REWARD: Design, Optimization, and Evaluation of a Real-Time Relative-Energy Wearable R-Peak Detection Algorithm*

Lara Orlandic[1], Elisabetta de Giovanni[1], Adriana Arza[1], Sasan Yazdani[2], Jean-Marc Vesin[2], and David Atienza[1]

***Abstract*— Wearable devices are an unobtrusive, cost-effective means of continuous ambulatory monitoring of chronic cardiovascular diseases. However, on these resource-constrained systems, electrocardiogram (ECG) processing algorithms must consume minimal power and memory, yet robustly provide accurate physiological information. This work presents REWARD, the Relative-Energy-based WeArable R-Peak Detection algorithm, which is a novel ECG R-peak detection mechanism based on a nonlinear filtering method called Relative-Energy (Rel-En). REWARD is designed and optimized for real-time execution on wearable systems. Then, this novel algorithm is compared against three state-of-the-art real-time R-peak detection algorithms in terms of accuracy, memory footprint, and energy consumption. The Physionet QT and NST Databases were employed to evaluate the algorithms' accuracy and robustness to noise, respectively. Then, a 32-bit ARM Cortex-M3-based microcontroller was used to measure the energy usage, computational burden, and memory footprint of the four algorithms. REWARD consumed at least 63% less energy and 32% less RAM than the other algorithms while obtaining comparable accuracy results. Therefore, REWARD would be a suitable choice of R-peak detection mechanism for wearable devices that perform more complex ECG analysis, whose algorithms require additional energy and memory resources.**

***Index Terms*— Wearable devices, Resource-constrained embedded systems, ECG, real-time R-peak detection, Ultra-low power devices.**

I. Introduction

Cardiovascular diseases (CVD) are the primary cause of death in the United States and account for 17% of national health expenditures. These ubiquitous, costly diseases currently affect 36.9% of the US population. By 2030, this percentage is projected to rise to 40.5% and cost the US health care system over $818 billion [1]. Wearable technologies are emerging solutions that unobtrusively acquire patients' physiological data, which enables remote CVD monitoring and diagnosis, reduces hospitalization costs, expands patients' mobility, and improves their quality of life [2].

However, the need for wearable, ultra-low-power, and low-cost wireless sensors imposes several design constraints. The sensor nodes must process data in real-time with minimal delays to provide timely assistance in medical emergencies [3]. Additionally, to perform long-term monitoring, they must maximize battery lifetime and therefore use an ultra-low-power microcontroller (MCU). A wearable wireless sensor node typically contains few kilobytes of memory, has a MCU running at a maximum clock speed between 8 and 32 MHz, and usually does not include hardware support for division and floating-point operations. For example, the SmartCardia INYU wearable ECG monitor [4] uses the 32-bit ARM Cortex-M3 MCU that operates at a maximum frequency of 32 MHz and has 48 of kB RAM and 384 kB of Flash. Consequently, the algorithms employed on these sensor nodes must consume minimal RAM, Flash, and energy, as well as limit power-hungry data transmission [5].

To perform CVD monitoring and diagnosis, wearable devices can measure the electrocardiogram (ECG) signal. Its main wave, which includes the R-peak, describes the electrical activity of the heart during a ventricular contraction. The frequency of its occurrence, i.e. heart rate (HR), provides valuable medical information [6]. R-peak detection is essential to more complex algorithms that screen for serious medical conditions, such as myocardial infarction [7] and atrial fibrillation [8]. Current real-time R-peak detection algorithms employ techniques such as signal derivative analysis [9], adaptive thresholds and parameters [10], and variations of the Wavelet Transform [11]–[13]. These algorithms have been widely compared in terms of R-peak detection accuracy [14]. However, few of these works have performed a complete study on the energy and memory footprint trade-offs of the algorithms when implemented on resource-constrained real-time systems [5], [9], [13]. Furthermore, each work tests its proposed algorithm on a different hardware processor or simulator platform, which makes comparative assessment of the algorithms difficult.

Few works in the literature have compared the feasibility of implementing different real-time R-peak detection techniques on embedded systems. Braojos et al. assess the accuracy of a morphological derivative-based versus a wavelet-based ECG delineation algorithm and measure their memory footprints [5]. Elgendi et al. compare a variety of algorithms in terms of robustness to noise and qualitatively assess their numerical efficiency [14]. Nevertheless, to the best of our knowledge, no existing work compares a wide variety of real-time R-peak detection algorithms in terms of accuracy, robustness, memory footprint, and energy consumption using the same hardware platform.

In this work, we design and optimize REWARD, the

*This work has been partially supported by the MyPreHealth (grant no. 16073) project funded by Hasler Stiftung, and by the ONR-G (Award Grant No. N62909-17-1-2006).

[1]L. Orlandic, E. de Giovanni, A. Arza, and D. Atienza are with the Embedded Systems Laboratory (ESL) of the Swiss Federal Institute of Technology (EPFL), 1015 Lausanne, Switzerland.

