An Efficent Lossless Compression Scheme for ECG Signal

O. *El B'charri, R. Latif, A. Abenaou, A. Dliou, W. Jenkal
ESSI, National School of Applied Sciences
Ibn Zohr University
Agadir, Morocco

***Abstract*—Cardiac diseases constitute the main cause of mortality around the globe. For detection and identification of cardiac problems, it is very important to monitor the patient's heart activities for long periods during his normal daily life. The recorded signal that contains information about the condition of the heart called electrocardiogram (ECG). As a result, long recording of ECG signal amounts to huge data sizes. In this work, a robust lossless ECG data compression scheme for real-time applications is proposed. The developed algorithm has the advantages of lossy compression without introducing any distortion to the reconstructed signal. The ECG signals under test were taken from the PTB Diagnostic ECG Database. The compression procedure is simple and provides a high compression ratio compared to other lossless ECG compression methods. The compressed ECG data is generated as a text file. The decompression scheme has also been developed using the reverse logic and it is observed that there is no difference between original and reconstructed ECG signal.**

Keywords*—*ECG; lossless compression; data encoding; compression ratio

I. INTRODUCTION

In every year, according to an estimate given by the 2012 World Health Organization (WHO) statistics report, 56 million people died worldwide, of whom the heart diseases remained the leading cause of death throughout the world. The cardiovascular diseases killed 17.5 million people in 2012. That is 3 in every 10 deaths. Of these, 7.4 million people died of ischaemic heart disease and 6.7 million from stroke [1]. Some of these lives can be often saved if acute care by detecting the myocardial infarction early. So that those patients will have medical attention as soon as possible.

Electrocardiography is a fundamental part in both patient monitoring and cardiovascular assessment. It is an essential tool for investigating cardiac arrhythmias and is also useful in diagnosing cardiac disorders such as myocardial infarction [2]. It deals with the electrical activity of the central of the blood circulatory system, i.e., the heart. The contraction and relaxation of cardiac muscle result from the depolarization and repolarization of myocardial cells. These electrical changes produce currents that radiate through the surrounding tissue to the skin. The electrodes sense electrical currents when they are hooked up to the skin. The recorded currents are then transformed into waveforms called electrocardiogram (ECG).

To detect and identify cardiac problems, ECG signals should be continuously monitored typically over 24h period. Digitizing the ECG signals is performed at sampling rates ranging from 100 to 1000 Hz with a resolution of 8 or 12 bits per sample [3]. As a result, the data sizes of the produced ECG recording will enormously increase and fill up available storage space. Nowadays, storage space is relatively cheap. However, ECG data archives could easily become exceedingly large and expensive. Moreover, in mobile monitoring environments, compression is a fundamental tool to resolve and transmit physiological signals from the human body to any location, especially for real-time transmission applications.

The ECG compression methods are generally classified into two main groups, namely one-dimensional (1-D) and two-dimensional (2-D). 1-D ECG compression algorithms are the most widely employed in literature and can be further classified into four categories: direct time domain compression, model based compression, transformed domain compression and hybrid compression algorithms. In 2-D ECG compression algorithms, 1-D ECG signals are represented in 2-D then the transformation is applied on those 2-D representation.

The classification of ECG compression algorithms can also be observed by another angle, lossy and lossless compression. Although lossy compression has an important benefit of high Compression Ratio (CR), it introduces distortion into the original ECG signal and may lose some features that could be very important for future analysis of crucial patients. Considering lossless compression, it provides a moderate to high CR and maintains the original ECG signal away from any notable distortion..

Motivating by works of Mukhopadhyay et al. [4-6], which are based on voltages to text encoding, we propose a novel lossless ECG compression scheme that combines the advantages of the stating works. In [4], authors developed a lossless compression method that can preserve all the information in the reconstructed signal. However, the compression ratio has an unsatisfactory performance. The authors have also developed lossy compression algorithms in [5,6]. In [5], the entire signal values are quantified in the compression process while this quantification is only applied outside of QRS regions in [6]. The QRS regions are compressed using the same process as [4]. These lossy compression methods can achieve high CR but the major problem is that the reconstructed signal has a significant distortion that may even delete some important features in the ECG signal such as the T wave. Furthermore, from juridical and clinical point of view [7], it is strongly recommended to work on lossless ECG signal compression.

