department of Electronics, Setif University, Algeria*
boukhennoufa.nabil@gmail.com, khierben@lycos.com, rbenzid@lycos.com

Abstract

This paper presents an Enhanced version of the Embedded Zerotree Wavelet (EEZW) dedicated to Phonocardiogram (PCG) compression. This algorithm is constituted of six steps: applying Discrete Wavelet Transform, uniform scalar quantizing, creating dominant and subordinate lists, generating preliminary symbol stream and finally using arithmetic coding. The compression algorithm has been tested and evaluated by its application on several databases. The adapted 1-D algorithm shows better performance than the original EZW, the modified Ouafi's EZW (MEZW) and the popular MP3 audio-specific format, regarding to the well-known trade-off rate/distortion. Also a low computational complexity of our method is obtained. It is due to the only one time subordinate pass calculated one time after the achievement of all dominant passes.

Keywords: *PCG Compression, Wavelet Transform, EZW, Compression ratio, PRD*

1. Introduction

PCG is the graphical representation of heart sounds and murmurs. The timing and pitch of a heart sound are of significant importance in a diagnosis of a heart condition [1].

The features of a PCG signal such as heart sounds, the number of components for each sound, their frequency content and their time interval, can be measured more accurately by recent digital signal processing techniques [2].

According to the advances of signal processing techniques, PCG can be a useful diagnostic tool, revealing information that the human ear cannot offer. By the fact, expert systems PCG-based can be made. A new tentative to materialize such systems can be found in [2, 3, 4].

Compared to ECG, PCG diagnosis is much easier by just placing the stethoscope against the skin. The current problem with many PCG systems is noises from sounds of breathes, contact of the stethoscope with the skin and other ambient noises, which may corrupt the heart signals. Consequently, the PCG would be much more useful for diagnosis in homecare system, if the noises were eliminated. The PCG is of major importance to achieve a basic diagnosis when high-cost techniques, like echocardiography, are not available [5, 6].

To record PCG signal waveform, a large amount of data is stored. Therefore, using the compression methods, one can reduce the space of the PCG signals data storage. The main goal of any compression technique is to attain maximum data reduction while preserving the significant signal morphology features upon reconstruction [7]. The nature of medical signals requires the implementation of lossy compression algorithms subjected to severe constraints [8]. Data compression methods can be divided into two groups: direct methods and transform methods [9]. Direct methods are performed by acting directly on the samples of the original waveforms in the time domain. The disadvantages of direct techniques are their sensitiveness to sampling rate, quantization levels, and high frequency interference. Usually, these techniques fail to achieve high data reduction along with the preservation of clinical information. However, the transform-based techniques including: Fourier Transform, Karhunen-Loeve Transform (KLT), Fast Walsh Transform, Discrete Cosine Transform (DCT) and Wavelet Transform, achieve higher compression ratio than direct algorithms [10]. Among them, the latter transform (which is a multi-scale representation) has been used successfully in a broad range of applications and signal compression in particular. Especially it has been applied to several problems in cardiology signals, including data compression, the analysis of ventricular late potentials, and the detection of characteristic points [11].

Several compression algorithms using wavelets have been proposed, such as EZW (Embedded Zero-trees Wavelet) [12, 13], the SPIHT (Set Partitioning in Hierarchical Trees) [9, 10, 14] and the SPECK (Set Partitioning Embedded Block) [10, 12]. These algorithms which rely on embedded coding create an embedded binary flow, a progressive data transmission that allows the signal to be reconstructed using various compression ratios [12]. Image coding (or compression), in first time, was an important application for the EZW method, it is based on the wavelet theory. Later, EZW was applied for (1-D) signals [15].

In this paper, we propose a PCG compression technique. It is based on a modified version of the embedded zero wavelet tree coding technique [13]. The proposed method consists of three stages: a reversible wavelet transform, a modified EZW data structure to order the coefficients and an arithmetic coding. The paper is organized as follows: Discrete Wavelet-Transform applied to 1-D signals is described in Section 2. We present in Section 3, the classic EZW developed to 1-D signals. In Section 4, the proposed method is detailed. EEZW algorithm is evaluated in Section 5 using several PCG records and the simulation (quantitative and qualitative) results are compared with recently reported powerful compression methods in the literature such as Shapiro's algorithm (EZW) [13], Ouafi et al method (MEZW) [12] and MP3 coder. Finally, in Section 6 conclusion is drawn.

2. Discrete Wavelet Transform

The idea behind the use of the transform techniques consists on transforming the signal from the time domain to the frequency domain. It provides multiresolution representation by using a set of analyzing functions that are dilations with a scale parameter "a" and translations by a factor "b" of a single function called the mother wave [16].

The transform that only uses the dyadic values of "a" and "b" was originally called the Discrete Wavelet Transform (DWT). The dilation function of the DWT can be represented, as a tree of low and high pass filters, with each step transforming the output of the low pass filter. Figure 1 shows the wavelet decomposition tree of the original signal V_0 [15, 16].

