26th IMEKO TC4 International Symposium
24th International Workshop on ADC and DAC Modelling and Testing
IMEKO TC-4 2023
Pordenone, Italy / September 20-21, 2023

Lossless real-time signal encoding for two-channel signals: A case study on ECG

Jozef Kromka, Ondrej Kovac, Jan Saliga

Technical University of Kosice, Letna 9, 04200 Kosice, Slovakia
e-mail addresses: {jozef.kromka, ondrej.kovac, jan.saliga}@tuke.sk

***Abstract* – This paper introduces a new encoding algorithm for the lossless, fast, and memory-efficient compression of two-channel signals. The proposed algorithm was evaluated using Electrocardiogram (ECG) signals. The results indicate that it can effectively compress ECG signals with a favorable compression ratio (CR), without relying on complex predictors, dictionaries, detectors, or additional encoding methods. By employing this method, an average CR of 1.98 was achieved, with a minimum CR of 1.54 and a maximum CR of 2.36. Moreover, it is assumed that the algorithm has the potential to enhance the achieved compression in Compressed Sensing (CS) systems.**

I. INTRODUCTION

Lossless encoding algorithms are a type of compression techniques that allows for the reduction of data size without losing any of the original information. This property makes them particularly useful for applications, where keeping the original quality of data is very important [1]. One of the fields where this property is vital is medicine, where doctors often need accurate data to precisely diagnose patients [2]. In modern medicine one of the most commonly studied signals is ECG. These signals are critical in the diagnosis and monitoring of heart conditions, and it is important to ensure that the data remains intact during the compression process. In addition to the medical field, CS systems [3], [4] could also benefit from the use of lossless encoding algorithms. The use of such algorithms in CS systems can aid in reducing the data size while maintaining the original CS information intact. This, in turn, leads to more efficient and effective compression and transmission of data, resulting in improved CS system performance.

In recent years, numerous lossless encoding algorithms have been proposed and tested on ECG signals. In one study [5], an adaptive linear prediction and two-stage Huffman coding approach was used to compress ECG signals. Another study [6] used an adaptive region prediction and variable length coding method. A peak detection and backward difference Huffman coding approach was presented in [7]. Lastly, two algorithms [8] and [9] utilizing adaptive linear prediction and Golomb-Rice coding were proposed. While these algorithms can achieve a CR greater than 2, they have the disadvantage of requiring some form of complex prediction algorithm or a dictionary, or they have a complex implementation. Hence, the encoding algorithms mentioned earlier may not be suitable for use in CS systems. CS systems are often required to compress data while maintaining low power consumption. The additional complexity introduced by these algorithms would increase the power consumption of such systems, rendering them impractical for CS applications.

This article presents a preliminary case study of the proposed lossless real-time encoding algorithm for ECG signals. The choice of ECG signals is based on the assumption that the proposed algorithm can be effectively applied to these signals. The main benefits of the proposed encoding algorithm include its memory efficiency, ease of implementation, lack of reliance on complex predictors, dictionaries, detectors, or additional encoding methods. The proposed method's advantages are intended to be utilized in real-time systems, including CS applications. However, at present, the proposed method has only been evaluated on full signals and has not been tested on signals sampled by CS systems.

The article is organized as follows: In Section II, the overall system design for the proposed encoding algorithm together with a flowchart is presented. The experimental results obtained by simulation are reported in III. The main conclusion and possible future work regarding the method are outlined in Section IV.

II. THE PROPOSED ENCODING ALGORITHM

In this paper, we propose an encoding algorithm inspired by the "quite OK image format" (QOI) which is a lossless image encoding algorithm presented by Dominic Szablewski in [10]. The algorithm has been formally verified by [11]. QOI is characterized by its ease of implementation and fast processing and incorporates commonly used techniques such as run-length encoding and dictionary-based compression.

The proposed algorithm uses some of the techniques that are used in QOI, but also some techniques that have been optimized for two-channel signals. The flowchart of the proposed encoding algorithm is shown in Fig. 1. The encoding method employs three variables in its process. The first variable, denoted by s, stores the current two-channel sample, while the second variable, l, holds the previous two-channel sample. The third variable, run, is utilized for storing run-length data. These variables are initialized to zero at the onset of the encoding process. Subsequently, the first two-channel sample is loaded, and encoding begins.