To overcome the aforementioned shortcomings, we developed a powerful lossless compression scheme that is able to reduce significantly the file size which provide a high compression ratio, all maintaining near-zero distortion in the reconstructed signal leading to an excellent quality score.

The paper is organized as follows. The proposed method of the ECG data compression and reconstruction is detailed in Section II. Section III presents the results of the proposed method of selected ECG records from the PTB diagnostic ECG database (PTB-DB). The performance analysis and comparison with other methods are also discussed. To conclude, some remarks and discussion are given in Section IV.

II. METHODOLOGY

The proposed algorithm is generally based on work done in [4], which achieves almost negligible distortion and a moderate good compression ratio. In this work, the compression ratio is highly improved by reducing the number of data sent to the output text file. Another improvement is made at the time of reconstruction to preserve the original signal reliably.

The compression scheme is divided into two main sections: data compression and data reconstruction. The key to this high compression lies in data encoding process and the rearrangement of the variables used in the output text file. The two ECG processing modules are described in the following of this section.

A. ECG data compression

The overall compression scheme is illustrated in figure 1. The boxes and the various shapes outlined in this figure are detailed step by step in the following of this subsection.

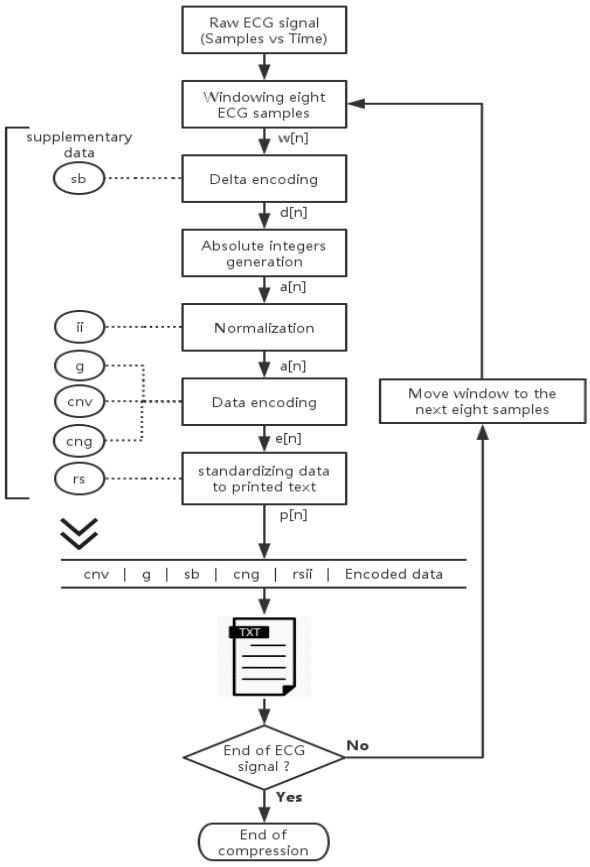

Fig. 1. Flowchart of the proposed scheme

1) Windowing eight ECG samples

The first step of the compression scheme is to take only ECG samples from the raw ECG signal file. The time axis is discarded since the sampling frequency is known. The ECG signal can be easily reconstructed by corresponding each sample to its equivalent instant of time. The whole compression procedure is applied on eight samples at a time until the end of the signal is reached.

2) Delta encoding

In order to get better compression, delta encoding is performed on those eight samples by subtracting two consecutive samples. The first value remained unchanged as described below.

$$d(0) = w(0)$$
$$for\ i = 1\ to\ 7$$
$$\quad d(i) = w(i) - w(i-1)$$
$$end$$

The motivation behind delta encoding is to get smaller number, which means concatenation possibility will increase in the later step of compression. Consequently, compression ratio will also increase.