V_0 is successively decomposed into components of lower resolution, while the high frequency components are not analyzed any further. At each step of the DWT algorithm, there are two outputs: the scaling coefficients $V_{j+1}(m)$, and the wavelet coefficients $W_{j+1}(m)$ [16, 17]. These coefficients are given respectively, by:

$$V_{j+1}(m)=\sum_{i=1}^{2m} h(2m-i)V_j(m) \quad (1)$$

and

$$W_{j+1}(m)=\sum_{i=1}^{2m} g(2m-i)V_j(m) \quad (2)$$

Where *j* represents the scale number, *h* and *g* represent the low pass and high pass filters transfer functions respectively [17].

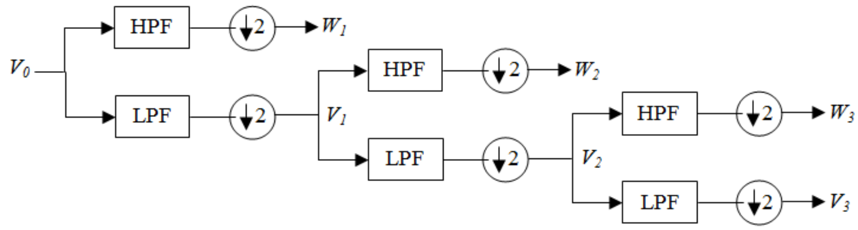


Figure 1. Three Level Pyramidal Decomposition by DWT

3. 1-D EZW Coding Algorithm

There are mainly three processors in a transform coder: (a) the wavelet transform, used to decorrelate the input samples; (b) the quantization processor that quantizes the transform coefficients where an information loss occurs; (c) the compression unit that produces the compressed values of quantized symbols as efficiently as possible. Finally, the transform coder (as shown in Figure 2) outputs a bit stream which represents the coded signal [15].

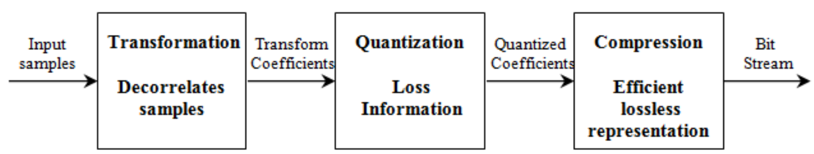


Figure 2. A Transform Coder

The parent-child dependencies of pyramid subband decomposition are shown in Figure 3 [15]. In this figure, the coefficient at LLH3 subband is the parent of two coefficients in the subband LH2, where L and H letters denote low and high, respectively. The filter outputs are collected in a vector in order to form the subband decomposition of the signal. The number of elements in LLH3 subband is the half of the number of elements in LH2 subband. Generally, it is organized as the lower subband coefficients V_3 (LLL3) to the higher subband coefficients W_1 (H1) in subband coding. This subband scale and wavelet coefficients are collected in a vector from the lowest subband to the highest subband coefficients as [15].

$$[V_3\ W_3\ W_2\ W_1] \tag{3}$$

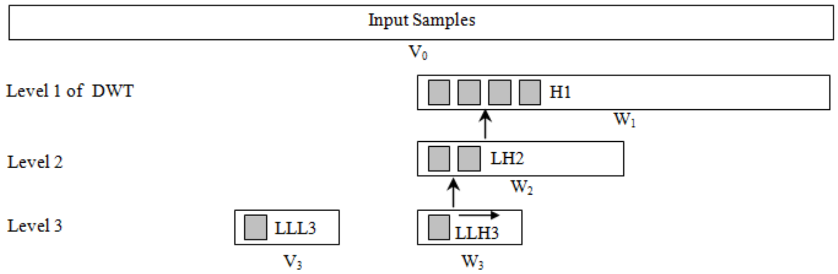


Figure 3. Parent–child Dependencies of Subbands for a 1-D Signal (decomposition depth is 3)

The embedded zerotree wavelet algorithm (EZW) encoder is specially designed to be used with WT, which was initially designed to operate on images (2-D signals) [13]. The bit stream of EZW is generated in order of importance, yielding a fully embedded code. The compressed data stream can have any bit rate desired [13].

The main steps of EZW algorithm are:

1. *Initialization: Set the threshold T to the smallest power of 2 that is greater than max(i) (*$|c_i|/2$*), where* c_i *are the wavelet coefficients.*
2. *Significance map coding (dominant pass): Scan all the coefficients in a predefined way and output a symbol. For each coefficient that has now become significant, put its magnitude on the subordinate list and remove it from the dominant list.*
3. *Refinement (subordinate pass):*
   - *Provide next lower significant bit on the magnitude of each coefficient on subordinate list.*
   - *Halve the quantizer cells to get the next finer quantizer.*
   - *If magnitude of coefficient is in upper half of old cell, provide "1".*
   - *If magnitude of coefficient is in lower half of old cell, provide "0".*
4. *Updating threshold: set T = T/2, and go to step 2 if more iterations are needed.*

In each iteration, the scan starts at the lowest frequency subband LLL3. For LLH3, LH2, H1, each parent in a subband is just scanned before its children at the next finer scale. A parent (*i*) at LLH3 must be scanned after the last descendants of the parent (*i*-1) in the same subband. All the coefficients are scanned in the order shown in Figure 3. This guarantees that when a node is visited, all its parents will already have been scanned. Each coefficient visited in the scan is classified as a zerotree root (*T*), an isolated zero (*Z*), positive significant (*P*), or negative significant (*N*)

4. Proposed Method

In this section, our enhanced EZW 1-D signal coding technique is discussed. The bit stream is generated by using the coder represented by the block diagram shown in Figure 4.