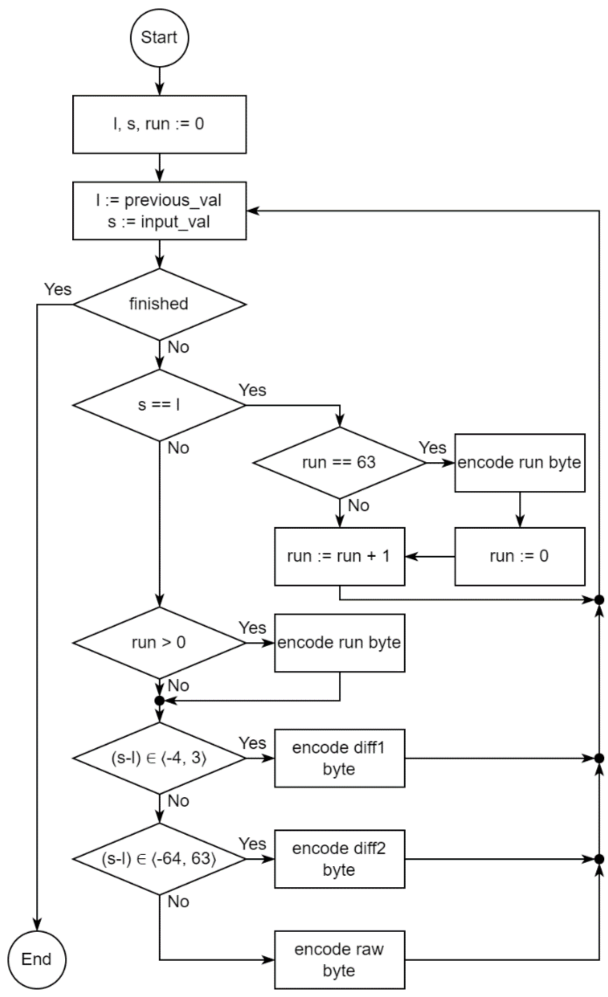

Fig. 1. The proposed encoding algorithm

The encoding process involves the comparison of the two-channel sample to the previous sample, and this comparison can yield four possible outcomes. Specifically, the sample may have the same value as the previous sample or the difference between the two may fall within the interval of $\langle -4, 3\rangle$. Alternatively, the difference between the two samples may lie within the interval of $\langle -64, 63\rangle$, or none of these cases. Each of these cases will be discussed in detail in the subsequent text.

In the initial case, where the value of the sample is the same as that of the previous sample, the run variable is incremented. Additionally, the algorithm verifies whether the run variable has surpassed the threshold of 63. If this threshold is exceeded, the run byte is encoded the run variable is set to zero. Moreover, the run byte is encoded if the run variable possesses a value greater than zero and the algorithm has not entered the initial case. It should be noted that the encoding of the run byte occurs prior to the execution of any other cases. The run byte is encoded and stored as illustrated in Fig. 2.

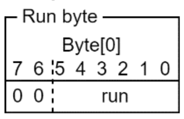

Fig. 2. Run byte encoding

The first two bits in the run byte represent the ID of the run byte, and the remaining 6 bits store the run variable.

When the difference between the current and previous samples falls within the interval of $\langle -4, 3\rangle$, the difference for both channels is calculated. This difference is then converted into a 3-bit number, encoded, and stored according to Fig. 3. The difference for the first channel is denoted as $d1$, while the difference for the second channel is denoted as $d2$.

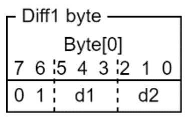

Fig. 3. Diff1 byte encoding

In the third case, when the difference between the two samples lies within the interval of $\langle -64, 63\rangle$ the encoding is similar to the previous case. The only difference is that two bytes are utilized, and the difference values are stored as 7-bit numbers as displayed in Fig. 4.

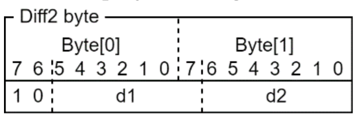

Fig. 4. Diff2 byte encoding

If none of the previously mentioned cases apply, a raw value of the two-channel sample is encoded. To encode raw values, three bytes are required. The raw values of the two-channel sample, denoted as $raw1$ and $raw2$, are stored as 11-bit values. This length was chosen because the database used for the case study contained data with this length. The encoding of raw values is depicted in Fig. 5.