[2]S. Yazdani and J.M. Vesin are with the Applied Signal Processing Group of EPFL.

Relative-Energy-based WeArable R-Peak Detection algorithm, which is a novel real-time R-peak detection mechanism based on a nonlinear filtering method called Relative-Energy [15]. We implement this method in real-time and design a peak detection procedure to complement it. Then, we optimize the REWARD algorithm specifically for use on resource-constrained systems, maximizing its accuracy while minimizing its energy and memory footprints. We compare this new algorithm against three state-off-the-art real-time R-peak detection algorithms and show that it performs comparably in terms of accuracy while consuming less energy and memory. In addition, we address the need for a comprehensive comparison of well-established approaches for real-time R-peak detection in wearable systems. The accuracy and robustness of these four algorithms are tested using standard databases, and their energy consumption, computational burden, and memory footprint are measured on the same hardware platform.

II. Related Work on Real-Time R-Peak Detection Algorithms

Many R-peak detection algorithms have been developed, but relatively few are designed for real-time implementation on ultra-low power embedded systems. Three algorithms that meet these design constraints are the Pan-Tompkins (PT) algorithm [10], a wavelet transform delineation (WTD) [13], and a derivative-analysis-based delineation (DAD) [9]. These algorithms are well-known for their high accuracy and previous implementation on real-time wearable systems. They also represent diverse R-peak detection methodologies.

An ECG R-peak detection algorithm generally consists of a two-step procedure, as depicted in Fig. 1. First, a preprocessing step may include a filtering method to suppress noise in the ECG excerpt, as well as specific methods to highlight the principal components of the ECG waveform. Then, a peak detection procedure locates the R-peaks.

This section provides an overview of the aforementioned three state-of-the-art R-peak detection algorithms, detailing the real-time implementation of each of them. All filters and algorithms are implemented in the C programming language using primarily 16-bit integer arithmetic, a sampling frequency of 250 Hz, and minimal buffer sizes to ensure a fair energy and memory consumption comparison.

A. Real-time Preprocessing Methods

Every algorithm contains its own preprocessing method for filtering and highlighting the principal components of the ECG waveform. These methods are often used to remove baseline wander, high frequency noise, and muscle artifacts. Two frequently-used real-time filtering approaches, a morphological filter (MF) [16] and a band-pass filter (BPF) [9], are analyzed to determine the benefits they provide to the accuracy of each algorithm versus the drawbacks of additional energy and memory consumption. The MF performs two operations, opening and closing, which respectively remove the peaks and valleys of the signal. These operations produce the baseline, which is then subtracted from the original signal to remove any baseline drift. An opening window of $0.2\,\mathrm{s}$ is used, along with a closing window of $0.3\,\mathrm{s}$. The filter is coded in C in real-time and introduces a delay of $0.49\,\mathrm{s}$. The FIR filter design of the BPF, with a passband of 0.3-40 Hz and order 32, is done offline, which suppresses both the baseline and high frequency noise. It is coded in C using mainly 16-bit integer operations and implemented using symmetry criteria for the filter coefficients, as described in [9] .

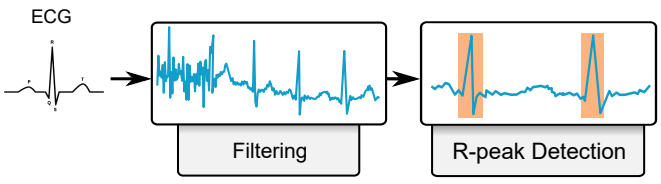

Fig. 1. Block diagram of a general real-time R-peak detection algorithm

B. Real-Time R-Peak Detection Algorithms

1) The Pan-Tompkins Algorithm (PT): Pan and Tompkins proposed a real-time ECG R-peak detection algorithm in 1985 [10], which has since been widely used in the literature. The 4-step preprocessing method of the algorithm consists of a 5-12 Hz bandpass filter, a derivative of the filtered signal, squaring the derivative to amplify the QRS complex, and a moving-window integrator. Then, the peaks of the ECG signal are identified by applying adaptive thresholds on the filtered and integrated signals. These thresholds use the past eight peak amplitudes and R-R intervals to identify peaks and ensure that they have an R-R interval above $0.2\,\mathrm{s}$.

The initial delay and buffer size of this algorithm include $5\,\mathrm{s}$ of signal, namely, $2\,\mathrm{s}$ to initialize the peak detection thresholds, plus $3\,\mathrm{s}$ to compute the initial R-R interval and search back for missed peaks. The algorithm is implemented in real-time in C using primarily 16-bit integer arithmetic.

2) Wavelet Transform Delineation (WTD): A widely-implemented algorithm that performs full ECG delineation is the Wavelet Transform (WT). The WT of a signal is proportional to the derivative of the signal with a smoothing impulse response at different scales. Therefore, the zero-crossings of the WT function correspond to the local maxima or minima of a signal at a given scale, and the peaks correspond to its maximum slopes. Five dyadic scales are chosen (i.e. 2^1 to 2^5), since most of the ECG signal energy lies within these scales [11]. Once the WT is applied to the signal, the R-peaks are identified as the zero-crossings that are common across scales 2^1 through 2^4, and which are preceded by a positive peak and followed by a negative peak.