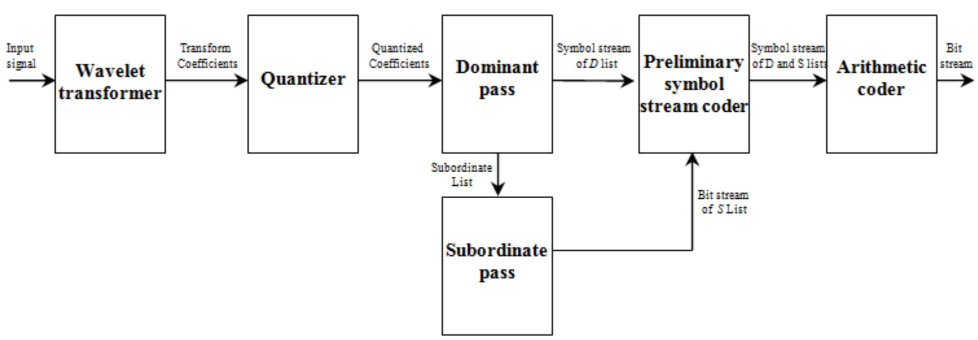


Figure 4. Block Diagram of the Proposed Coder

The proposed coding technique is divided in six parts. In the first stage, Wavelet coefficients are generated by using DWT, and then a uniform scalar quantizer is introduced. From quantized coefficients, two separate lists (Dominant and Subordinate lists) are outputted.

During a dominant pass, coefficients with coordinates on the dominant list, i.e., those that have not yet been found to be significant, are compared to the threshold T_i to determine their significance, and if significant, their sign.

In Shapiro's algorithm [13], a dominant pass is followed by a subordinate pass in which all coefficients on the subordinate list are scanned. The specifications of the magnitudes available to the decoder are refined to an additional bit of precision. A scanning of the coefficients is performed in such a way that no child node is scanned before its parent. The subordinate list contains the magnitudes of the coefficients found to be significant. The sequence of decreasing thresholds are $T_0\ T_1\ T_2\ ...\ T_{N-1}$ with $T_i=T_{i-1}/2$ and $|coefficients|<2T_0$.

To have a low computation complexity, only one subordinate pass is used after all dominant passes in our approach.

Finally, a preliminary symbol stream and arithmetic coders are used.

Using the quantized coefficients, bit stream is generated by following these five main steps:

Step 1: Initialization of the threshold, the dominant list and the subordinate list.

Step 2: Using dominant passes to create symbol stream of D list and insert significant elements in subordinate list.

Step 3: A bit stream of S list is generated after applying a low computation complexity using only one subordinate pass.

Step 4: Preliminary symbol stream is generated by using the diagram shown in figure 7 based on RLE method.

Step 5: An arithmetic coder must be used to achieve a high compression performance with this method.

4.1. Dominant Pass

Each coefficient and its descendants must be tested. If a coefficient is found significant, 'P' or 'N' symbols are inserted in the symbol stream. If a coefficient is found insignificant and all its descendants are also insignificant, the symbol '*T*' is outputted. Otherwise, if a descendant is judged significant the symbol '*Z*' is assigned.

All the significant coefficients are placed without their sign on the subordinate list. A coefficient is coded if its magnitude (in absolute value) is smaller than $2T_i$. This will prevent this coefficient from being coded again. In each iteration, the set of the symbols generated must contain all symbols. The algorithm of the dominant pass is presented in the following:

```
While (abs(coefficient)<2 Ti)
begin
    code scanned coefficient
    output the symbol
    if coefficient was coded as P or N then
   begin
       add abs(coefficient) to subordinate list
```

end

scan next coefficient

end

Let us consider the coefficients of vector test shown in Figure 5.

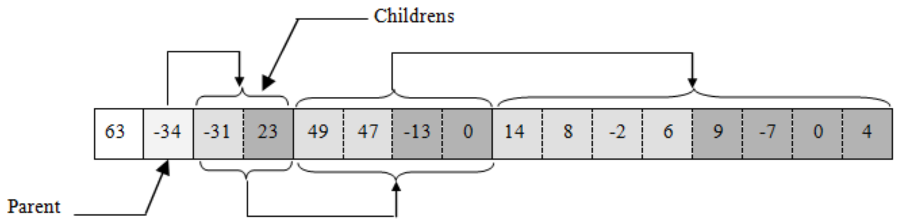


Figure 5. Example of Decomposition up to the Level 3 for 16 Cells

D_0 is the first symbol stream obtained for the first iteration, where T_0=32.

