IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.Doi Number

Factor Analysis for the Performance Impacts of Real-Time Ventricular Fibrillation Detection on Microcontroller

Jungyoon Kim[1], Jaehyun Park[2], and Misun Kang[3]

[1]Department of Computer Science, Kent State University, Kent, OH 44242 USA
[2]Department of Industrial and Management Engineering, Incheon National University (INU), Incheon, Republic of KOREA
[3] Department of Computer Software Engineering, Soonchunhyang University, Asan-si, Chungcheongnam-do 31538, Republic of KOREA

Corresponding author: Misun Kang (e-mail: ms.kang@sch.ac.kr).

This work was supported by the Soonchunhyang University Research Fund. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean Government (MSIT) (No. RS-2022-00165681).

ABSTRACT Ventricular fibrillation (VF) is one of the most serious cardiovascular diseases that must be detected reliably and dealt with in a timely manner to improve the chance of patient survival from heart attacks. Early research focused on developing effective algorithms for VF detection; while most of the evaluations have been conducted offline with prefiltered data sets, practical application requires these tests to be performed in real time. Because there are many factors that may impact detection effectiveness, it is important to understand the impact of factors that improve detection accuracy. In this study, we developed an integrated simulated environment using IAR Embedded Workbench software to build an embedded system using a MSP430 microcontroller and Visual studio tool for S/W build; we then used this system to conduct real-time experiments for evaluating five lightweight VF detection algorithms and to examine factors that may impact their performance in terms of sensitivity, specificity, positive-predictivity, accuracy and computational time. The results were cross-validated using a prototype of a wearable Electrocardiogram (ECG) system developed by this study. The study showed that 1) the chosen detection algorithm, data filtering, and window size all have a significant impact on the performance of real-time VF detection; among these, the detection algorithm had the greatest impact so it must be carefully selected; 2) it is important to select the proper threshold value that affects tradeoffs in performance metrics. Among the five algorithms that this study evaluated, the Time Delay (TD) algorithm outperformed the others independent of window size or filtering method. This paper analyzed seven factors and examined the impacts of three of them on real-time VF detection. Based on analysis, the scaling process is very important, and a good detection method will reduce the degree of impact to a minimum level; otherwise, a filtering method should be considered. Considering the tradeoff between robustness and efficiency, TD is preferable because detection accuracy and robustness are more critical.

INDEX TERMS Heart attack; ventricular fibrillation (VF); factor analysis; real time; VF detection;

I. INTRODUCTION

According to the World Health Organization (WHO), cardiovascular diseases (CVDs) are the number one cause of death worldwide [1], and among CVDs, Ventricular Fibrillation (VF) is one of the most critical life-threatening cardiac arrhythmia diseases. Once a patient has suffered a VF attack, accurate detection and quick first aid are essential for improving the chance of survival.

Previous research on VF detection primarily has focused on two main topics: 1) developing and evaluating the relative performance of detection algorithms [2–19], and 2) developing handheld devices for real-time monitoring [20–23]. Most previous performance studies have been conducted offline using prefiltered data sets, fixed threshold values, and a single time-window size (often 8 s); recently machine/deep learning-based methods have been proposed [10-19]. However, practical application requires these tasks to be performed in real time, and while there are several important factors that may affect detection accuracy, existing research has not fully examined the potential impact of such factors or

system parameters (e.g., threshold values). (See our summary of related work in Section 2.)

Handheld or wearable electrocardiogram (ECG) devices have been introduced as a major solution for next-generation healthcare [21,25]. Most available handheld devices can be used to instantly capture and display only heart signals and basic information (e.g., heartbeat); they normally have limited domain intelligence built in for timely and accurate detection [12,26]. The data must be sent via mobile phone or offline to central computers to be analyzed by an intelligent system or a domain expert, resulting in a major time delay in providing feedback and substantial consumption of battery power. There has been a recent effort using intelligent methods based on a tiny machine learning approach to detect ventricular arrhythmia using a microcontroller [27]. This means that new testing environments and methodologies in the wearable or embedded-system-based health-monitoring area will be required for real-time VF detection.

The main purposes of this study are fourfold: 1) to identify factors critical to real-time VF detection, 2) to examine and select easily-adaptable algorithms with the best performance for handheld devices, 3) to propose an objective approach to determination of system parameters, and 4) to develop an integrated environment using public databases for emulating real-time monitoring and detection processes. Five lightweight VF detection algorithms have been evaluated: Threshold Crossing Interval (TCI) [28], Threshold Crossing Sample Count (TCSC) [6], Time Delay (TD) [5], VF filter (VFF) [29], and Pan & Tompkins (TOMP) [30]. Methods of real-time filtering, data-scaling, and determining the optimal threshold value have been proposed and tested as part of the integrated environment.

