QOE – Lossless Real-Time Multichannel Signal Encoding Algorithm

Ondrej Kovac, Antonia Kovacova, Jan Saliga, Jozef Kromka

Technical University of Košice, Faculty of Electrical Engineering and Informatics, Letná 9, 04200 Košice, Slovakia

{ondrej.kovac; antonia.kovacova; jan.saliga; jozef.kromka}@tuke.sk

Abstract: This study presents the Quite OK Encoding (QOE), a novel encoding algorithm designed for the efficient compression of multi-channel signals with a focus on losslessness, speed, and memory utilization. The evaluation of this algorithm was conducted using an electrocardiogram (ECG), a seismograph, and water parameter signals. The results indicate that it can effectively compress ECG signals, without relying on complex predictors, dictionaries, detectors, or additional encoding methods. The experiments targeted both two-channel and twelve-channel ECG signals from the MIT-BIH arrhythmia database as well as the PTB Diagnostic ECG Database. The properties of the proposed algorithm were assessed concerning cross-correlation between channels. The compression properties of the proposed algorithm were evaluated as well with seismic and water analysis signals. The computing efficiency of the proposed algorithm was assessed on the ATMega328p and STM32F446RE microcontroller systems. The findings indicate that the STM32F446RE demonstrated superior results with fewer computing instructions required.

Keywords: Multichannel compression; Lossless compression; QOI; QOE

1 Introduction

Lossless encoding algorithms are a type of compression technique that allows 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 lossless encoding 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 the 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 recent years, numerous lossless encoding algorithms have been proposed and tested on ECG signals. In the study [3], an adaptive linear prediction and a two-stage Huffman coding approach were used to compress ECG signals. Another study

[4] used an adaptive region prediction and variable length coding method. A peak detection and backward difference Huffman coding approach was presented in [5]. The algorithms proposed in [6] and [7] employ adaptive linear prediction and Golomb-Rice coding. While these algorithms can achieve a Compression Ratio (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. In the study [8], compression of ECG based on the Run Length Encoding (RLE) was proposed. Although this approach achieves high compression ratios (around 18), it is lossy and relies on Discrete Wavelet Transform (DWT) decomposition. In another study [9], the ECG compression method using ASCII character encoding was presented. The relatively high compression (around 7) was achieved, but the method contains operations that lead to computational complexity. The multi-lead measurement of ECG enables the diagnosis of cardiovascular diseases that may only be visible in specific channels. The 3-lead ECG is normally sufficient, but for high-precision measurement, mainly in hospitals, the 12-lead ECG monitoring system is used [10], [11], [12]. The multi-channel nature of the data is particularly intriguing for encoding using the proposed method.

The seismograph signals are another typical example of multi-channel signals. We selected seismic activity signals due to the significant portion of these signals exhibiting either no change or only small fluctuations in amplitude. The proposed method is expected to achieve high compression ratios on seismic signals recorded during periods of low activity. Consequently, the proposed methods are expected to be well-suited for the long-term monitoring of slow-changing signals. According to [13], the seismic data are compressed through a quantization-based approach [14], prediction-based [15], transformation-based, Run Length Encoding [16] approach as well as compression based on machine learning [17], [18]. The study [19] presented a compression algorithm based on deep learning, where the compression ratio is strongly influenced by the SNR (signal-to-noise ratio). Furthermore, the method is single-channel oriented and too complex for microcontroller implementation. In [20], a highly effective multitone model-based seismic data compression method was proposed. While the method achieves high compression, it also imposes high demands on computational power. The proposed method could be considered, to some extent, a hybrid solution integrating RLE with elements of prediction-based techniques.

The multi-channel water parameter signals consist of signals originating from various sources. Probes placed on the surface of the river measure several parameters, including pH, salinity, temperature, and many others [21]. The proposed method is suitable for compressing multichannel signals of diverse origins.

This article presents a universal multichannel lossless real-time encoding algorithm. The proposed method was initially introduced at the 26th IMEKO TC4 International Symposium and the 24th International Workshop on ADC/DAC Modelling and Testing, featuring a case study focusing on ECG signals [22]. The main benefits of the proposed encoding algorithm include its memory efficiency, ease of implementation, and lack of reliance on complex predictors, dictionaries, detectors, or additional encoding methods. The proposed method's advantages are intended to be used in real-time systems, including Compressed Sensing (CS) applications. The CS systems [23], [24] could benefit from the use of lossless encoding algorithms.

The article is organized as follows: The second section describes the proposed algorithm. This algorithm is explained in terms of a two-channel implementation, where both the algorithm and the compression data structure used are clarified. Subsequently, multichannel compression is described. The third section focuses on the experimental evaluation of the proposed algorithm. At the beginning of the third section, a comparison of the proposed method with some of the aforementioned ECG compression methods is conducted. Subsequently, the implementation of the proposed method on microcontrollers is evaluated. The subsequent experiments, addressed in the third section, focus on the compression of signals with more than two channels, such as twelve-lead ECG, seismic activity, and water parameters. At the end of this section, the influence of cross-correlation between channels on the compression achieved by the proposed method is assessed.

2 The Proposed 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 [25]. The other algorithm of this author, which was inspiring for our work, is the lossy audio compression format, the “Quite OK Audio Format” (QOA) [26]. The QOI has been formally verified by [27], and is characterized by its ease of implementation and fast processing, and incorporates commonly used techniques such as run-length encoding as well as difference and dictionary-based compression. To achieve memory efficiency, the proposed method does not utilize dictionaries, instead, it relies on the RLE of sample differences. The RLE reduces the size of repetitive sequences by replacing consecutive identical elements with a count of the repetition and the element itself [28], [29]. For example, "AAAABBBCCDAA" would be encoded as "4A3B2C1D2A". RLE reduces redundancy with high effectiveness in data with long sequences of repeated elements. Due to its nature, RLE is not effective for data characterized by rapid amplitude fluctuations or frequent changes in values. The differentiation of samples is highly effective for suppressing redundancy before encoding, thereby introducing additive compression.