The WTD algorithm analyzed in this work is the optimized, single-lead, offline ECG delineation algorithm presented in [11], implemented by [12], and extended in [13]. This delineator detects all characteristic points of an ECG waveform using a quadratic spline wavelet transform. The algorithm is implemented in real-time with a buffer size of $1.024\,\mathrm{s}$. It is coded in C using primarily 16-bit integer operations.

3) Derivative Analysis Delineation (DAD): Recently, Bote et al. proposed a derivative-based, low-complexity algorithm for ECG delineation [9], which has a modular design. It can perform either full ECG delineation, or operate in a low-power mode that only detects R-peaks, the latter of which is analyzed in this work. First, the signal is preprocessed with a 14 Hz lowpass filter. Next, the first and second derivatives in a 2 s window are analyzed to identify the R-peaks as points at which 1) there is a zero crossing of the first derivative, 2) the RR-interval is higher than 0.25 s, and 3) the magnitude of the second derivative exceeds 0.33x the average of the past five minimum/maximum window values. This algorithm is implemented in real-time with a buffer length of 2 s. It is coded using primarily 16-bit integer arithmetic.

III. REWARD Design and Real-Time Optimization

The REWARD algorithm includes two main components. Firstly, the ECG signal is preprocessed to highlight its peaks and suppress its baseline using the Relative Energy (Rel-En) nonlinear filtering method proposed in [15]. Secondly, the R-peaks are detected from the filtered signal. In this work, we have developed the first real-time implementation of the Rel-En preprocessing method and optimized it for use on ultra-low power wearable systems. Then, it is paired with the R-peak detection procedure that we designed, all of which is described next.

A. Rel-En Preprocessing Implementation and Optimization

The Rel-En preprocessing method considers the energies of a long sliding window l_{win} (0.95 s) and a short sliding window s_{win} (0.14 s), both centered at sample *n*. l_{win} describes the long-term behavior of the ECG signal *x*, while s_{win} can capture an R-peak occurrence, resulting in a larger short-term energy than when no peak occurs. The ratio between the energies of these windows, the coefficient *c(n)*, is multiplied by *x(n)* resulting in a signal x_{RE}, in which the peaks are amplified, as depicted in (1) and (2). The parameter *w* in (1) represents a Hamming window function, and $p = 2$.

$$c(n) = \frac{\sum_{i=n-swin/2}^{n+swin/2} |x(i)|^p}{\sum_{j=n-lwin/2}^{n+lwin/2} |w(j) \times x(j)|^p} \quad (1)$$

$$x_{\text{RE}}(n) = c(n)x(n) \quad (2)$$

In this work, the Rel-En preprocessing method is ported from MATLAB to C and optimized for single-lead, real-time ECG R-peak detection on resource-constrained wearable systems. First, the computation is changed from floating point arithmetic (32-bit) to short integer (16-bit) to consume less energy and memory on the MCU. Subsequently, the Rel-En method is implemented using circular buffers to minimize RAM usage and processing delays. It considers a centered sliding window of 0.95 s of the ECG signal for preprocessing to obtain the x_{RE} signal.

Finally, the Rel-En preprocessing method is simplified to reduce its energy consumption. In [15], the Hamming window function is used to smooth the long-term energy of each coefficient *c(n)*, which represents a significant number of operations performed per coefficient output. Specifically, suppressing this step reduces the computational load by a factor of $N=fs*l_{win}$ where fs is the sampling frequency, so for each coefficient there must be N calculated Hamming window coefficients and N multiplications. In order to reduce the algorithm's complexity and consequent energy consumption, we removed the Hamming window function from the long-term window calculation, i.e. $w(j) = 1$, without any significant cost to the algorithm's performance, which is shown in Section V-A.

B. REWARD Peak Detection

To complete the REWARD algorithm, we paired the Rel-En method with a real-time peak detection procedure that is both adaptable and computationally simple. Our algorithm is based on the hysteresis comparator [17], and several optimizations are applied to improve its detection accuracy. The R-peaks are detected using a window of 1.75 s. Consequently, the initial delay of this algorithm is $(0.95/2+1.75)$s. For each R-peak detection window, the algorithm first checks if the dominant peak is positive or negative. Then the hysteresis comparator is applied to identify possible peaks. Next, the algorithm selects the peaks that meet a set of criteria, such as representing a HR between 30-240 BPM and a peak width in the same range as that of the previously-selected R-peak.

The first step of the algorithm is a real-time variation of the negative peak identification procedure described in [18]. Within the 1.75 s peak detection window, if the minimum amplitude relative to the mean value of x_{RE} is greater than 70% of the maximum amplitude relative to the mean, it is presumed that the R-peak is negative.