This study used complete data sets from the Creighton University (CU) ventricular tachyarrhythmia database [31] to generate a real-time ECG signal and feed it into proposed virtual patients. The performance results were measured in terms of the most popular quality parameters [e.g., sensitivity (*Sn*), selectivity (*Sp*), positive predictivity (*Pp*), and accuracy (*Ac*)], computational time (*Ct*), the receiver operating characteristic (ROC) curve, and the Area under the Curve (*AUC*) [32]. Statistical methods such as the analysis of variance (ANOVA) and paired-t tests were then applied to assess their significance in computational efficiency.

The contributions of this research are threefold:

(1) We have conducted a factor analysis based on the variance of detection algorithms, data filtering and window sizes that affects the performance of real-time VF detection using a low-powered microcontroller.
(2) We have concluded that the TD algorithm outperformed the other algorithms regardless of window size or filtering method.
(3) We proposed a new real-time VF detection testing environment based on the available dataset for a low-powered microcontroller.

This article is an extended work based on Chapter 4 of the author's PhD thesis [33]. It focuses more on statistical analysis of five lightweight VF algorithms used in a real-time environment and discusses various factors involved in real-time VF detection.

II. RELATED WORK

There have been many studies focused on evaluating effectiveness of VF detection algorithms. Table 1 summarizes selected related work and highlights relevant factors. This study reviewed related works according to six factors: the filtering method (used to reduce unnecessary signal (noise) when detecting VF from the raw data), window size (used to segment the signals for processing), detection algorithm (algorithm or classifier for generating feature values), threshold value (decision value for separating the normal signal from VF), the data set used, and performance measures used.

References [7,8] proposed a neural network approach for VF detection and compared its performance with four other conventional algorithms. Study [3] proposed a new algorithm, the Signal Comparison Algorithm (SCA), and compared its performance with five well-known VF and two QRS detection algorithms. Those researchers found that QRS detection algorithms are not suitable for VF detection even if the threshold values are carefully selected. In a follow-up study, Amann et al. developed two new VF detection algorithms, Hilbert transform (HILB) [4] and Time Delay (TD) [5], based on phase space reconstruction (PSR), and evaluated them against four extant algorithms. They concluded that TD is 1.1% more accurate than HILB. Ismail et al. compared five VF detection algorithms and three sets of combined pairs of these VF detection algorithms [9] and concluded that combining VF algorithms with the fine-tuning of critical thresholds can improve accuracy.

Reference [2] proposed a sequential detection algorithm using empirical mode decomposition (EMD), and compared its performance with five other VF detection algorithms. The researchers showed that EMD performed the best, followed by TCSC, TD, HILB, SPEC, and TCI. However, the results may be biased as they aggregated results from three databases, each producing quite different results. Study [6] adapted a time-domain algorithm TCSC from TCI, and compared its performance with seven other VF detection algorithms. The researchers showed that TCSC performed the best, followed by TD and HILB. They also showed that TD performed better than SPEC. Moreover, TCSC performed much better than TCI with a positive threshold. Recently, many machine/deep learning-based methods have been proposed for detecting life threatening ventricular arrhythmia, and these methods exhibited high performance [10-19].