2.1 The Two-Channel QOE

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 Figure 1.

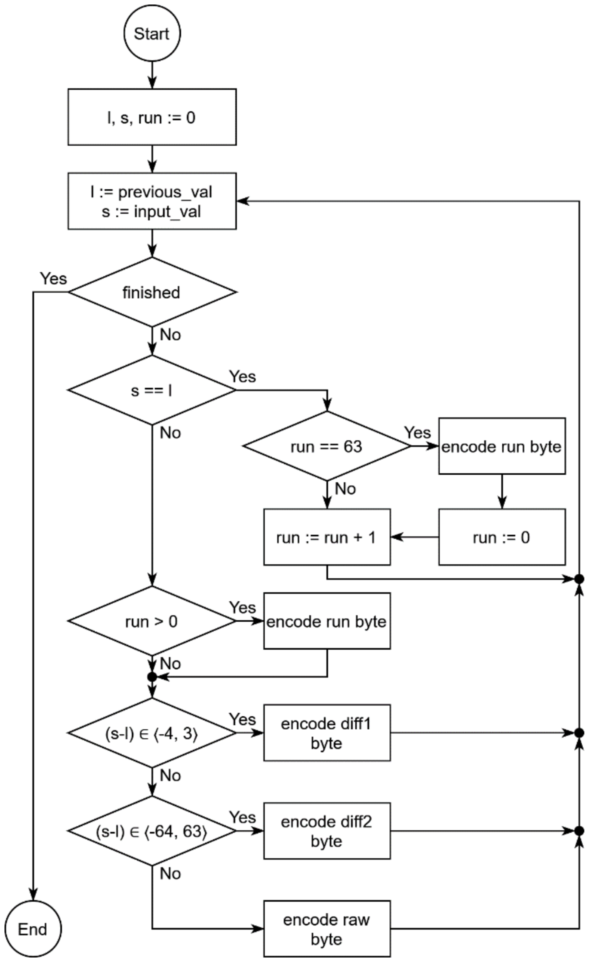


Figure 1

The proposed encoding algorithm

For simplification, Figure 1 shows the algorithm for two-channel encoding. 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 used for storing Run-length data. These variables are initialized to zero at the onset of the encoding process. Subsequently, 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, in any one of the channels, or the difference between the two may fall within the interval of $\langle -4, 3 \rangle$. Alternatively, the difference between the two samples, in any one of the channels, may lie within the interval of $\langle -64, -4) \cup (3, 63 \rangle$, or none of these cases. Each of these cases will be discussed in detail in the subsequent text.

In the initial case, when the current sample value matches the previous one, the run variable is incremented. The algorithm then checks whether this variable has exceeded the threshold of 63. If so, a Run packet is encoded, and the run variable is reset to zero. Moreover, the *Run packet* 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 packet* occurs before the execution of any other cases. The Run byte is encoded and stored as illustrated in Figure 2.

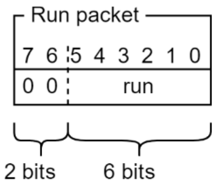


Figure 2
Run packet encoding

The first two bits in the *Run packet* represent the ID of the packet, 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 Figure 3a. The difference for the first channel is denoted as d_1, while the difference for the second channel is denoted as d_2. 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 used, and the difference values are stored as 7-bit numbers as displayed in Figure 3b. 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 raw_1 and raw_2, are stored as 11-bit values. The encoding of Raw values is depicted in Figure 3c. 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.

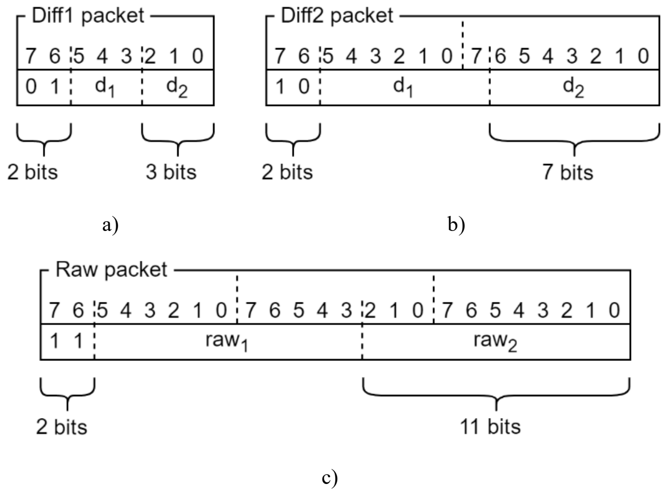


Figure 3
The packet used by two-channel QOE a) Diff1, b) Diff2, and c) Raw

The decoding process is a straightforward procedure. The decoding algorithm maintains a record of the preceding multi-channel sample and reads the encoded bytes. Depending on the mask, the algorithm reads either one, two, or three bytes. If a *Run* packet is encountered, the algorithm based on the value encoded in the *Run* packet produces samples of the previous value. If the *Diff1* or *Diff2* packet is read, the algorithm adds the value of the previous sample and the decoded difference. In the final scenario, when the Raw packet is read, the algorithm outputs only the Raw data contained within that packet.

2.2 The Multichannel-Channel QOE

The suggested approach is adaptable for two-channel as well as multichannel implementation. The block diagram of the multichannel QOE implementation is shown in Figure 4, where the length of the *Run packet* is intended to be 6 bits.