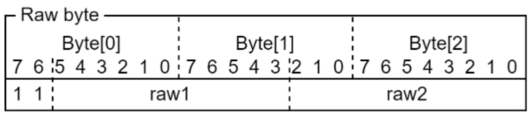

Fig. 5. Raw byte encoding

Each byte is encoded using a unique ID, which is employed in the decoding process to enable the decoding algorithm to determine how to decode the byte. The decoding process is a straightforward procedure. The decoding algorithm maintains a record of the preceding two-channel sample and reads the encoded bytes. Depending on the mask, the algorithm reads either one, two, or three bytes. If a run byte is encountered, the algorithm produces run samples of the previous value. If the diff1 or diff2 byte is read, the algorithm outputs the value of the previous sample plus the difference. In the final scenario, when the raw byte is read, the algorithm outputs only the raw data contained within that byte.

III. PRELIMINARY RESULTS AND DISCUSSION

The proposed method was evaluated using the MIT-BIH arrhythmia database [12]. This database contains a set of 48 ECG records sampled at 360 Hz with 11-bit resolution. The evaluation was based on two metrics: CR and the time required for encoding and decoding. The CR was calculated using (1)

$$\mathrm{CR} = \frac{n_i}{n_o}, \qquad (1)$$

where n_i represents the number of bits used in the original data and n_o represents the number of bits after compression.

Table 1 displays the average result obtained by the proposed method. The results from the previously mentioned methods are provided for comparison as well.

Table 1. Performance comparison of the proposed method with other methods in the MIT-BIH database

Encoding technique	Average CR
Adaptive linear prediction + two stage Huffman coding [5]	2.53
Adaptive region prediction + variable length coding [6]	2.67
Peak detection + backward difference Huffman coding [7]	2.64
Adaptive linear prediction + content adaptive Golomb-Rice coding [8]	2.77
Adaptive linear prediction + Golomb-Rice coding [9]	2.89
The proposed method	1.98

Based on the obtained results, it can be concluded that the proposed method produced an average CR that was approximately 25-45% lower than that of the other recent methods. As part of our analysis, we also report the minimum and maximum CR. The minimum CR of 1.54 was achieved for record 112 and the maximum CR of 2.36 was obtained for record 205.

In addition to evaluating the method's CR, an assessment was conducted to analyze its encoding and decoding speed, algorithm size, and the average number of instructions per sample. The preliminary evaluation was carried out on a personal computer equipped with an Intel Core i7-10700 processor, operating at a frequency of 2.9 GHz, and 16 GB of RAM. The proposed method was implemented in the C programming language and compiled on an Ubuntu operating system running under the Windows Subsystem for Linux. The algorithm was executed 1,000 times for each record resulting in average encode and decode times of 6.46 ms and 5.29 ms, respectively. With 650,000 samples per record, the average encode and decode times per sample were 9.93 ns and 8.14 ns. Notably, the simulation's performance may have been constrained by the operating system's multitasking, limiting access to full hardware resources. The algorithm's compiled size was around 5 kilobytes, though it can vary across systems. For 32-bit or 8-bit microcontrollers, the algorithm size could be significantly smaller. The final evaluation focused on assessing the average execution of instructions per sample. This assessment revealed that the system achieved an average execution of approximately 66 instructions, providing a quantitative measure of its computational performance. However, it's important to consider that this number would variate across different systems. Optimizing the algorithm's implementation could potentially reduce the instructions per sample, leading to improved performance.

While the proposed method achieves a lower CR than other recent methods, its primary advantage lies in the algorithm's simplicity of implementation and memory efficiency. In contrast to other methods, it does not require complex predictors, dictionaries, detectors, or additional encoding methods. The proposed method on the low level utilizes only subtraction and bit-shifting operations. Consequently, the proposed method is well-suited for use in systems that require real-time encoding or have limited memory or computing power resources. This feature of the proposed method could be utilized in CS systems to improve their efficiency as well. The proposed method would not significantly influence the energy consumption nor system resources and could potentially increase CR achieved by these systems.

It is worth noting that the current implementation of the method was designed such that all encoding bytes are multiples of 8 bits, which allows for byte alignment. This design feature makes it easier to implement the method on both microcontrollers and computers. We assume that it is possible to achieve a higher CR value by using encoding bytes with different lengths other than 8 bits. This approach may be well-suited for an FPGA device. The algorithm that inspired the proposed method [10] already has an FPGA implementation, which suggests that the proposed method could also be easily implemented for an FPGA device.