TABLE I
SUMMARY OF RELATED WORK

Ref.	Filtering Method	Window Size	Detection Algorithms	Threshold Value	Data Set	Performance Measures
[7,8] (1993,1994)	-	4 sec.	TCI, ACF, VFF, SPEC, Neural network*	105 (TCI); A1 > 0.19, A2 > 0.45, A3 > 0.09 (SPEC); 0.625 (VFF)	Freeman Hospital CCU	*Sn, Sp*
[3] (2005)	*filtering.m* (pre-filtered)	3 sec. (TCI); 4 sec. (VF); 8 sec. (all)	ACF, CPLX, LI, MEA, SCA*, SPEC, STE, TCI, TOMP, VFF, WVL	400 (TCI); 6.61 (ACF); 0.406 - 0.625 (VFF); 0.173 - 0.426 (CPLX); 250 - 180 (STE, MEA); 2 - 32 (TOMP)	BIH-MIT; CU; AHA (7001-8210)	*Sn, Sp, Pp, Ac*, ROC, *Ct*
[4] (2005)	*filtering.m* (pre-filtered)	8 sec.	CPLX, SPEC, TCI, HILB*, VFF	-	BIH-MIT; CU; AHA (7001-8210)	*Sn, Sp, Pp, Ac*, ROC, *Ct*
[5] (2007)	*filtering.m* (pre-filtered)	8 sec.	CPLX, SPEC, TCI, TD*, VFF	-	BIH-MIT; CU; AHA (7001-8210)	*Sn, Sp, Pp, Ac*, ROC, *Ct*
[9] (2008)	*filtering.m* (pre-filtered)	8 sec.	CPLX, MEA, TCI, TD, VFF, Combining 2 algorithms*	400 (TCI); 0.406 - 0.625 (VFF); 0.125 (CPLX); 225 (MEA); 0.15 (TD); 2 - 32 (TOMP)	CU; File I (2276); File II (1501)	*Sn, Sp*, ROC
[2] (2010)	*filtering.m* (pre-filtered); HPF (1Hz); LPF (20Hz)	8 sec. 10 sec.	HILB, PSR, SPEC, TCI, Count, MAV & EMD*	Count < 250 for 10 sec. (Le); Count <200 for 8 sec. (Le)	BIH-MIT; CU; VFDB	*Sn, Sp, Pp, Ac*
[6] (2009)	*filtering.m* (pre-filtered)	8 sec.	CPLX, HILB, MEA, STE, TCI, TCSC*, TD	400 (TCI); 250 (STE); 230 (MEA); 0.426 (CPLX); A2,0 = 0.45 (SPEC); 0.15 (HILB, TD)	BIH-MIT; CU	*Sn, Sp, Pp, Ac*, ROC
[11] (2017)	*HPF (1Hz); LPF (45Hz)*	Window reference mark between 0.5 sec. and 1.2 sec.	Wigner Ville distribution; Machine Learning Classifiers	-	AHA; MIT-BIH	*Sn, Sp*
[12] (2018)	*HPF (0.5Hz); LPF (45Hz)*	4 sec.; 5 sec.; 8 sec.	DTFT; SVM; Radial Basis Function (RBF)	-	CU; VFDB; MIT-BIH	*Sn, Sp, Ac*
[13] (2018)	*HPF (1Hz); LPF (30Hz)*	5 sec.	SVM; C4.5	-	CU; VFDB	*Sn, Sp, Ac*
[14] (2018)	*HPF (0.5Hz); LPF (35Hz)*	1.091 ± 0.012 sec.; 2.843 ± 0.337 sec.	Mode Energy Variance; Mode Sample Entropy; Boosted-CART	-	CU; VFDB; MIT-BIH;	*Sn, Sp, Ac*
[15] (2019)	Wavelet-based filtering	2 sec.	FE; Renyi entropy (RenE)		CU; VFDB; MIT-BIH	*Sn, Sp, Ac*
[16] (2020)	*HPF (1Hz); FFREWT; Filter-bank*	2 sec.; 4 sec.; 5 sec.; 8sec.	Deep CNN	-	CU; VFDB	*Sn, Sp, Ac, F1*
[17] (2021)	*NLM*	2 sec.;5 sec.	SVM; AdaBoost; DE	-	CU; BIT-BIH	*Sn, Sp, Ac*
[18] (2023)	-	2.5 sec.	VANet	-	MIT-BIH Supraventricular Arrhythmia DB	*Ac*

**Support Vector Machine (SVM);* Digital Taylor-Fourier Transform (DTFT); Fuzzy entropy (FE); Convolutional Neural Network; Adaptive boosting (AdaBoost); Differential Evolution (DE); *Fixed frequency range empirical wavelet transform (FFREWT); Non-local Means Denoising (NLM)*

From the above analyses, we can see that 1) there are conflicting results regarding algorithmic performance such as TD, TCSC, and EMD; 2) all these evaluations were conducted offline in a stable environment, using prefiltered data sets, fixed and fine-tuned threshold values, and fixed window size

(often 8 s); 3) the studies commonly used annotated databases including Boston's Beth-Israel Hospital and MIT arrhythmia database (BIH-MIT), the Creighton University ventricular tachyarrhythmia database (CU), and the American Heart Association database (AHA); and 4) most studies, except [3–5], did not evaluate computational efficiency.

Clearly, the performance of the various algorithms is highly dependent on several factors such as data filtering, window size, and threshold value used. These factors must be carefully selected or managed to obtain a good performance. Determining the best threshold value and window size, often through extensive trial and error is critical for obtaining good results. This study intends to bridge these gaps in existing research, especially those related to a real-time testing environment.