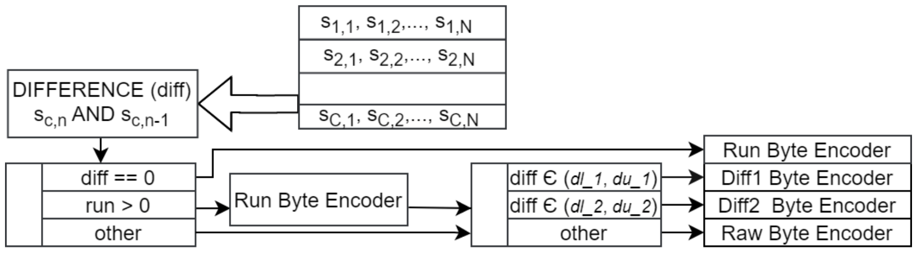


Figure 4
The block diagram of the multichannel QOE implementation

However, the *Run* packet's length can vary depending on signal properties; in such cases, the decision levels will be adjusted accordingly. In the block diagram, it is evident that a first-order difference is applied to each signal in the multichannel input. In the subsequent step, if all differences are equal to zero, the internal counter is incremented. When the counter surpasses a maximum value (it is 63 for "normal encoding"), the RunByte Encoder is activated. If the absolute value of the difference is greater than zero, the Run Encoder is employed to encode the preceding number of zero-value differences. Following this, the values of the differences or *Raw* data are encoded based on the maximum difference value among all channels. To maintain generality, decision levels in the block diagram are denoted as dl_1, du_1, dl_2, and *du_2*. In our experiments for the "normal encoding", these values are given as follows: $dl\_1 = -4, du\_1 = 2, dl\_2 = -64$, and $du\_2 = 63$. During the experiment, we will examine different constellations of packet lengths, and the scenario yielding the best results will be denoted as the "best-case". The *Run* packet is the same as for the two-channel implementation. The *Diff1*, *Diff2*, and *Raw* packets will exhibit similarities, with the sole distinction being their appropriate length corresponding to the number of channels. The packets used for multichannel encoding are shown in Figure 5.

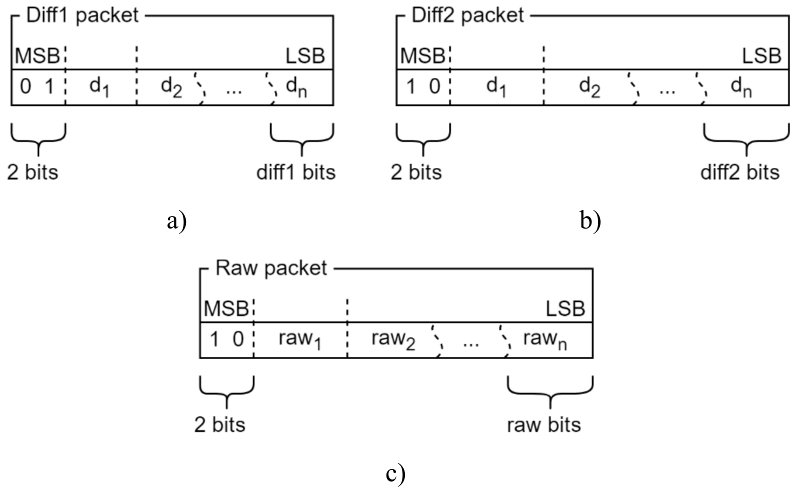


Figure 5

The packet used by multi-channel QOE a) Diff1, b) Diff2, and c) Raw

PSEUDOCODE:	
s	– samples (vector Nx1, where N is the number of channels)
ptr	– memory location where we save encoded data
pre_s	– previous samples (vector Nx1)
Run	– number of repeating samples
d1_l, d1_u	– lower and upper decision levels for Diff1 encoding
d2_l, d2_u	– lower and upper decision levels for Diff2 encoding
RunMax	– limit for Run (for 6b *RunMax = 64*)
// Global Variables (in RAM)	

```
pre_s = 0
Run = 0
FUNCTION encode_sample(s, ptr) RETURNS out_bytes
  out_bytes = 0
  IF ALL(s == pre_s) THEN
    IF Run >= RunMax THEN
      out_bytes ++
      encode_Run_sample(ptr)
      Run = 0
    END IF
    Run = Run + 1
    update_variables()
    RETURN out_bytes
  END IF
  IF Run > 0 THEN
    out_bytes ++
    encode_Run_sample(ptr)
    Run = 0
  END IF
  IF ALL(d1_l <= (s - pre_s) <= d1_u) THEN
    out_bytes ++
    encode_Diff1_sample(ptr)
    update_variables()
    RETURN out_bytes
  END IF
  IF ALL(d2_l <= (s - pre_s) <= d2_u) THEN
    out_bytes = out_bytes + 2
    encode_Diff2_sample(ptr)
    update_variables()
    RETURN out_bytes
  END IF
  out_bytes = out_bytes + 3
  encode_Raw_sample(ptr)
  update_variables()
  RETURN out_bytes
END FUNCTION
```

3 Experimental Results

The initial experiment was concentrated on assessing the CR compared with five methods outlined in [5], [6], [7], [8], [9]. Emphasizing the swift implementation of the other methods under consideration, all algorithms, including QOE, underwent evaluation on a PC. The evaluation used the MIT-BIH arrhythmia database [30]. This database contains a set of 48 ECG records sampled at 360 Hz with 11-bit resolution. The CR was calculated using (1).

$$\mathrm{CR} = \frac{n_i}{n_o}, \tag{1}$$

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 [3]	2.53
Adaptive region prediction + variable length coding [4]	2.67
Peak detection + backward difference Huffman coding [5]	2.64
Adaptive linear prediction + content adaptive Golomb-Rice coding [6]	2.77
Adaptive linear prediction + Golomb-Rice coding [7]	2.89
Proposed Quite OK Encoding	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 speed, an assessment was conducted to analyze its encoding and decoding speed, algorithm size, and the average number of instructions per sample. The 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. For timing evaluation, the function *clock()* and *clock_t* data type from *time.h* library were used. 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. Certainly, the simulation's performance may have been influenced by the multitasking capabilities of the operating system, thereby restricting access to complete hardware resources. Consequently, a more precise evaluation of the algorithm's speed is conducted on various MCUs in the subsequent subsection.