D_0: *P N Z T P P Z Z Z Z*

The algorithm is repeated on the signal residue by dividing the threshold T_i by two. For T_1=16, the sequence D_1 is obtained.

D_1: *N P T T T T*

This process is reiterated until the desired quality of the reconstructed signal is reached or until the number of transferable bits required is exceeded.

4.2. Subordinate Pass

Note that each subordinate pass in EZW method emits one bit for each coefficient which has been found significant in the previous significance passes. During the first subordinate pass using a threshold 32 (Figure 5), the uncertainty interval for the magnitudes of all the significant coefficients is the interval [32, 64]. The first subordinate pass will refine these magnitudes and identify them as being either in interval [32, 48], which will be encoded with the symbol “0”, or in the interval [48, 64], which will be encoded with the symbol “1”. Note that the decision limit is the magnitude 48. In the second subordinate pass (threshold 16) new coefficients will be inserted with the previous magnitudes in dominant pass. In this case, the three uncertainty intervals are [48, 64], [32, 48] and [16, 32] were used. If a coefficient in the subordinate list is found in lower half of an interval, the symbol “0” is outputted; else the symbol “1” is emitted if coefficient is in the upper half of the interval. The processing is continued between dominant and subordinate passes.

With this new approach, a sequence of the subordinate pass is created. It performs same results as Shapiro’s algorithm [13]. The bit stream of *S* list is produced by using the following algorithm:

For all elements on subordinate list do

begin

binary_element= binary form of element without MSB

Place binary value in vertical position *end*

Generate bit stream of S list (S={S_0, S_1, …}) by scanning bits in horizontal positions

This method is based on the Figure 6. In each iteration, all significant coefficients found are converted from decimal to binary values. For more illustration, we continue with the previous example:

Consequently $(63)_{10}$ is binary coded by $(111111)_2$, $(34)_{10}=(100010)_2$, …, until $(2)_{10}$ which is coded with $(10)_2$. Except the MSB of each number (which is always equal to 1), the other less significant bits are placed in vertical position (as shown in Figure 6). The resulting sequences which are respectively S_0(1010), S_1(100110), S_2(1001111100), S_3(1101110100110) and S_4(10111110010100) represent the bit streams generated in only one subordinate pass.

T_0=32	63	34	49	47
T_1=16	31	23		
T_2=8	13	14	8	9
T_3=4	6	7	4	
T_4=2	2			

	63	34	49	47	31	23	13	14	8	9	6	7	4	2
S_0 :	1	0	1	0										
S_1 :	1	0	0	1	1	0								
S_2 :	1	0	0	1	1	1	1	1	0	0				
S_3 :	1	1	0	1	1	1	0	1	0	0	1	1	0	
S_4 :	1	0	1	1	1	1	1	0	0	1	0	1	0	0

Figure 6. Subordinate Pass of All Iterations of Vector Test

It is clear that in the lossless reconstruction case, all bit streams are taken in consideration. In the counter part, for lossy reconstruction, when stopping in a threshold T_i, only $S_0, S_1, \ldots, S_i$ are needed for the decoder. The reconstruction magnitudes are calculated at the level i using expression (4):

$$T_0 + \sum_{j=0}^{i} b_j T_{j+1} + \frac{T_{i+1}}{2} \tag{4}$$

Where b_j for $j=0..i$ are the bits generated from level 0 to level i. For example, approximating the first coefficient 63 (which starts from level 0) until the level 1 yield to:

$$T_0 + b_0.T_1 + b_1.T_2 + \frac{T_2}{2} = 32 + 1.16 + 1.8 + 4 = 60$$

Similarly, the coefficient 31 (which starts from level 1) is reconstructed at the level i=1 (for example) by:

$$\frac{T_0}{2} + \sum_{j=1}^{i} b_j T_{j+1} + \frac{T_{i+1}}{2} \tag{5}$$

$$\frac{T_0}{2} + b_1 T_2 + \frac{T_2}{2} = 16 + 1.8 + 4 = 28$$

In general approximating a coefficient starting from level k until a level of approximation i is governed by (6):

$$\frac{T_0}{2^k} + \sum_{j=k}^{i} b_j T_{j+1} + \frac{T_{i+1}}{2} \tag{6}$$

The reconstruction is progressive and the refinement is ensured according to the addition of S_i sequences.

At the reception of S_0 by the decoder, the restored information is:

63	34	49	47	31	23	13	14	8	9	6	7	4	2	63	34
56	40	56	40	-	-	-	-	-	-	-	-	-	-	-	-

By adding the sequence S_1, the information is refined to:

63	34	49	47	31	23	13	14	8	9	6	7	4	2	63	34
60	36	52	44	28	20	-	-	-	-	-	-	-	-	-	-

The same reasoning is preserved until the needed reconstruction quality to be reached.