Next, the hysteresis comparator method is implemented to identify the locations of the peaks based on two adaptive thresholds, as illustrated in Fig. 2. Segments in which the signal goes above the upper threshold, Th_{upper}, and subsequently below the lower threshold, Th_{lower}, are considered active peak regions. The maximum of the signal between these two points is considered a peak candidate. The purpose of having two thresholds is to eliminate false R-peak candidates due to high-frequency oscillations or false peaks at the threshold boundary. This procedure is less complex than initially searching for all local maxima in a window and then applying a threshold to select the peaks, since false peaks

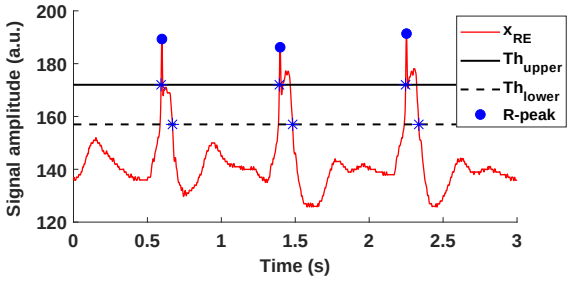

Fig. 2. The hysteresis comparator detects the peaks of sel102 of the QTDB.

often exist at the threshold boundary due to signal noise.

In order for the thresholds to adapt to changes in the signal's amplitude from one window to the next, they are defined as shown in Lines 2-4 of Algorithm 1. Avg and Max denote the mean and maximum values in one window of $\mathrm{x_{RE}}$. To determine the optimal thresholds, the unfiltered REWARD algorithm was tested on the QTDB with every combination of threshold constants T_u and T_l. Thresholds that are too high result in missed peaks and a low sensitivity, whereas thresholds that are too low result in a low positive prediction value. Thus, after a careful experimental validation, we conclude that the pair of threshold constants producing the highest G-mean (98.61%) is T_u = 0.4 and T_l = 0.15.

False peaks often occur due to enlarged T-waves that exceed Th_{upper}. To solve this issue, we employed the physiological characteristics of the R-R interval and the T-wave: The R-R interval lies between 0.25-2 s, while the T-wave is typically wider than the QRS complex and occurs within 0.5 s after the R-peak. Accordingly, we implemented the peak selection procedure described in Lines 5-18 of Algorithm 1. First, we compute the peak widths in Line 5 and check the corresponding R-R interval. The peak width is the time interval between the peak onset and offset, determined by Th_{upper} and Th_{lower}. If the R-R interval is longer than 0.5 s, the current peak is kept, and if it is less than 0.25 s, the peak is discarded. Finally, if the R-R interval is between 0.25-0.5 s, the widths of both peaks are compared to determine if both peaks are valid or one is an enlarged T-wave. If one peak is more than 35% wider than the other, the peak is discarded. This percentage value was chosen empirically (cf. Section IV-A for details about the used databases), considering the physiology of the ECG waveform.

Our real-time peak detection procedure is more adaptable and computationally simple than the peak detection proposed in [15]. In particular, in the original offline method, the entire signal was first normalized based on its minimum and maximum values, the mean of the signal was subtracted, and then a fixed threshold was applied. This normalization and mean subtraction introduces significant computational complexity, since each function performs at least one mathematical operation on every sample in the signal. In contrast, we implemented the hysteresis-based procedure described in Algorithm 1 whose peak detection thresholds, $\mathrm{Th_{upper}}$ and $\mathrm{Th_{lower}}$, are adapted based on the average and maximum values of each peak detection window.

Algorithm 1 R-peak selection within a long window

1: **function** PEAKSEL($\mathbf{x_{RE}}, R_{last}, T_u, T_l$)
2: $(Avg, Max, Min) = statSigInWindow(\mathbf{x_{RE}})$;
3: $Th_{upper} = Avg + T_u \times |Avg - Max/Min|$;
4: $Th_{lower} = Avg + T_l \times |Avg - Max/Min|$;
5: $\mathbf{pks^{loc,wid}} = findPks(\mathbf{x_{RE}}, Th_{upper}, Th_{lower})$;
6: **for** $n = 2 : length(\mathbf{pks^{loc}})$ **do**
7: **if** $0.25s < pks_n^{loc} - R_{last}$ **then**
8: $discardPeaks(pks_n^{loc})$;
9: **else if** $pks_n^{loc} - R_{last} > 0.5s$ **then**
10: $keepPeaks(pks_n^{loc})$;
11: **else**
12: **if** $0.65 < pks_n^{wid}/pks_{n-1}^{wid} > 1.35$ **then**
13: $discardPeakWithLargerWidth()$;
14: **else**
15: $keepPeaks(pks_n^{loc})$;
16: **end if**
17: **end if**
18: **end for**
19: **end function**

IV. Experimental Setup

A. Standard Databases and Metrics for Accuracy Evaluation

In order to quantify the detection accuracy of the algorithms, we use two public databases provided by Physionet [19]. The first is the QT Database (QTDB) [20], which consists of 105 two-channel ECG Holter recordings with a wide variety of ECG morphologies, including various arrhythmias and sudden death cases. Each 15-minute recording contains two leads, sampled at 250 Hz. At least 30 beats of each recording have been manually annotated by an expert to identify the locations of several ECG fiducial points. Out of the 3622 annotated beats in the database, a total of 3587 annotations include the R-peak.