3) Absolute integers generation

The sign of the each element of the array d[n] is verified. For those that contain positive numbers a binary zero (0) is made, and those containing negative numbers a binary (1) is made. The string obtained from those zeros and ones, which represents the sign of those eight samples, is converted to its equivalent decimal number and stored in a variable named ‘sb’. The absolute value of each sample is then multiplied by 1000 to avoid the fractional part since in standard ECG database; samples are recorded up-to three decimal points. This procedure is performed as follows:

$$for\ i = 0\ to\ 7$$
$$\quad if\ d(i) < 0$$
$$\quad\quad sb(i) = 1$$
$$\quad else$$
$$\quad\quad sb(i) = 0$$
$$\quad end$$
$$\quad a(i) = abs\{d(i)\} * 1000$$
$$end$$
$$(sb)_{10} \leftarrow (sb)_2$$

4) Normalization

In the delta encoding step, the first sample is kept unchanged. The problem starts when the ECG data begin with a high voltage (greater than |0.255|). As those samples after multiplying by 1000 become greater than 255 they cannot be printed in the output text file. Moreover, another problem has been observed at the time of dissociation in the decompression scheme that will be discussed in later section. So the first sample is chosen to be less than 200. To overcome this issue, the following instructions is used:

$$ii = a(0)/200$$
$$a(0) = a(0)\%200$$

Where 'ii' is assigned with the integer value from the division operation and a(0) is holding the reminder of this division.

5) Data encoding

In this section of the algorithm the main compression has occurred. To reduce the data size, these eight samples are minimized using the following logic:

$$if\ ([a(i) * 100] + a(i+1) < 255)$$
$$e(j) = [a(i) * 100] + a(i+1)$$
$$j = j + 1$$
$$else\ if\ (a(i) + [a(i+1) * 100] < 255)$$
$$e(j) = a(i) + [a(i+1) * 100]$$
$$j = j + 1$$
$$end$$

For the above-described encoding algorithm, the variable 'i' is initialized by zero and incremented by a factor of two.

The samples are encoded and stored in a new array as described above. The new array is constructed from the previous one by either concatenating or not two consecutive samples:

- Regular direction: If two consecutive samples undergoes the first condition, then these samples are concatenated in the regular direction.
- Opposite direction: If the first condition is not satisfied then we check that whether the second formula is verified or not. If so, then these consecutive samples are concatenated in the opposite direction.
- None: If concatenation is not possible in both direction then, those samples are left unchanged and stored separately in the new array.

The encoding step is achieved. However, in the decoding scheme we cannot identify what type of concatenation is carried out. To memorize the type of concatenation, instead of using three variables as done in [4], we take only one variable 'g'. This variable is taken as a binary number (one byte). Each 1 is inserted in its even bits corresponds to the regular direction concatenation (it is supposed that the most significant bit is an even bit), while a 1 is inserted in the odd bits corresponds to the opposite direction concatenation. If two consecutive even and odd bits equals to zero means that there was no concatenation.

It has been observed that some concatenated data lead to wrong samples at the decompression scheme. For an example purpose, suppose an encoded value like '209'. It could be concatenated from three case:

▪ Case 1	▪ Case 2	▪ Case 3
0 and 209	1 and 109	2 and 9

The first case is eliminated during the normalization section (the numbers in a[n] array are less than 200). To differentiate the two other cases, we take a variable named 'cnv' as a binary number to denote the second case in the fourth last bits of 'cnv' (the fourth most significant bits). The third case will be treated as a normal concatenation case.

To further clarify the data encoding step, let us take an array a[n] having sample values as shown in figure 2. In the following scheme, Reg and Opp denote regular direction and opposite direction, respectively.