This approach is simple and it requires less computing time than Shapiro's algorithm [13].

In the following, the bits "0" and "1" of *S* lists will be replaced by "*R*" and "*U*" respectively.

4.3. RLE and Arithmetic Coders

Preliminary symbol stream coder contains the block diagram shown in Figure 7.

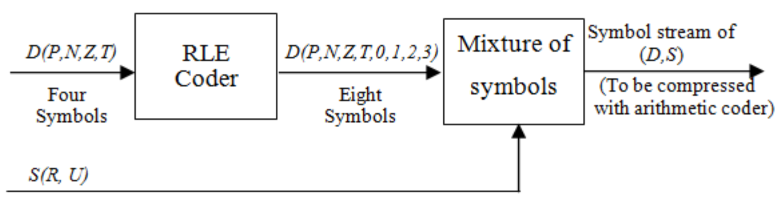


Figure 7. Block Diagram of Symbols Coder

Our approach, distributes entropy differently than Shapiro's algorithm [13] by using the four symbols used in EZW and four other symbols. In order to increase compression ratio, pre-processing approach (RLE) is implemented for symbol stream of dominant pass before arithmetic compression. The RLE compression method converts consecutive identical symbols into a code consisting of the character and the number marking the length of sequence. The symbols are encoded as the count of repetitive characters followed by the symbol itself. In our approach, if the count of repetitive character equal to 1 (no repetition) the symbol is inserted without the repetition number. This algorithm is very easy to implement and it does not take much CPU power. Note that the numbers of repetitions are converted from decimal to base 4. In this case only the numbers 0, 1, 2 and 3 are generated. Four new symbols are inserted with the four symbols used by Shapiro's algorithm [13].

For illustration, the following results show the RLE encoding for two iterations. We consider that the string of data to be compressed is:

D_0 : *N P Z P P T T T P P P P P P Z Z Z Z* (18 symbols)

D_1 : *P Z Z T T T N P T T Z Z Z Z P* (15 symbols)

The resulting RLE compressed list of symbol stream is:

D_0 : *N P Z 2 P 3 T 1 2 P 1 0 Z* (13 symbols)

D_1 : *P 2 Z 3 T N P 2 T 1 0 Z P* (13 symbols)

Each sequence of *D* list is grouped with bit stream of *S* list (*U*, *R*) as indicating in the following:

$D_0\ S_0\ D_1\ S_1\ \ldots$

Finally, an arithmetic coder is used. It is the method of choice for adaptive coding on multi-symbol alphabets because of its speed, low storage requirements, and efficiency of compression.

5. Experimental Results

In order to test the performance of the proposed method, the PCG databases available at [18, 19, 20, 21] have been used. Signals "Normal", "4hs", "Sumg" [18], "As-early", "Lateas", "Mr", "Ps", "Ar", "Ms" [19], "Late", "Split", "Dia", "S3", "S4", "NL" [20] and "A_regurg" [21] were used. The resolution of the input PCG waveforms tested is 16 bits with different sampling frequencies. Many measures of performance exist. Two measures are considered, the well-known PRD and CR. The PRD is expressed in (7).

$$PRD=\sqrt{\frac{\sum_{k=1}^{n}\left(X_{ori}[k]-X_{rec}[k]\right)^2}{\sum_{k=1}^{n}\left(X_{ori}[k]\right)^2}}x100 \tag{7}$$

Where X_{ori} and X_{rec} denote the original and the decompressed signal respectively. The CR is defined as the total number of bits ratio of the original and compressed signal, i.e.

$$CR=\frac{Total\ number\ of\ bits\ of\ X_{ori}}{Total\ number\ of\ bits\ of\ X_{rec}} \tag{8}$$

Note that we have used the bior4.4 mother wave to achieve a pyramidal decomposition up to the level 10.

In order to demonstrate the effectiveness of our algorithm, a first comparative study is given facing results of the original EZW, Modified EZW (MEZW) and EEZW (proposed method) using “Normal” PCG signal (Table 1).

In the paper [12], Ouafi has developed the MEZW method to reduce the total amount of information. Using six instead of the four symbols used in EZW, new symbols “*Pt*” and “*Nt*” are inserted. This method optimizes the coding by a binary grouping of elements before coding. Note that in our proposed method (EEZW), eight symbols (*P, N, T, Z, 0, 1, 2 and 3*) were used besides the two symbols of subordinate lists.

Table 1 shows appearance frequency, probability and entropy of symbols used by cited methods. In our approach, the redistribution of probabilities offers better results than the EZW and MEZW. Thus, the total amount of information in the EEZW algorithm is less than the EZW and MEZW algorithm, making the coding more effective.