3.1 Different MCUs Implementation

The third experiment was focused on the proposed algorithm's performance implemented on various microcontroller systems. The evaluation was performed on the ATmega328p and STM32F446RE. The clock frequency of both MCUs was set to 16 MHz. The algorithm was implemented in C as well as the assembler (ASM) programming language. The performance was assessed on real hardware boards. MCU ATMega was programmed in the Arduino IDE and was implemented on Arduino UNO. The STM32F446RE was programmed in STM32CubeIDE and implemented on NUCLEO-F411RE. The measurement was performed using a 4-

channel RIGOL MSO5104 oscilloscope (100 MHz clock frequency and 8 GSa/s sampling rate). The encoding time was measured on pins 2 and 13 for STM and Arduino, respectively. When the encoding begins, the pin is set to logical 1, and when the encoding process completes, the pin switches to logical 0. At the beginning of the measure, the initial pulse was produced by the MCU to determine the duration of one pulse. The pulse duration can be easily converted to the number of MCUs' instructions. Subsequently, the next pulses are produced during *Run*, *Diff1*, *Diff2,* and *Raw* encoding. The measurements were performed based on the scenario for two-channel encoding. The scenario simulates the worst-case encoding time for all encoding cases. The scenario is given as follows:

```
PSEUDOCODE:
out_buff – output buffer address
// Experiment start pulse
set_pin()
reset_pin()
// Perform Run encoding
set_pin()
encode_sample(0, 0, out_buff)
reset_pin()
// Perform Diff1 encoding
set_pin()
encode_sample(0, 3, out_buff)
reset_pin()
// Perform Run encoding to make the next encoding longer
encode_sample(0, 3, out_buff)
// Perform Diff2 encoding
set_pin()
encode_sample(0, 63, out_buff)
reset_pin()
// Perform Run encoding to make the next encoding longer
encode_sample(0, 63, out_buff)
// Perform Raw encoding
set_pin()
encode_sample(0, 256, out_buff)
reset_pin()
// Experiment stop pulse
set_pin()
reset_pin()
```

For example, the output from RIGOL MSO5104 for C-programmed STM32 tested according to the scenario listed above is shown in Figure 6. The ATMega328p and the ASM programming were evaluated in the same way as the STM32 shown in Figure 6.

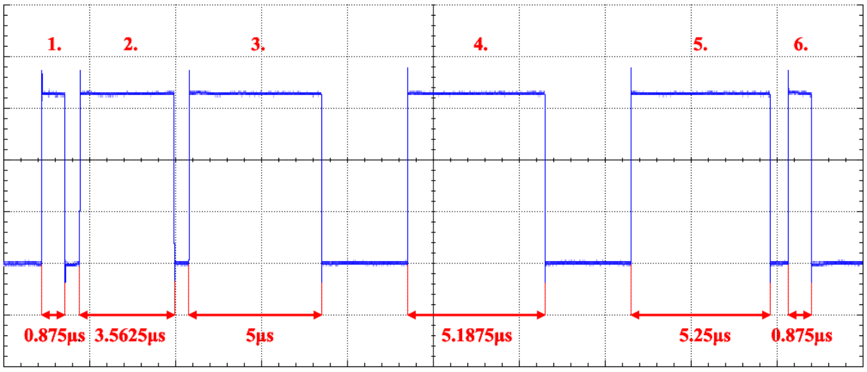


Figure 6

The output from RIGOL MSO5104 for the C-programmed STM32

The impulses are numbered in Figure 6 as follows: 1 – *Initial pulse*, 2 – *Run*, 3 - *Diff1*, 4 – *Diff2*, 5 – *Raw*, and 6 - *Ending pulse*. Hence, if 0.875 µs is the duration of the starting pulse, which has to be subtracted from the measured time, then the *Run* encoding for the 16 MHz system has (3,5625-0.875)/0.0625 = 43 instructions. The comparison of the compiled algorithm size and average number of executions for various microcontrollers is shown in Table 2.

Table 2

The comparison of the compiled algorithm size and average number of executions for the ATMega328p and STM32F446RE MCUs

MCU		ATMega328p			STM32F446RE	
Programming		**ASM (PUSH)**	**ASM (NO PUSH)**	**C**	**ASM**	**C**
Number of instructions	**RUN**	77 i	49 i	46 i	62 i	43 i
	DIFF1	114 i	86 i	106 i	82 i	66 i
	DIFF2	**121 i**	**93 i**	100 i	**87 i**	69 i
	RAW	112 i	84 i	**142 i**	86 i	**70 i**
Algorithm implementation size		252 B	224 B	380 B	252 B	208 B

The cells with a maximum number of instructions are highlighted, which represents the maximum possible time needed for encoding of the 2-channel sample. These results suggest that the STM32 provides much better performance. The best results (70 instructions and 208 B) are obtained for a C language programmed STM32. The best results for ATMega328p (93 instructions, and 224 B) are obtained for ASM implementation, where the used registers were not pushed on the stack (NO PUSH). Better results of STM32 are most probably achieved thanks to the 32-bit architecture compared with the 8-bit used by the ATmega328p. The better performance of STM programmed with the C language is most probably caused by the good optimization of the C compiler implemented in the STM32CubeIDE studio.