Next, to test the algorithm' robustness to noise, the Noise Stress Test Database (NSTDB) [21] is used. The NSTDB consists of two clean 30-minute-long signals of the MIT-BIH Arrhythmia Database (MITDB), to which five varying amounts of noise were added such that the Signal-to-Noise Ratio (SNR) decreased by 6 dB for each noise addition. The signals were originally sampled at a frequency of 360 Hz and re-sampled to 250 Hz to maintain consistency when testing the algorithms. The original R-peak annotations from the MITDB are used as ground-truth.

To assess the performance of the analyzed algorithms, the true positives (TP), false positives (FP), and false negatives (FN) of the detected peaks were computed using 150 ms of tolerance from the annotated peak [22]. Accordingly, we use the performance metrics of sensitivity (SE), positive prediction value (PPV), geometric mean (G-mean), and detection error rate (DER). Finally, the mean error (m) and its standard deviation (σ) are measured.

B. Hardware Platform for Real-time Implementation

To measure the energy consumption of the state-of-the-art and proposed algorithms and filters on resource-constrained wearable devices, they were implemented on the Silicon Labs EFM32 Leopard Gecko 32-bit MCU [23]. This board contains a 48 MHz ARM Cortex-M3 CPU, 32 kB of RAM, and 256 kB of flash memory. These hardware specifications are similar to those found on wearable ECG sensor nodes; the aforementioned INYU device uses the same processor. Every algorithm is run using the -O3 compiler optimization level, which is the best optimization tolerated by the EFM32. The board, along with its development environment Simplicity Studio, includes an energy profiler that measures the

TABLE I

R-PEAK DETECTION RESULTS ON THE QT DATABASE ON 3587 EVALUATED BEATS

Methods	**REWARD**			**PT [10]**			**WTD [13]**			**DAD [9]**		
Filters	**NF**	**MF**	**BPF**	**NF**	**MF**	**BPF**	**NF**	**MF**	**BPF**	**NF**	**MF**	**BPF**
TP	3550	3563	3546	2694	3281	2602	3574	3581	3576	3445	3421	3443
FP	63	26	53	129	32	209	0	0	0	4	3	2
FN	37	24	41	833	306	985	13	6	11	142	166	144
SE (%)	98.97	99.33	98.86	76.38	91.47	72.54	99.64	99.83	99.69	96.04	95.37	95.99
PPV (%)	98.26	99.28	98.53	95.43	99.03	92.56	100.0	100.0	100.0	99.88	99.91	99.94
G-mean (%)	98.61	99.30	98.69	85.38	95.18	81.94	99.82	99.92	99.85	97.94	97.62	97.94
DER (%)	2.74	1.38	2.58	26.31	9.34	31.45	0.36	0.17	0.31	4.07	4.71	4.07
m$\pm\sigma$(ms)	9.5±4	9.3±3.5	9.7±4.3	100±9.6	13±4.4	100±10	11±3.8	7.5±3.3	11±3.7	11±3.3	7.5±3.2	7.6±3.2

execution time and total energy consumed by the algorithms within a specified execution window. The board is placed into a sleep mode before and after each algorithm execution to ensure that the energy consumption and execution time only reflect those of the algorithm. The Simplicity Studio development environment also measures the amount of memory, both RAM and Flash, consumed by each algorithm. Flash memory permanently stores the variables and instructions of a program, whereas RAM performs run-time operations on the variables it retrieves from Flash.

The algorithms were tested independently on the EFM32 board, considering 12 seconds (3000 samples) of 4 different recordings of the QT Database, which were chosen to contain varying degrees of R-peak detection accuracy.

V. EXPERIMENTAL RESULTS

A. Accuracy and Energy Impact of REWARD Real-Time Design and Optimization

As REWARD was designed for use on real-time, ultra-low power systems, several optimizations were performed to minimize its energy and memory footprints while increasing its R-peak detection accuracy. This involved multiple changes to the original Rel-En preprocessing method proposed in [15], as well as our design of a paired peak detection algorithm.

First, the original Rel-En preprocessing method was ported from MATLAB to C, changed from 32-bit floating point to 16-bit integer arithmetic, and run on the EFM32 using the -O3 optimization level. We then optimized the Rel-En method by removing the Hamming window function $w(j)$ in Equation (1), as described in Section III-A. Multiplying each value of the long window by its corresponding Hamming window coefficient significantly increased the number of operations performed per coefficient output. Consequently, the Hamming window function consumed 96% of the total energy of REWARD. This optimization only decreased the unfiltered REWARD G-mean by 0.13%, but dramatically reduced the energy consumption by more than three orders of magnitude (1316x).

The original algorithm described in [15] was tested on the QTDB with an offline 4-40 Hz BPF and produced a G-mean of 99.97%. The final G-mean of our optimized, real-time REWARD algorithm with a real-time 0.3-40 Hz BPF was only 1.28% lower than that of the original algorithm.