The entropy of each symbol *i* used is given by:

$$H_i=-log_2\, p_i \tag{9}$$

The total information *L* is calculated by:

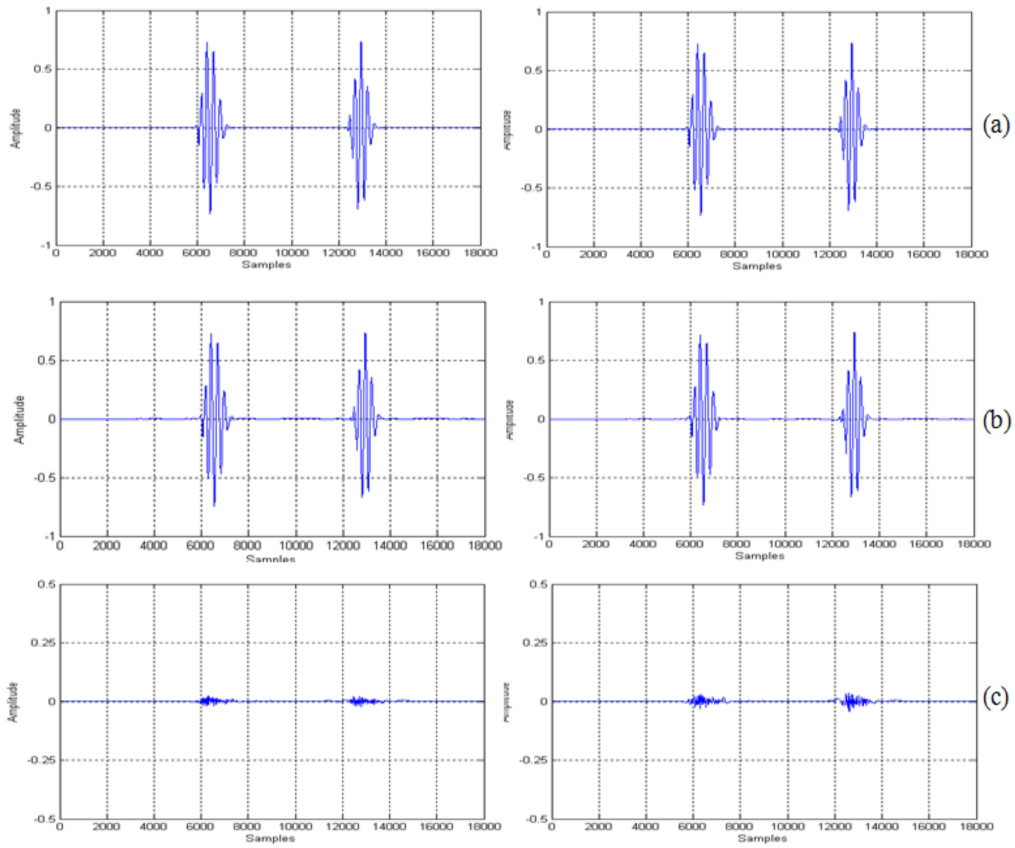
$$L=\sum_{i=1}^{n} f_i.H_i \tag{10}$$

n is the total number of symbols used in each technique. *n*=6, 8 and 10 for EZW, MEZW and for the proposed method EEZW respectively.

Table 1. Appearance Frequency, Probability and Entropy of Symbols of the EZW, MEZW and EEZW applied to “Normal” for a Threshold Th = 500

Symbol	EZW			MEZW			**Proposed method (EEZW)**		
	Appearance frequency *fi*	Probability p_i	Entropy H_i (bits)	Appearance frequency *fi*	Probability p_i	Entropy H_i (bits)	**Appearance frequency *fi***	**Probability p_i**	**Entropy H_i (bits)**
T	7602	0,4103	1,2851	4501	0,2918	1,7770	**2691**	**0,2032**	**2,2989**
Z	5100	0,2753	1,8610	5100	0,3306	1,5967	**2101**	**0,1587**	**2,6560**
P	789	0,0426	4,5534	51	0,0033	8,2406	**649**	**0,0490**	**4,3508**
N	789	0,0426	4,5534	53	0,0034	8,1851	**645**	**0,0487**	**4,3597**
Pt	-	-	-	738	0,0478	4,3855	-	-	-
Nt	-	-	-	736	0,0477	4,3894	-	-	-
0	-	-	-	-	-	-	**283**	**0,0214**	**5,5482**
1	-	-	-	-	-	-	**665**	**0,0502**	**4,3156**
2	-	-	-	-	-	-	**1461**	**0,1103**	**3,1801**
3	-	-	-	-	-	-	**501**	**0,0378**	**4,7242**
U	1920	0,1036	3,2704	1920	0,1245	3,0061	**1920**	**0,1450**	**2.7859**
R	2326	0,1256	2,9936	2326	0,1508	2,7293	**2326**	**0,1757**	**2.5092**
	Total information L= 39688 bits			Total information L= 35583 bits			**Total information L= 28855 bits**		

Figure 8 shows the visual performance of the proposed approach using “Normal” PCG signal, which compares the original signal with two reconstructed PCG signals for two thresholds.