3.2 Compression Performance for Different Multichannel Signals

The previous experiments were focused on the two-channel signals. The next experiments will be focused on multichannel signals. ECG signals can be acquired from multiple leads, with options ranging from 1 to 12 leads. In the next experiment, the proposed method was applied to the ECG signals from another database popular among scientists. The PTB [31] database contains 12-lead ECG signals, thus, the multichannel compression of the proposed method can be evaluated. This database contains a set of 549 ECG records from 290 subjects, sampled at 1 kHz with 16-bit resolution. Besides the conventional 12 leads, the database contains 3 Frank lead ECGs, respiration, and line voltage channels, which were excluded from our experiments. The CR was obtained according to (1). The proposed method applied to 12-lead ECG signals from the PDB database was evaluated on a PC on all records. The results of encoding performance are listed in Table 3, where the normal encoding and best-case lengths are shown. For the normal encoding, the length was chosen as shown in Table 3. The algorithm was additionally evaluated for all reasonable combinations of *Run*, *Diff1*, and *Diff2* length. The average CR was calculated for the whole database, and then the combination with the highest CR is called the best-case. Based on the results listed in Table 3, it is possible to conclude that the QOE applied to 12-lead ECG signals achieves comparable results to the case of two-channel ECG encoding.

Table 3

The comparison of Run, Diff1, Diff2, and Raw packet lengths and the evaluation of the average CR achieved by the normal and best-case encoding of 12-lead ECG records (PTB Database)

	Normal encoding	**The best-case**
RUN	6	1
DIFF1	3	6
DIFF2	7	9
RAW	15	16
Average CR *(549 ECG records)*	**2.174** ±0.238	**2.350** ±0.192

Numerous signals could be processed across multiple channels. The seismic activity record serves as a typical example of multichannel signals. The upcoming experiment focuses on compressing a seismic activity record from the earthquake that occurred in Japan on January 1, 2024. The selected record for our experiment spans 2 hours (from 6 a.m. to 8 a.m. UTC) and was captured by an STS-2 device [32] located at the JSD station. The sampling frequency was 20 Hz with a 24-bit quantization resolution. The data were acquired from [33]. The quantization levels for our experiment were re-quantized to 11-bit, resulting in an average SNR of 35.25 dB across all three channels. In Table 4, the mean value, as well as the best achieved CR for all 3 channels, are listed.

Table 4

The comparison of Run, Diff1, Diff2, and Raw packet lengths and the evaluation of the CR achieved by the normal and best-case encoding of a 3-channel record of SA

	Normal encoding	**The best-case**
RUN	6	4
DIFF1	3	2
DIFF2	7	5
RAW	11	11
CR	**5.841**	**6.648**

The following case study presents an evaluation of time-varying CR achieved by the proposed method when applied to the Seismic Activity (SA) record. In Figure 7, the 3-channel recording of the Ishikawa Prefecture earthquake from January 2024 is depicted.

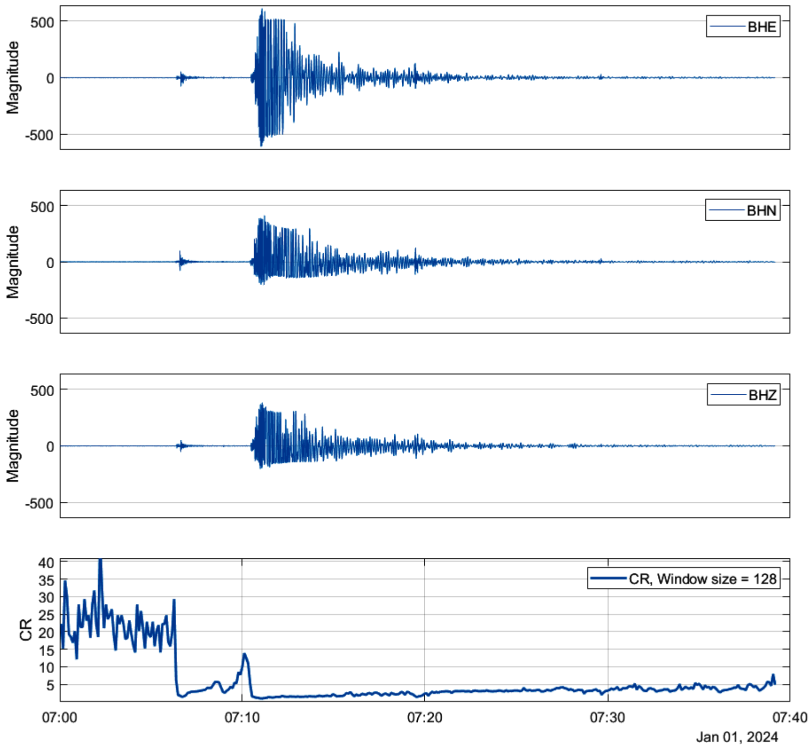


Figure 7

The 3-channel recording of the Ishikawa Prefecture earthquake from January 2024

Signals within the first 6 minutes represent a period of no SA, resulting in a high CR. The mean CR value is 22.22 with a standard deviation of 5.81. This high CR is attributed to the abundance of samples with identical values, allowing for efficient RLE. Over the following four minutes, there is a recording of low SA with a decreasing trend, leading to a gradual increase in CR. Moments before the earthquake, the CR reached approximately 14. During the main SA, all three channels exhibit high magnitudes, and the signal amplitudes change rapidly, resulting in a minimal CR of 1.02. As the SA diminishes, the CR gradually rises. However, records after the main wave contain numerous small oscillations, necessitating encoding with *Diff1* and *Diff2* packets. The mean CR value after the onset of the main wave is 3.19, with a standard deviation of 1.53. Based on this experiment, it can be concluded that the QOE is a suitable choice for long-term measurements where the observed phenomenon exhibits a slow-changing behavior. When rapid changes occur in the measured signals, the CR tends to be low. However, since high SA represents only a small portion of the total recording time, the lower CR during such periods does not significantly impact overall performance. Notably, the proposed method is compatible with low-power consumption MCUs, making it advantageous for implementation in outdoor measuring systems.

The signals encoded by QOE do not have to be for the same physical unit. In the next experiment, the Water Parameter Signals (WPS) [34] were encoded by the proposed method. The signals contained in WPS were of various origins. These parameters are commonly measured in water: temperature, conductivity, salinity, PH, oxygenation, and re-dox. The records used in our experiment were taken by 8 probes. In Figure 8, the record of atmospheric pressure, temperature, PH, and salinity of water from one of the probes placed in a river for four days is shown. From the chart, it is possible to conclude that the PH and salinity during the measurement time change slowly. The changes in the pressure are significant. These conclusions are valid for all of the probes used in the experiment.