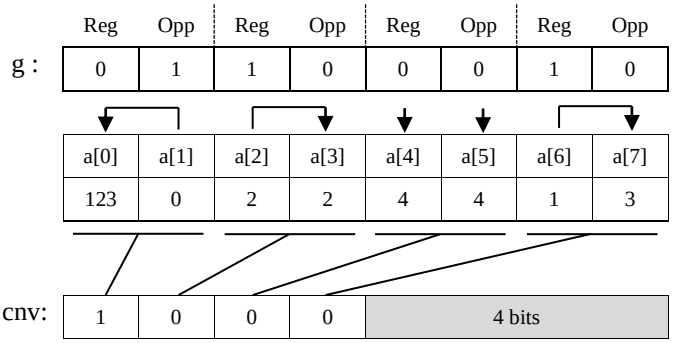

Fig. 2. An illustrative example of data encoding step

6) Standardizing data to printed text

The ECG samples are encoded in the range of text characters. However, some special characters are not stamped in the text file, which results to loss of data at the decompression stage. Those characters are 10 (line feed), 13 (carriage return), 26 (substitution) or 255 (blank). To avoid such problem, the numbers corresponding to those characters are substituted by some suitable numbers. To denote the modified numbers, it is necessary to add extra variables as described below.

- An extra variable named 'rs' is temporary taken which signifies whether the 'sb' is one of those special characters or not. If the variable 'sb' is changed by other number, 'rs' will be set to 1 (rs = 1) otherwise to 0 (rs = 0). Finally 'rs' is multiplied by 100 and is added with 'ii'. The result is stored in a new variable named 'rsii' which minimizes the 'rs' and 'ii' variables in one variable.
- A variable say 'cng' is taken as binary number to denote the positions of those special characters in e[n].
- There is a seldom probability that 'rsii', 'cng', and 'g' may contain any of those special characters. If it happens, some other suitable numbers will replace them. These changes will be denoted in the three first

unused bits of 'cnv' respectively (i.e. the three least significant bits). The 'cnv' variable will never be a critical number if we put a zero in its forth bit as described in figure 3.

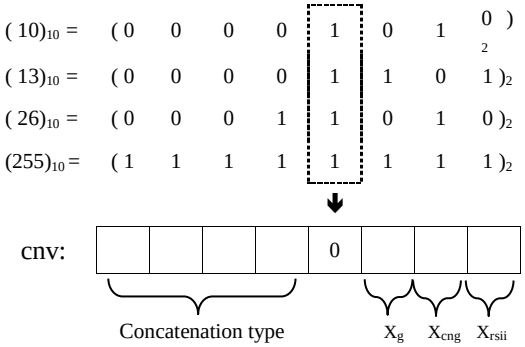

Fig. 3. The construction of the 'cnv' variable

Where X_i can be either 1 or 0 value depending if it was a special character or not.

Finally the encoded data along with all the necessary information (cnv, g, sb, cng, and rsii) are organized as a frame in the sequence shown in Figure 1 and are printed in their text code in a text file. The length of each frame may vary according to the concatenation possibilities. The order of these characters has a particular importance. Since from the variables 'cnv' and 'g' we can easily determine the length of each frame sent to the text file, the frames are sent tied to each other without any separation character. Following this procedure, we can minimize one separation character for each sent frame, which leads to a better compression.

Moving the window to the next eight samples, the whole compression scheme is repeated until the end of ECG signal is achieved.

B. ECG data decompression

Decompress the ECG data is necessary to retrieve the original signal at the receiving end. For this purpose, a decompression algorithm is also developed. Since the decompression scheme uses the same reverse logic described in figure 1, we briefly describe the procedure of the decompression. Some of the relevant steps are more detailed.

1) Extracting Frames from the text file

The first step of the decompression scheme is to identify the length of each frame from the continuous text contained in the compressed file. Here, the variable 'g' has an important feature apart from denoting the concatenation, it can also determines the length of the frame by calculating the sum of ones in 'g', then subtract this sum from 13. Where the value 13 represents the maximum length that can occupy a frame in the text file. This process is carried out by first checking the variable 'g' if it was a special character by testing on the third bit of the variable 'cnv' if it was 1. If it was the case, the original value of the variable 'g' is placed back. The decompression procedure is then applied on the extracted frame.

2) Replacement of the original numbers

Since the order of the variables is known in each frame, the modified data during subsection A.6 can be easily recovered by analyzing the variables 'cnv', 'cng' and 'rsii'.