B. R-peak Detection Accuracy

The accuracy results of the four algorithms are displayed in Table I, which lists the performance with no additional filters "NF", and with a MF or BPF applied. Moreover, the algorithms' performance in terms of G-mean is summarized in Fig. 3. First, the benefit of the two filters was assessed. Due to their design, using a BPF alongside the REWARD, WTD, DAD, and PT algorithms did not significantly improve their accuracy; their G-means increased by 0.08% or less. This is because the REWARD preprocessing amplifies the peaks and the hysteresis comparator counteracts the effects of high frequency noise. Similarly, PT and DAD use aggressive lowpass filters, while their use of derivatives mitigates baseline drift. MF, on the other hand, resulted in higher G-mean increases in the algorithms. The use of MF increased PT's G-mean by 11.5%, that of REWARD by 0.70%, and that of WTD by 0.10%. Overall, filtering does not significantly increase the accuracy of these algorithms, but it is still advisable for medical applications in which precise results are indispensable.

Analyzing the SE and PPV of the algorithms reveals that REWARD produced high accuracy results both with and without filtering. In terms of SE, REWARD paired with MF was only 0.50% less than WTD with MF. In terms of PPV, REWARD with MF was 0.72% lower than WTD with and without filtering. The four achieved high PPVs with MF: over 99.0%. WTD produced the highest G-mean, with the unfiltered G-means of REWARD and DAD trailing that of WTD by only 1.2% and 1.89%, respectively. With MF applied, the REWARD G-mean was only 0.62% lower than that of WTD. Furthermore, REWARD produced comparable accuracy results to both WTD and other state-of-the-art algorithms, and MF further increased its performance [14].

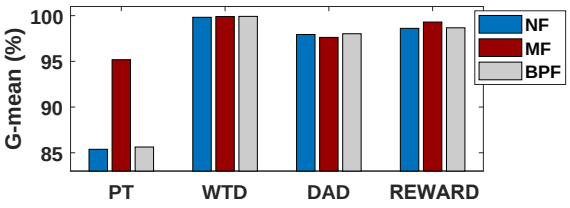

Fig. 3. The G-means of the four R-peak detection algorithms with their respective filters applied on the QTDB.

TABLE II

ENERGY PROFILE AND MEMORY FOOTPRINT OF R-PEAK DETECTION ALGORITHMS AND FILTERS

		Computational burden (%)	Total Energy (mJ)	RAM (kB)	Flash (kB)
Filtering	MF	0.95	5.52	1.88	15.8
	BPF	0.29	1.92	0.228	7.50
R-peak detection	PT	16.4	90.6	28.6	16.7
	WTD	0.56	3.30	6.26	23.10
	DAD	0.37	2.49	2.34	8.51
	REWARD	0.16	0.916	1.58	12.20

C. Robustness to Noise Evaluation

REWARD, WTD, and DAD were tested on the NSTDB to determine how signal quality degradation affects their R-peak detection accuracy. In Fig. 4, the SNR of each signal in the NSTDB is plotted against the corresponding G-mean of each algorithm. For signal 118, G-mean of REWARD was an average of 3.77% lower than that of DAD for the three highest SNRs, and an average of 7.67% lower overall. REWARD performed poorly on signal 119, however, due to overly high hysteresis thresholds for this particular ECG morphology. The REWARD G-mean decreased fairly consistently with each 6 dB drop, while the DAD and WTD G-means stayed nearly the same for SNRs between 12 and 24 dB and then dropped sharply. These results indicate the algorithms' behavior in the presence of noise, which can occur in daily environments using a wearable sensor.

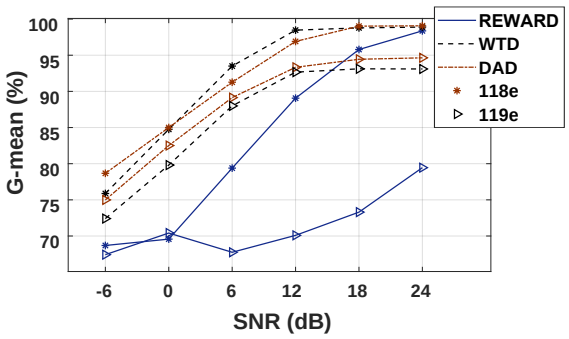

Fig. 4. The accuracy results of WTD, DAD and REWARD algorithms with MF and BPF on the NSTSB signals with varying SNRs.

D. Energy Consumption and Memory Footprint Assessment

The energy metrics of the algorithms and filters on the four 12-second-long (3000 samples) QTDB signals, when run on the EFM32 MCU using -O3 compiler optimization, are averaged and presented in Table II. First, the table displays the computational burden, which is the total code execution time divided by the total ECG signal acquisition time (i.e. 12 s). This metric indicates what percentage of the ECG sampling period (i.e. 4 ms) is spent performing the algorithm's functions. The table also shows the total energy that it takes to process the 12 s. REWARD consumed the least energy: only 916 µJ, which corresponds to 305 nJ per processed sample. DAD and WTD consumed 2.72x and 3.6x more energy than REWARD, respectively. Furthermore, when the code was run using no compiler optimizations (-O0), DAD consumed 2.29x more energy than REWARD, indicating that increased optimization levels lead to a comparatively better performance for REWARD. PT consumed over 98x as much energy as REWARD.