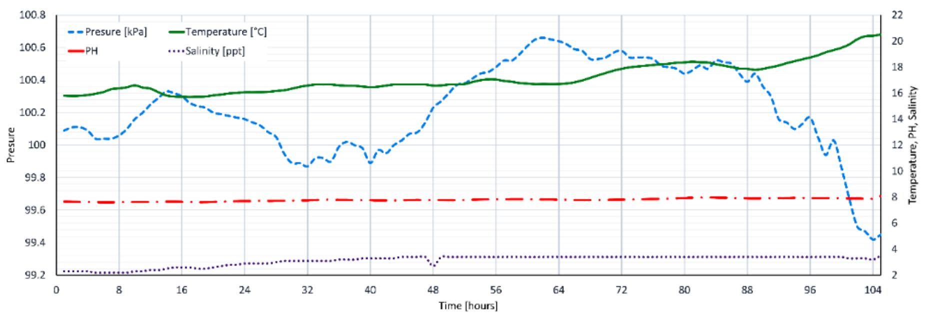


Figure 8
The WPS record - atmospheric pressure, temperature, PH, and salinity of water

Each record has 105 entries taken in a one-hour interval. The four-channel encoding was applied to pressure, temperature, PH, and salinity. In Table 5, the mean value, as well as the best achieved CR for all 8 probes, are listed.

Table 5
The comparison of RUN, DIFF1, DIFF2, and RAW packet lengths and the evaluation of the average CR achieved by the normal and best-case encoding of a 4-channel record of WPS

	Normal encoding	**The best-case**
RUN	6	1
DIFF1	3	4
DIFF2	7	6
RAW	11	11
Average CR *(8 probes)*	**1.649** ±0.0418	**1.991** ±0.0213

3.3 The CR Dependence on the Cross-Correlation between the Encoded Channels

The last experiments were conducted to evaluate the performance of the proposed method concerning cross-correlation [35] between encoded channels. The initial assumption is that higher channel correlation leads to higher CR, as more samples can be encoded using *Diff1* or *Diff2* instead of *Raw*. The experiment is evaluated statistically for the Lobachevsky University Electrocardiography Database (LUDB) [36]. The cross-correlation was evaluated by using (2).

$$\rho(x,y) = \frac{1}{N-1}\frac{\sum_{i=1}^{N}(x_i-\mu_x)(y_i-\mu_y)}{\sigma_x\sigma_y} \tag{2}$$

Where x and y are channels of ECG records, N is the number of samples, μ_x μ_y, are the mean values, and σ_x, and σ_y are the standard deviations, respectively.

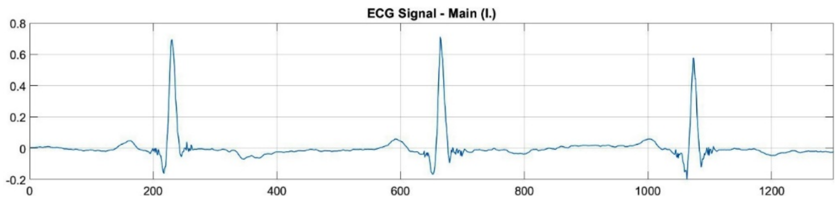
Channels I and V6 are on average highly correlated (84.59%). On the other hand, channels I and AVR have minimal correlation (2.55%). The results of compression for 50 signals of LUDB are shown in Table 6.

Table 6
The comparison of RUN, DIFF1, DIFF2, and RAW packet lengths and the evaluation of cross-correlation impact on the average CR achieved by the normal and best-case encoding of two channels of 12-lead ECG records (LUDB Database)

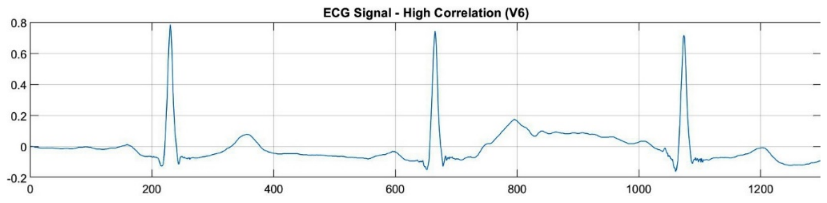
	Highly correlated ECG		**Low correlated ECG**	
	Normal encoding	**The best-case**	**Normal encoding**	**The best-case**
RUN	6	1	6	1
DIFF1	3	4	3	4

DIFF2	7	9	7	9
RAW	15	15	15	15
Average CR *(50 ECG records)*	**3.330** ±0.338	**4.643** ±0.362	**3.560** ±0.440	**4.999** ±0.410

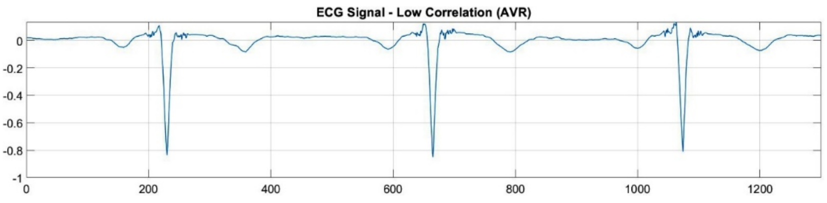
We show a case study to show the influence of cross-correlation on the CR achieved by QOE. In Figure 9 (a and b), two channels (I and V6) of record #16 (samples: from 712 to 2211) are shown. The cross-correlation between these channels is 89.49%. The achieved CR was 3.346. In Figure 9c, the same record is shown, but the second channel was selected by minimal cross-correlation (0.47%). In this case, the average CR was 3.237.



a)



b)



c)

Figure 9
The ECG signals of record #16 from the LUDB Database:
a) channel I, b) channel V6, and c) channel AVR

Based on the mean value of CR and standard deviation achieved during the last experiment, it is possible to conclude that the cross-correlation between channels of 12-lead ECG signals has minimal influence on CR.