3) Data decoding

The binary form of the variable 'g' gives the positions of concatenation. If it is found a one (1) in even bits (starting from the most significant bit) that represent the regular direction, while a one (1) in the odd bits represent the opposite direction. Or else, if two consecutive even and odd bits equal to zero (0), then there was no concatenation. These concatenated numbers are then dissociated according to the type of concatenation labeled in the four most significant bits of the variable 'cnv',

Now 'ii' is multiplied by 200 and added with a[0], the result is stored in a[0] position as the reverse was done during the 'Normalizing' subsection in the compression procedure.

As used in the 'Normalizing' subsection in the compression procedure, the variable 'ii' is multiplied by 200 and added to a[0]. The result is stored in the same a[0].

4) Signed values recovering

In this section, the variable 'sb' will be converted into its corresponding 8-bits equivalent. If any bit is '1' the corresponding positional element of d[n] array will be multiplied by (-1). The next step consist of dividing every number in this array by 1000.

5) Creating original ECG signal

To produce the original ECG signal, we have to generate ECG samples and the corresponding time axis. To get the original ECG samples, we use the reverse of delta encoding, e.i. each positional number is added with the previous value except the first one. Finally, each sample is stamped with its equivalent instant of time using the known sampling frequency.

Moving to the next frame, the whole decompression scheme is repeated until the end of the text file is achieved.

III. RESULTS AND DISCUSSION

A. Evaluation factor of the compression scheme

The criteria for testing the performance of the compression algorithms consist of three important components: compression measure, reconstruction error and computational complexity.

The computational complexity component is often attached to practical implementation consideration, which is recommended to be as simple as possible.

The Compression Ratio (CR) represents the ratio between the size of the file containing original and compressed signal, given by:

$$CR = \frac{B_0}{B_c} \quad (1)$$

Where B_0 is the total number of bits in the original file and B_c is the total number of bits in the compressed file.

The maximum absolute error is defined as the maximum element of sample-to-sample difference array:

$$E_{max} = Max(x_0(n) - x_r(n)) \quad (2)$$

Where $x_0(n)$ is the original signal, $x_r(n)$ is the reconstructed signal.

The definition of the error criterion for assessing the distortion of the reconstructed signal compared to the original one is of primary importance, the Percentage Root-mean-square Difference (PRD) measure defined as:

$$PRD(\%) = \sqrt{\frac{\sum_{n=1}^{N}\left(x_0(n) - x_r(n)\right)^2}{\sum_{n=1}^{N} x_0^2(n)}} \times 100 \quad (3)$$

Where N represents the window length, over which the PRD is calculated. In all scientific literature interested by ECG compression techniques, evaluation of this error is crucial and is the one commonly used. The clinical acceptability of the reconstructed signal is based on this criterion.

The normalized version of PRD is PRDN, which is independent of the signal mean value $\bar{x}$, is defined as:

$$PRDN(\%) = \sqrt{\frac{\sum_{n=1}^{N}\left(x_0(n) - x_r(n)\right)^2}{\sum_{n=1}^{N}(x_0(n) - \bar{x})^2}} \times 100 \quad (4)$$

One another evaluation factor given in (5), is Quality Score (QS). It quantifies the global performance of compression algorithm taken into account both the CR and PRD.

$$QS = \frac{CR}{PRD} \quad (5)$$

This all factors are evaluated on the proposed algorithm, and are given on their average value in Table 1.

The ECG data files under test are chosen from PTB diagnosis ECG database available under Physionet. To assess and compare the performance, the leads of every record were choosing the same as [4]. From table 1, we can see that using the proposed algorithm we can get average PRD of about 0.0092% and average CR of about 18.84:1. Since the developed method stands in the direct time domain compression and the compressed samples are encoded without any data truncation, the reconstructed signal from the decompression scheme is guaranteed to be the same as the original one without any distortion. Hence, we can have a zero PRD value. It should be noted that PRD, PRDN and E_{max} values depend strongly on the length of the original signal. If this length is a multiple of eight, these values is confirmed to be zero. Otherwise, the remaining ECG samples will not be compressed and will not be presented in the decompressed ECG signal which generate a non-zero PRD, PRDN and E_{max}. Regarding the compression ratio, we can observe that record n° S0305 provide the highest CR value, which means that the concatenating possibility was high compared to other signals.