Fig. 5 depicts the energy consumed by each algorithm when it processes 12 s of data from the QTDB, when both filter are applied. It shows that three of the four algorithms (REWARD, WTD, and DAD) had comparable energy consumptions, while PT consumed much more.

Table II also displays the memory footprints of the algorithms and filters. REWARD consumed by far the least amount of RAM; 1.51x less than DAD. WTD consumed 3.96x more RAM than REWARD, while PT consumed nearly all of the available 32kB of RAM. REWARD and DAD consumed less Flash than the other algorithms, followed by PT. Finally, WTD consumed 1.89x more Flash than REWARD. Overall, REWARD and DAD were the most memory-efficient algorithms.

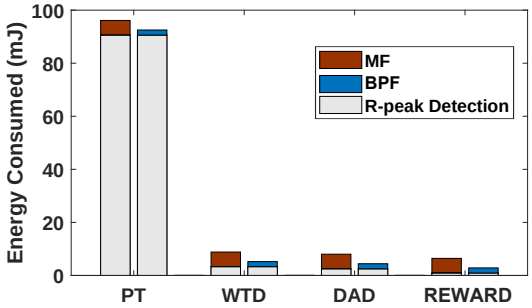

Fig. 5. The energy consumed by four R-peak detection algorithms with and without filters on 12 s of samples from the QTDB.

E. Energy and Memory vs Accuracy Analysis

Fig. 6 displays the energy and RAM consumption of each algorithm-filter combination, except those of PT, plotted against their G-means from Table I. This figure shows that without filters, REWARD achieves the lowest energy and RAM footprint, while WTD has the highest accuracy but also the highest RAM and energy consumption. It also shows that applying a MF to REWARD increases both the accuracy and energy consumption, whereas applying filters to WTD and DAD increases their energy consumption without a significant impact on their accuracies.

REWARD has the lowest energy consumption and a small memory footprint, but its current implementation is less robust than that of WTD and DAD. WTD produces the most

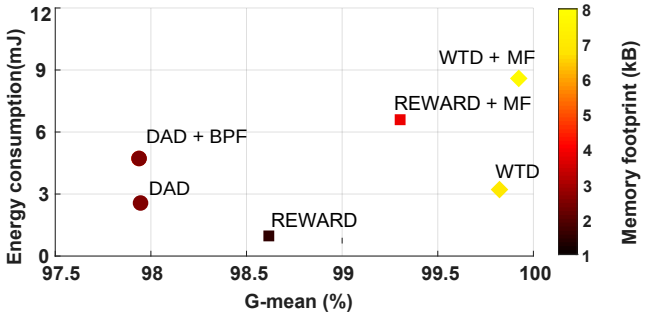

Fig. 6. The energy and RAM consumed by the R-peak detection algorithms with their respective filters versus their accuracies tested on the QTDB.

accurate, robust results, and consumes little energy, but its memory footprint is significantly higher than that of DAD and REWARD. Similarly, DAD exhibits the best robustness to noise and lowest Flash memory consumption, but its SE when tested on the QTDB is low compared to the WTD and REWARD, which implies that it does not correctly identify R-peaks of various ECG morphologies present in the QTDB. Finally, for the filters, Table II shows that BPF is more energy-efficient than MF, using 65% less energy, whereas Table I shows that MF leads to higher G-means. Though the use of MF only slightly improved the accuracy results of the four algorithms, it should still be considered for processing noisier signals from wearable sensors.

While each algorithm's specific computational burden and energy consumption may vary depending on the selected hardware, testing all four algorithms on the same ARM Cortex-M3-based platform provides a comparative analysis of the algorithms relative to each other. This assessment enables wearable technology designers to select the algorithm and filter that fits the accuracy, energy consumption, and memory footprint constraints of their device. Moreover, our results indicate that the new optimized REWARD algorithm is the optimal choice for wearable medical devices in which on-board machine learning is necessary, since its low energy consumption and memory footprint leave room for additional processing capabilities. For example, using REWARD for R-peak detection would leave RAM available for complex cardiological analysis, such as the HR variability analysis algorithm described in [24] (which consumes 8 kB of RAM), and the atrial fibrillation detection algorithm in [8] (which uses 2 kB of RAM).