Conclusion

This article introduced a novel encoding algorithm that is well-suited for the lossless, real-time, and memory-efficient encoding of multi-channel signals. This algorithm was named “Quite OK Encoding” (QOE). The algorithm is derived from the Quite OK Image algorithm, initially developed for fast and memory-efficient compression of images. The proposed algorithm was modified to multi-channel signal encoding.

Initially, QOE was tested on two-channel ECG signals from the MIT-BIH arrhythmia database, and the results demonstrated that it achieved an average CR of 1.98. 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 lowlevel uses only subtraction and bit-shifting operations. 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. Thus, the QOE was implemented in the ATmega328p and STM32F446RE microcontrollers as well. The results suggest that the STM32F446RE achieves better performance than the ATmega328p. Subsequent observations suggest that implementation in the C language is more effective than the ASM implementation on STM32F446RE. This is likely due to the optimization of the C compiler in STM32CubeIDE. The best results for ATMega328p were achieved by the ASM programming language without using push instructions.

To prove that the QOE is suitable for encoding various signals, we presented the experimental results achieved on 12-lead ECG signals from the Physikalisch-Technische Bundesanstalt (PTB) database, 3-channel recordings of SA during the earthquake as well as 4-channel water parameter output of probes placed in the Slovak rivers. Based on the results of these experiments, it is possible to conclude that the QOE is suitable for lossless encoding of various multichannel data. Based on the results for SA compression, it is possible to say that the proposed method achieves a high CR if the input signals exhibit very few changes. On the other hand, when significant changes occur in the signal (i. e. the earthquake), the CR goes lower. Hence, the proposed method is suitable for long-term measurement of the signals without very frequent significant changes. The compression of WPS data was less effective than that of seismic data, mainly because the measurements were taken only once per hour. For example, atmospheric pressure or temperature may vary significantly over one hour. If the measurements were taken within a much smaller interval, the CR could be higher. Since all evaluated signals (ECG, SA, and WPS) were obtained from real-world sources, they inherently include additive noise. As such, the reported compression results already account for the algorithm's behavior under realistic noise conditions.

The last experiment was focused on the cross-correlation between channels. This experiment was conducted on the recordings of a twelve-lead ECG from the LUDB. The main hypothesis was that if the channels are highly correlated, the CR should be higher. However, this hypothesis was not confirmed. Although the CR was slightly higher for low-correlated signals, we conclude that cross-correlation has limited influence on the compression efficiency of QOE.

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.*, Vol. 110, No. 6, pp. 482-484, Mar. 1989, doi: 10.7326/0003-4819-110-6-482

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

[4] K. Li, Y. Pan, F. Chen, K.-T. Cheng, and R. Huan, “Real-time lossless ECG compression for low-power wearable medical devices based on adaptive region prediction,” *Electron. Lett.*, Vol. 50, No. 25, pp. 1904-1906, 2014, doi: 10.1049/el.2014.3058

[5] S.-C. Lai, P.-C. Tail, M.-K. Lee, S.-F. Lei, and C.-H. Luo, “Prototype System Design of ECG Signal Acquisition with Lossless Data Compression Algorithm Applied for Smart Devices,” in *2018 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, May 2018, pp. 1-2, doi: 10.1109/ICCE-China.2018.8448842

[6] T.-H. Tsai and W.-T. Kuo, “An Efficient ECG Lossless Compression System for Embedded Platforms With Telemedicine Applications,” *IEEE Access*, Vol. 6, pp. 42207-42215, 2018, doi: 10.1109/ACCESS.2018.2858857

[7] T.-H. Tsai and F.-L. Tsai, “Efficient lossless compression scheme for multi-channel ECG signal processing,” *Biomed. Signal Process. Control*, Vol. 59, p. 101879, May 2020, doi: 10.1016/j.bspc.2020.101879

[8] M. H. Kolekar, C. K. Jha, and P. Kumar, “ECG Data Compression Using Modified Run Length Encoding of Wavelet Coefficients for Holter Monitoring,” *IRBM*, Vol. 43, No. 5, pp. 325-332, Oct. 2022, doi: 10.1016/j.irbm.2021.10.001

[9] S. K. Mukhopadhyay, S. Mitra, and M. Mitra, “A lossless ECG data compression technique using ASCII character encoding,” *Comput. Electr. Eng.*, Vol. 37, No. 4, pp. 486-497, Jul. 2011, doi: 10.1016/j.compeleceng.2011.05.004

[10] Y. Jin *et al.*, “Multi-class 12-lead ECG automatic diagnosis based on a novel subdomain adaptive deep network,” *Sci. China Technol. Sci.*, Vol. 65, No. 11, pp. 2617-2630, Nov. 2022, doi: 10.1007/s11431-022-2080-6

[11] D. Moses and D. C, “A survey of data mining algorithms used in cardiovascular disease diagnosis from multi-lead ECG data,” *Kuwait J. Sci. Partnering Elsevier*, Vol. 42, No. 2, pp. 206-235

[12] M. Shen, L. Wang, K. Zhu, and J. Zhu, “Multi-lead ECG classification based on Independent Component Analysis and Support Vector Machine,” in *2010 3rd International Conference on Biomedical Engineering and Informatics*, Yantai, China: IEEE, Oct. 2010, pp. 960-964, doi: 10.1109/BMEI.2010.5639841