IV. CONCLUSION

This article introduced a novel encoding algorithm that is well-suited for the lossless, real-time, and memory-efficient encoding of two-channel signals. The proposed algorithm was tested on ECG signals, and the preliminary results demonstrated that it achieved an average CR of 1.98. Although this CR was 25-45% lower than that achieved by other recent methods, it was noted that the proposed method does not require the use of complex predictors, dictionaries, detectors, or additional encoding methods. These design features make the proposed algorithm a promising candidate for use in systems with limited memory or computing power resources, as well as in real-time applications.

Future work is directed to: (i) evaluating the efficiency of the proposed encoding algorithm for ECG signals from the PTB Diagnostic ECG dataset [13], [14], (ii) examining the efficiency of the method when applied to signals sampled by CS systems, (iii) performing a hardware implementation of the proposed method using a microcontroller and FPGA device, in order to thoroughly assess the algorithm's speed and memory efficiency.

ACKNOWLEDGMENT

The work is a part of the project supported by the Science Grant Agency of the Slovak Republic (No. 1/0413/22).

REFERENCES

[1] K. Sayood, *Lossless Compression Handbook*. Elsevier, 2002.

[2] J. F. Burnum, "The Misinformation Era: The Fall of the Medical Record", *Ann. Intern. Med.*, roč. 110, č. 6, s. 482–484, mar. 1989, doi: 10.7326/0003-4819-110-6-482.

[3] L. De Vito, E. Picariello, F. Picariello, S. Rapuano, a I. Tudosa, "A dictionary optimization method for reconstruction of ECG signals after compressed sensing", *Sensors*, roč. 21, č. 16, 2021, doi: 10.3390/s21165282.

[4] J. Kromka, O. Kováč, J. Šaliga, a L. Michaeli, "Multiwavelet-based ECG compressed sensing with samples difference thresholding", v *25th IMEKO TC-4 INTERNATIONAL SYMPOSIUM ON MEASUREMENT OF ELECTRICAL QUANTITIES*, Brescia, Italy, sep. 2022, s. 215–220.

[5] G.-A. Luo, S.-L. Chen, a T.-L. Lin, "VLSI implementation of a lossless ECG encoder design with fuzzy decision and two-stage Huffman coding for wireless body sensor network", v *2013 9th International Conference on Information, Communications & Signal Processing*, dec. 2013, s. 1–4. doi: 10.1109/ICICS.2013.6782955.

[6] K. Li, Y. Pan, F. Chen, K.-T. Cheng, a R. Huan, "Real-time lossless ECG compression for low-power wearable medical devices based on adaptive region prediction", *Electron. Lett.*, roč. 50, č. 25, s. 1904–1906, 2014, doi: 10.1049/el.2014.3058.

[7] S.-C. Lai, P.-C. Tail, M.-K. Lee, S.-F. Lei, a C.-H. Luo, "Prototype System Design of ECG Signal Acquisition with Lossless Data Compression Algorithm Applied for Smart Devices", v *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, máj. 2018, s. 1–2. doi: 10.1109/ICCE-China.2018.8448842.

[8] T.-H. Tsai a W.-T. Kuo, "An Efficient ECG Lossless Compression System for Embedded Platforms With Telemedicine Applications", *IEEE Access*, roč. 6, s. 42207–42215, 2018, doi: 10.1109/ACCESS.2018.2858857.

[9] T.-H. Tsai a F.-L. Tsai, "Efficient lossless compression scheme for multi-channel ECG signal processing", *Biomed. Signal Process. Control*, roč. 59, s. 101879, máj. 2020, doi: 10.1016/j.bspc.2020.101879.

[10] "QOI — The Quite OK Image Format". https://qoiformat.org/ (cit 03. február 2023).

[11] M. Bucev a V. Kunčak, Ed., "Formally Verified Quite OK Image Format", *Proc. 22nd Conf. Form. Methods Comput.-Aided Des. – FMCAD 2022*, 2022, doi: 10.34727/2022/isbn.978-3-85448-053-2_41.

[12] G. B. Moody a R. G. Mark, "The impact of the MIT-BIH Arrhythmia Database", *IEEE Eng. Med. Biol. Mag.*, roč. 20, č. 3, s. 45–50, máj. 2001, doi: 10.1109/51.932724.

[13] G. Al *et al.*, "PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals", *Circulation*, roč. 101, č. 23, jún. 2000, doi: 10.1161/01.cir.101.23.e215.

[14] R.-D. Bousseljot, D. Kreiseler, a A. Schnabel, "The PTB Diagnostic ECG Database". physionet.org, 2004. doi: 10.13026/C28C71.