CR=191.29 and PRD=2.96% CR=271.33 and PRD=4.62%

Figure 8. (a) A sample waveform of original signal "Normal" (b) The reconstructed waveforms of the compressed signals (c) error signals

The reconstructed signals shown in figure 8.b are obtained with: (CR=191.29:1 and PRD=2.96%) and (CR=271.33:1 and PRD=4.62%). However, Figure 8.c presents the resulting errors. We remark, the good compression ratios obtained according to a very good reconstruction quality. It means that the clinical information is preserved after compression/decompression.

Figures 9 and 10 present quantitative results of the elaborated technique. It is clear that our approach is comparable to the EZW and the MEZW (high compression ratios according to a good retrieved quality).

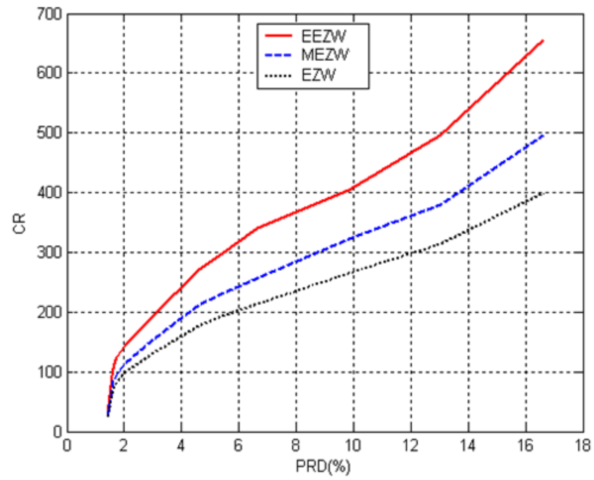


Figure 9. Comparison of Different Compression Methods Applied to "Normal" PCG Signal

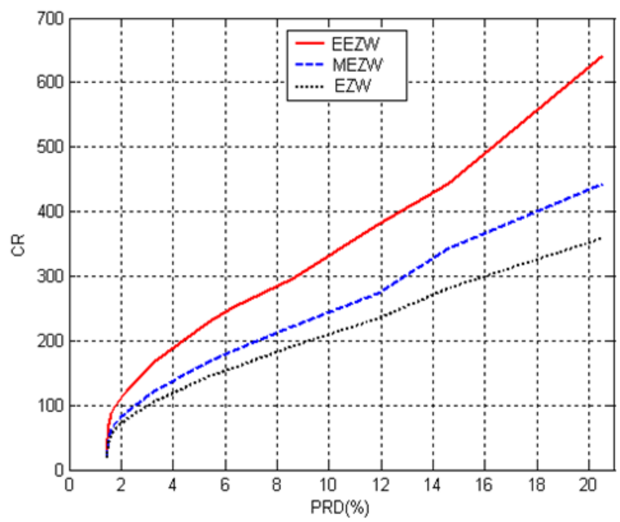


Figure 10. Comparison of Different Compression Methods Applied to "4hs" PCG Signal

The proposed technique is also compared with the popular MP3 coder using the previously mentioned signals of the database (Table 2). Results obtained in the Table 2 show that CR obtained by our technique are very high compared to CR obtained by the MP3 encoder with a bit rate 32kbps. For example CR=167.35:1 and PRD=5.35% for "S3" PCG signal using our approach against CR=5.30:1 with the same PRD for the same PCG signal applying MP3 coder. EEZW is very effective for all cited PCG signals. This confirms that our approach outperforms also the MP3 coder.

Table 2. Experimental Results for some PCG Signals in the Databases [18, 19, 20, 21]

PCG record	Sampling frequency	**EEZW CR**	MP3 32kbps CR	PRD (%)
S3		**167.35**		5.35
S4		**192.97**		5.43
NL	11025 Hz	**126.00**	5.30	5.49
Dia		**66.18**		5.49
Late		**67.60**		5.80
Ar		**45.38**		5.24
As_early		**149.02**		5.05
Lateas		**130.01**		5.19
Mr		**56.15**		5.31
Ms	22050 Hz	**140.84**	10.90	5.07
Ps		**87.43**		5.15
Sumg		**214.92**		5.18
4hs		**226.90**		5.31
Normal		**286.32**		5.08
A_regurg	44100 Hz	**117.76**	21.97	10.43

The process of searching for coefficients in each interval and in each refinement pass in EZW technique (detailed in Section 4.2) needs a considerable number of operations of comparisons to return the corresponding symbols ("0" or "1").

In our approach, in only one subordinate pass, the bit-plane (Figure 6) constituted by natural binary values of coefficients is generated requiring less time than EZW algorithm using Matlab.

Different tests were performed to evaluate the low computational complexity of our approach. Executed by a Pentium4 3.4GHz, the tests are applied to 16384 samples of PCG signals: "Normal", "S3", "Late", "Ps" and "A_regurg". The average time of subordinate passes using EZW is between 2 seconds and 5 seconds. Our technique needs only 1 until 2 seconds for the subordinate pass.