VI. Conclusion

Wearable technologies provide accurate, energy efficient means of health and pathology monitoring. This work has detailed the real-time implementation and optimization, in the context of resource-constrained wearable devices, of the low-complexity Rel-En preprocessing method, as well as the design of a novel R-peak detection algorithm to complement it. Furthermore, this work has addressed the need for a comprehensive comparison of three state-of-the-art real-time R-peak detection algorithms (PT, WTD, DAD), as well as the REWARD algorithm, to determine each algorithm's feasibility of implementation on ultra-low power real-time embedded systems. REWARD was the most efficient in terms of energy and memory compared with state-of-the art R-peak detection algorithms. It used at least 63% less energy and 32% less RAM than the other algorithms, and produced comparable accuracy results. This new and optimized real-time algorithm we have proposed can therefore be implemented on ultra-low power wearable devices to maximize battery lifetime and provide sufficient RAM for more complex cardiovascular anomaly detection algorithms.

References

[1] P. A. Heidenreich *et al.*, "Forecasting the future of cardiovascular disease in the United States: A policy statement from the American Heart Association," *Circulation*, vol. 123, no. 8, 2011.

[2] H. Alemdar *et al.*, "Wireless sensor networks for healthcare: A survey," *Comput. Netw.*, vol. 54, no. 15, 2010.

[3] M. Patel *et al.*, "Applications, challenges, and prospective in emerging body area networking technologies," *IEEE Trans. on Wireless Commun.*, vol. 17, no. 1, feb 2010.

[4] "Smartcardia," http://www.smartcardia.com/.

[5] R. Braojos *et al.*, "Embedded real-time ECG delineation methods: A comparative evaluation," in *Proc. of BIBE*. IEEE, nov 2012.

[6] B. U. Köhler *et al.*, "The principles of software QRS detection," *IEEE Eng. Med. Biol. Mag.*, vol. 21, no. 1, 2002.

[7] D. Sopic *et al.*, "Real-time event-driven classification technique for early detection and prevention of myocardial infarction on wearable systems," *IEEE Trans. Biomed. Circuits Syst*, vol. 12, no. 5, pp. 982–992, Oct 2018.

[8] F. Rincn *et al.*, "Automated real-time atrial fibrillation detection on a wearable wireless sensor platform," in *EMBS*. IEEE, Aug 2012.

[9] J. M. Bote *et al.*, "A modular low-complexity ECG delineation algorithm for real-time embedded systems," *IEEE J. Biomed. Health Inform.*, vol. 22, no. 2, mar 2018.

[10] J. Pan *et al.*, "A simple real-time QRS detection algorithm," in *EMBS*, vol. 4, no. 3. IEEE, 1985.

[11] J. P. Martínez *et al.*, "A wavelet-based ECG delineator: evaluation on standard databases." *IEEE Trans. Biomed. Eng.*, vol. 51, no. 4, apr 2004.

[12] N. Boichat *et al.*, "Wavelet-Based ECG Delineation on a Wearable Embedded Sensor Platform," in *Proc. of BSN*. IEEE, jun 2009.

[13] F. Rincón *et al.*, "Development and evaluation of multilead wavelet-based ECG delineation algorithms for embedded wireless sensor nodes," *IEEE Trans. Inf. Technol. Biomed*, vol. 15, no. 6, nov 2011.

[14] M. Elgendi *et al.*, "Revisiting QRS detection methodologies for portable, wearable, battery-operated, and wireless ECG systems," *PLoS ONE*, vol. 9, no. 1, 2014.

[15] S. Yazdani *et al.*, "A Novel Short-Term Event Extraction Algorithm for Biomedical Signals," *IEEE Trans. Biomed. Eng.*, vol. 65, no. 4, apr 2018.

[16] Y. Sun *et al.*, "ECG signal conditioning by morphological filtering," *Comput. Biol. Med.*, vol. 32, no. 6, nov 2002.

[17] X. Qian *et al.*, "A low-power comparator with programmable hysteresis level for blood pressure peak detection," in *TENCON 2009 - 2009 IEEE Region 10 Conference*, Jan 2009, pp. 1–4.

[18] S. Yazdani *et al.*, "Extraction of QRS fiducial points from the ECG using adaptive mathematical morphology," *Digital Signal Process.*, vol. 56, sep 2016.

[19] A. L. Goldberger *et al.*, "PhysioBank, PhysioToolkit, and PhysioNet : Components of a New Research Resource for Complex Physiologic Signals," *Circulation*, vol. 101, no. 23, 2000.

[20] P. Laguna *et al.*, "A database for evaluation of algorithms for measurement of QT and other waveform intervals in the ECG," in *CinC*. IEEE, 1997.

[21] G. M. Friesen *et al.*, "A Comparison of the Noise Sensitivity of Nine QRS Detection Algorithms," *IEEE Trans. Biomed. Eng.*, vol. 37, 1990.

[22] AAMI, *Testing and reporting performance results of cardiac rhythm and ST-segment measurement algorithms*. The Association, 2008.

[23] "EFM32 Leopard Gecko 32-bit Microcontroller," https://www.silabs.com/products/mcu/32-bit/efm32-leopard-gecko.

[24] G. Karakonstantis *et al.*, "Low complexity spectral analysis of heart-rate-variability through a wavelet based fft," in *CinC*, Sep. 2012, pp. 285–288.