To provide an overall estimate of the computational complexity of the algorithm, we performed an amount of simulation to each record then we calculate the average compression time of each signal. Compression time shows that this algorithm can be used for real-time systems.

Figure 4 shows the original (blue-a) and reconstructed (green-b) ECG signals of the proposed ECG compression scheme. Through visual inspection of this figure, It is obvious that the reconstructed signal is the same as the original ECG signal since the compression scheme performed in this work is lossless.

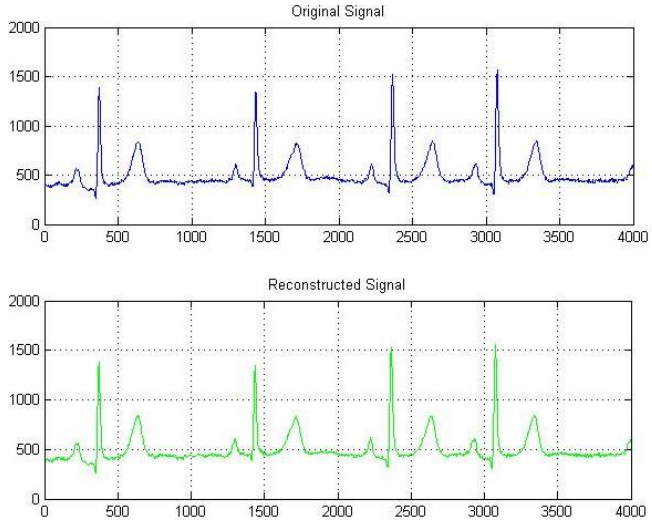

Fig. 4. Original (a) and Reconstructed (b) ECG signal. (File: S0305, Lead I, Duration: 4s)

TABLE I. THE PERFORMANCE MEASUREMENT OBTAINED FOR VARIOUS ECG FILES

Records	Performance evaluation contents *					
	Duration (s)	*PRD (%)*	*PRDN (%)*	*CR*	$\lvert Emax \rvert$	*Compression time (ms)*
S0022LRE	15	0.0000	0.0000	18.27	0	329.34
S0021ARE	15	0.0000	0.0000	18.86	0	417.67
S0015LRE	15	0.0000	0.0000	17.87	0	389.83
S0304	26.342	0.0082	0.0099	18.70	0.0003	717.00
S0305	23.770	0.0292	0.0811	19.76	0.0083	642.18
S0301	22.084	0.0181	0.0191	19.61	0.0028	613.74

(*) All the factors are represented in the average value

B. Performance comparaison

Comparison of some compression's evaluation factor of various methods with the proposed method is given in Table 2. It is divided into three parts namely lossy compression, near lossless compression and lossless compression.

The lossy compression methods are known by their high compression ratio but unfortunately, they may lose some

important information about the signal that can be noted in high values of PRD, while the near lossless compression algorithms give a moderate compression ratio, preserving the value of the PRD as low as possible. In the lossless compression algorithms, the importance is given to the value of PRD, which should be almost zero.

If lossy compression methods have a high compression rate, our method can be classified with these methods while maintaining almost zero distortion.

TABLE II. PERFORMANCE COMPARAISON OF VARIOUS ECG COMPRESSION ALGORITHMS

Algorithm	PRD (%)	CR	QS	PRDN (%)
Lossy compression				
m-AZTEC [8]	25.5	5.6	0.22	-
Mukhopadhyay et al. [5]	7.89	15.72	1.99	20.60
Mukhopadhyay et al. [6]	7.58	22.47	2.97	13.28
Near lossless compression				
USZZQ and Huffman coding of DSM [9]	2.73	11.06	4.05	-
SPIHIT [10]	1.18	8	6.78	-
Lossless compression				
JPEG2000 [11]	0.86	8	9.30	-
Fira and Goras [12]	0.61	12.74	20.89	48.38(max) -7.42(min)
SangJoon Lee et al. [13]	0.61	16.5	27.11	-
Mukhopadhyay et al. [5]	0.023	7.18	312.17	-
Proposed	**0.0092**	**18.84**	**2047**	**0.018**

Among the four lossless compression algorithms presented in table 2, the proposed method provides high CR (18.84), which is comparable with the CR of lossy compression methods generally known by their high compression ratio. The values of PRD and PRDN are negligible (almost zero) and are better than any other methods, wich leads to a very high QS (2047).