[13] H. Nuha, M. Mohandes, B. Liu, and A. Al-Shaikhi, “Seismic Data Compression: A Survey,” in *Advances in Geophysics, Tectonics and Petroleum Geosciences*, M. Meghraoui, N. Sundararajan, S. Banerjee, K.-G. Hinzen, M. Eshagh, F. Roure, H. I. Chaminé, S. Maouche, and A. Michard, Eds., in Advances in Science, Technology & Innovation. , Cham: Springer International Publishing, 2022, pp. 253-255, doi: 10.1007/978-3-030-73026-0_58

[14] D. Salomon, *A concise introduction to data compression*. Springer Science & Business Media, 2007

[15] M. A. Razzaque, C. Bleakley, and S. Dobson, “Compression in wireless sensor networks: A survey and comparative evaluation,” *ACM Trans. Sens. Netw.*, Vol. 10, No. 1, pp. 1-44, Nov. 2013, doi: 10.1145/2528948

[16] A. Z. Averbuch, F. Meyer, J.-O. Stromberg, R. Coifman, and A. Vassiliou, “Low bit-rate efficient compression for seismic data,” *IEEE Trans. Image Process.*, Vol. 10, No. 12, pp. 1801-1814, Dec. 2001, doi: 10.1109/83.974565

[17] T. A. Reddy, K. R. Devi, and S. V. Gangashetty, “Nonlinear principal component analysis for seismic data compression,” in *2012 1st International Conference on Recent Advances in Information Technology (RAIT)*, Dhanbad, India: IEEE, Mar. 2012, pp. 927-932, doi: 10.1109/RAIT.2012.6194558

[18] H. H. Nuha, A. Balghonaim, B. Liu, M. Mohandes, M. Deriche, and F. Fekri, “Deep Neural Networks with Extreme Learning Machine for Seismic Data Compression,” *Arab. J. Sci. Eng.*, Vol. 45, No. 3, pp. 1367-1377, Mar. 2020, doi: 10.1007/s13369-019-03942-3

[19] E. B. Helal, O. M. Saad, A. G. Hafez, Y. Chen, and G. M. Dousoky, “Seismic Data Compression Using Deep Learning,” *IEEE Access*, Vol. 9, pp. 58161-58169, 2021, doi: 10.1109/ACCESS.2021.3073090

[20] B. Liu, M. Mohandes, H. Nuha, M. Deriche, F. Fekri, and J. H. McClellan, “A Multitone Model-Based Seismic Data Compression,” *IEEE Trans. Syst. Man Cybern. Syst.*, Vol. 52, No. 2, pp. 1030-1040, Feb. 2022, doi: 10.1109/TSMC.2021.3077490

[21] N. Hassan Omer, “Water Quality Parameters,” in *Water Quality - Science, Assessments and Policy*, K. Summers, Ed., IntechOpen, 2020, doi: 10.5772/intechopen.89657

[22] J. Kromka, O. Kováč, and J. Šaliga, “Lossless real-Time signal encoding for two-channel signals: A case study on ECG,” presented at the 26th IMEKO TC4 International Symposium and 24th International Workshop on ADC/DAC Modelling and Testing, Pordenone: International Measurement Confederation (IMEKO) 29.9 2023

[23] L. De Vito, E. Picariello, F. Picariello, S. Rapuano, and I. Tudosa, “A dictionary optimization method for reconstruction of ECG signals after compressed sensing,” *Sensors*, Vol. 21, No. 16, 2021, doi: 10.3390/s21165282

[24] J. Kromka, O. Kováč, J. Šaliga, and L. Michaeli, “Multiwavelet-based ECG compressed sensing with samples difference thresholding,” in *25th IMEKO TC-4 INTERNATIONAL SYMPOSIUM ON MEASUREMENT OF ELECTRICAL QUANTITIES*, Brescia, Italy, Sep. 2022, pp. 215-220

[25] “QOI — The Quite OK Image Format.” Accessed: Feb. 03, 2023 [Online] Available: https://qoiformat.org/

[26] D. Szablewski, “QOA - The Quite OK Audio Format for Fast, Lossy Compression.” Accessed: Feb. 14, 2023 [Online] Available: https://qoaformat.org/

[27] M. Bucev and V. Kunčak, Eds., “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

[28] B. Strasser, A. Botea, and D. Harabor, “Compressing Optimal Paths with Run Length Encoding,” *J. Artif. Intell. Res.*, Vol. 54, pp. 593-629, Dec. 2015, doi: 10.1613/jair.4931

[29] U. S. Mehta, K. S. Dasgupta, and N. M. Devashrayee, “Run-Length-Based Test Data Compression Techniques: How Far from Entropy and Power Bounds?—A Survey,” *VLSI Des.*, Vol. 2010, pp. 1-9, Mar. 2010, doi: 10.1155/2010/670476

[30] G. B. Moody and R. G. Mark, "The impact of the MIT-BIH Arrhythmia Database," *IEEE Eng. Med. Biol. Mag.*, Vol. 20, No. 3, pp. 45-50, May 2001, doi: 10.1109/51.932724

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

[32] S. Rosat, M. Calvo, J. Hinderer, U. Riccardi, J. Arnoso, and W. Zürn, "Comparison of the performances of different spring and superconducting gravimeters and STS-2 seismometer at the Gravimetric Observatory of Strasbourg, France," *Stud. Geophys. Geod.*, Vol. 59, No. 1, pp. 58-82, Jan. 2015, doi: 10.1007/s11200-014-0830-5

[33] The International Federation of Digital Seismograph Networks (FDSN), "Japan Meteorological Agency Seismic Network." Accessed: Feb. 07, 2024 [Online] Available: https://www.fdsn.org/networks/detail/JP/

[34] J. Šaliga *et al.*, "Multiparametric sensor network for water quality monitoring," presented at the IMEKO TC19 Workshop on Metrology for the Sea, Naples, IT, 13.10 2017

[35] P. Bourke, "Cross correlation, autocorrelation, 2d pattern identification"

[36] A. Kalyakulina *et al.*, "Lobachevsky University Electrocardiography Database." PhysioNet. doi: 10.13026/WDHQ-RS83