6. Conclusion

In this paper, we have proposed an enhanced version of EZW algorithm, applied to PCG signals. The main advantages of EEZW are:

- Low computational complexity compared to Shapiro [13] and MEZW of Ouafi, et, al., [12]. It is due to the only-one subordinate pass resulting after the all dominant passes.

- Our algorithm can be considered as the best compression method compared to EZW and MEZW. Also, our technique shows its high effectiveness by comparison with the well-known MP3 format. The quality of the reconstructed PCG signals is very good for the analysis and diagnosis.

Finally, EZW, MEZW and especially our approach (EEZW) demonstrate their efficiency in PCG compression.

References

[1] O. Say, Z. Dokur and T. Olmez, “Classification of heart sounds by using wavelet transform”, Proceedings. of the 2nd Annual International Conference of the IEEE-EMBS, **(2002)**, pp.128-129.

[2] S. M. Debbal and F. Bereksi-Reguig, “Automatic measure of the split in the second cardiac sound by using the wavelet transform technique”, Computers in Biology and Medicine, vol. 37, **(2007)**, pp. 269-276.

[3] S. M. Debbal and F. Bereksi-Reguig, “Computerized heart sounds analysis”, Computers in Biology and Medicine, vol. 38, **(2008)**, pp. 263-280.

[4] S. R. Debbal, J. Agzarian and D. Abbott, “Optimal wavelet denoising for phonocardiograms”, Microelectronics Journal, vol. 32, **(2001)**, pp. 931-941.

[5] W. -C. Kao, W. -H. Chen, C. -K. Yu, C. -M. Hong and S. -Y. Lin, “Portable real-time homecare system design with digital camera platform”, IEEE Transactions on Consumer Electronics, vol. 51, no. 4, **(2005),** pp. 1035-1041.

[6] J. Martinez-Alajarin and R. Ruiz-Merino, “Wavelet and wavelet packet compression of phonocardiograms”, Electronics Letters, vol. 40, **(2004)**, pp. 1040-1041.

[7] M. D. M. Elena, J. M. Quero and I. Borrego, “Optimal selection of wavelet coefficients for electrocardiograph compression”, ETRI Journal, vol. 29, **(2007)**, pp. 530-532.

[8] R. Benzid, F. Marir and N. E. Bouguechal, “Electrocardiogram compression method based on the adaptive wavelet coefficients quantization combined to a modified two-role encoder”, IEEE Signal Processing Letters, vol. 14, **(2007)**, pp. 373-376.

[9] M. Pooyan, A. Taheri, M. Moazami-Goudarzi and I. Saboori, “Wavelet compression of ECG signals using SPIHT algorithm”, Proceedings of World Academy of Science, Engineering and Technology, vol. 2, **(2005)**, pp. 284-287.

[10] M. Sabarimalai Manikandan and S. Dandapat, “Wavelet threshold based ECG compression using USZZQ and Huffman coding of DSM”, Biomedical Signal Processing and Control, vol. 1, **(2006)**, pp. 261–270.

[11] R. Shantha Selva Kumari and V. Sadasivam, “A novel algorithm for wavelet based ECG signal coding”, Computer and Electrical Engineering, vol. 33, **(2007)**, pp. 186–194.

[12] A. Ouafi, A. Taleb Ahmed, Z. Baarir and A. Zitouni, “A modified embedded zerotree wavelet (MEZW) algorithm for image compression”, Journal of Math Imaging Vis, vol. 30, **(2008)**, pp. 298–307.

[13] J. M. Shapiro, “Embedded image coding using zerotrees of wavelet coefficients”, IEEE Transactions on Signal Processing, vol. 41, **(1993)**, pp. 3445-3462.

[14] Z. Lu, D. Y. Kim and W. A. Pearlman, “Wavelet compression of ECG signals by the set partitioning in hierarchical trees algorithm”, IEEE Transactions on Biomedical Engineering, vol. 47, **(2000)**, pp. 849-856.

[15] G. Tohumoglu and K. Erbil Sezgin, “ECG signal compression by multi-iteration EZW coding for different wavelets and thresholds”, Computers in Biology and Medicine, vol. 37, **(2007)**, pp. 173–182.

[16] N. Boukhennoufa, K. Benmahammed, M. A. Abdi and F. Djeffal, “Wavelet-based ECG signals compression using SPIHT technique and VKTP coder”, International Conference on Signals, Circuits and Systems - SCS **(2009)** November 6-8; Medenine, Tunisia.

[17] S. M. Ahmed, A. Al-Shrouf and M. Abo-Zahhad, “ECG data compression using optimal non-orthogonal wavelet transform”, Medical Engineering & Physics, vol. 22, **(2000)**, pp. 39-46.

[18] http://www.dundee.ac.uk/medther/Cardiology/3hs.htm.

[19] http://depts.washington.edu/physdx/heart/demo.html.

[20] http://www.egeneralmedical.com/listohearmur.html.

[21] http://www.openheartsurgery.com/heart_sounds.html.