Our algorithm was based on the work of [4]. If we compare the results of our work with their results, we can found that the CR is considerably increased compared to the value of their CR. The value of the PRD is also improved since we treated some specific cases during the reconstruction of the signal.

IV. CONCLUSION

In this research, a high lossless compression scheme for ECG signals is proposed. The proposed algorithm was tested for the compression of normal and pathological types of cardiac beats ECG signals. This technique provides significant improvement in term of compression ratio compared to other lossless compression techniques, all ensuring a near-zero distortion that is notable, either through visual inspection, or the measured value of error loss. Consequently, the quality score that quantify the overall performance was far superior to the other compared algorithms. The proposed algorithm is simple and easy to implement. The output text file can be further compressed, using some standard text compression techniques. The compressed file can be either stored or transmitted over wireless network as text file for real time ECG analysis. The proposed scheme is suitable to use in portable and mobile ECG data monitoring system.

REFERENCES

[1] World Health Organization (WHO), Statistics 2012. Available online: http://www.who.int/mediacentre/factsheets/fs310/en/ (accessed on 21 July 2016).

[2] F. Morris, W. J. Brady and J. Camm, “ABC of Clinical Electrocardiography”, 2nd ed., Blackwell Publishing Ltd, 2008, pp. 1.

[3] M. S. Manikandan and S. Dandapat, “Wavelet threshold based TDL and TDR algorithms for real-time ECG signal compression”, Elsevier Ltd Biomedical Signal Processing and Control, vol. 3, 2008, pp. 44–66.

[4] S. K. Mukhopadhyay, S. Mitra, and M. Mitra, “A lossless ECG data compression technique using ASCII character encoding”, Computers and Electrical Engineering, vol. 37, 2011, pp. 486–497.

[5] S. K. Mukhopadhyay, S. Mitra, and M. Mitra, “An ECG signal compression technique using ASCII character encoding”, Measurement, vol. 45, 2012, pp. 1651–1660.

[6] S. K. Mukhopadhyay, M. Mitra, S. Mitra, “ECG signal compression using ASCII character encoding and transmission via SMS”, Biomedical Signal Processing and Control, vol. 8, 2013, pp. 354– 363.

[7] Koski A, “Lossless ECG encoding”, Computer Methods and Programs in Biomedicine, vol. 52, January 1997, pp. 23–33.

[8] V. Kumar, S.C. Saxena, V. K. Giri and D. Singh, “Improved modified AZTEC technique for ECG data compression: Effect of length of parabolic filter on reconstructed signal”, Computers and Electrical Engineering, vol. 31, issues 4-5, June-July 2005, pp. 334–344.

[9] M. S. Manikandan and S. Dandapat, “Wavelet threshold based ECG compression using USZZQ and Huffman coding of DSM”, Biomedical Signal Processing and Control, vol. 1, Issue 4, October 2006, pp. 261-270.

[10] Z. Lu, D. Y. Kim, W. A. Pearlman. “Wavelet compression of ECG signals by the set partitioning in hierarchical trees algorithm”, IEEE Trans. Biomedical Engineering, vol. 47, issue 7, July 2000, pp. 849–856.

[11] A. Bilgin, M. W. Marcellin and M. I. Altbach, “Compression of electrocardiogram signals using JPEG2000”, IEEE Trans. Consumer Electronics, vol. 49, issue 4, November 2003, pp. 833–840.

[12] Fira CM and Goras L. “An ECG signals compression method and its validation using NNs”, IEEE Trans. Biomed. Eng., vol. 55, no. 4, April 2008, pp. 1319–1326.

[13] S. J. Lee, J. Kim, and M. Lee, “A Real-Time ECG Data Compression and Transmission Algorithm for an e-Health Device”, IEEE Trans. Biomedical Engineering, vol. 58, issue 9, September 2011, pp. 2448-2455.