III. FACTORS THAT IMPACT VF DETECTION

The process of detecting VF abnormalities often entails six major steps. Figure 1 shows the information flow along with potential research issues associated with each step. Most of these issues are common to any VF detection, whether performed offline or in real time. Meanwhile, although the VF detection process may appear simple and straightforward, various external factors can affect its overall performance. When VF detection process is conducted and evaluated using a low power microcontroller, it is necessary to check not only the appropriateness of such VF detection processing, but also the computational time that will be related to power consumption. Features for evaluation of potential and technical usability of wearables and mobile solutions for heart heath using statistical analysis were therefore extracted, including analysis of variance (ANOVA) and paired-T tests.

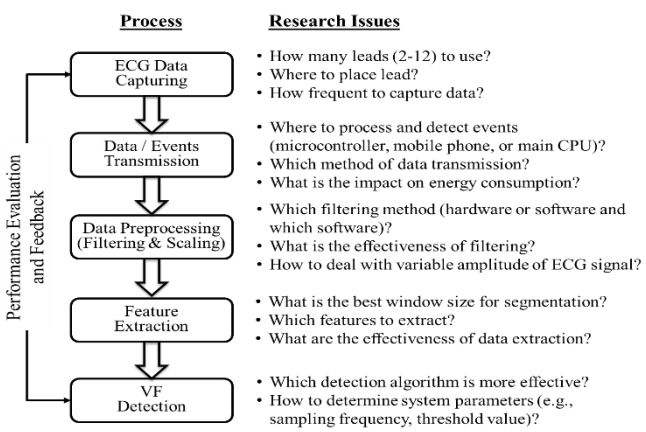


FIGURE 1. **Information flow of VF detection with research issues.**

A. SIGNAL CAPTURING AND DATA TRANSMISSION

In real-time health monitoring, an ECG signal is captured by connecting electrodes with the human body to a handheld or wearable ECG device. With the number of leads and electrode positioning affecting the quality of ECG capture [34]. The capture frequency provides the number of data values, resulting in a data-quantity-vs.-quality issue. While these issues are essential and must be determined for all VF detections, they are not the focus of our experiment.

There are three issues associated with data transmission: a) where (e.g., main computer, mobile, or microcontroller) to process and detect events, b) how to transmit data or events, and c) the impact on energy consumption. Among these, energy consumption is a major concern. Energy consumption in a wireless sensing device can be divided into four categories: scheduling of transitions, dynamic voltage scaling, memory, and radio transceivers [35]. While energy consumption depends on the types of microprocessors and radio transceivers used, in general, data transmission or computation in a microprocessor consumes most of the energy resource in the battery [36]. In this study, we propose to process data in an embedded microcontroller and to send only detected events to mobile phones the follow-up response. This study does not focus on a technical analysis of energy consumption, since it is beyond the scope of this paper.

B. FILTERING METHODS AND SCALING MECHANISMS

During real-time monitoring, several aspects (e.g., electrode location and the device itself) may generate errors and unexpected noise such as low- or high-frequency noises and baseline waves. This means that selecting and using the proper filtering method is important for accurate detection. Most early studies filtered the entire data set in advance using a well-known filtering process called *filtering.m* (available online [37]) that consists of four successive steps: 1) mean value subtraction, 2) 5th-order moving average filtering, 3) drift suppression using a high-pass filter (1-Hz cutoff frequency), and 4) high-frequency suppression using a low-pass Butterworth filter (30-Hz cutoff frequency) [6].

However, this approach cannot be used for real-time monitoring in which the signals are captured at fixed-time intervals (the window size), followed by filtering and scaling processes performed for each window size. To fit real-time needs, we removed the first two subprocesses of *filtering.m* and applied a common Kalman filter for tracking the baseline of an ECG signal in the testbed. In this study, we compared the potential effect of using conventional *filtering.m* (prefiltered) and real-time filtering methods with the results obtained when not using a filtering method.

Although a filtering method can help to remove outliers and noise, the amplitude of the ECG signal may still vary depending on sinus VF signals. In real-time detection, if the minimum and maximum peak values of a signal change in segment blocks, causing VF detection algorithms misinterpret the signal and leading to selection of an incorrect threshold value. Moreover, if the period of segmentation for the scale is too short, it is possible to lose the detection of QRS complexes.

On the other hand, if the period is too long, short portions of sequential VF events with small peaks can be falsely

detected as a sinus rhythm, so a scaling mechanism should be applied to gain better results. No previous study has noted or addressed this important factor To address this problem we propose a real-time scaling method applied in each window segment for all detection algorithms; this can be considered a control factor.

Figure 2 illustrates the proposed scaling process. Let S_t be the value of the raw ECG data at time *t*, V_t is the scaled value at time *t*, L_{max} is the max-limitation of the entire ECG signal, L_{min} is the min-limitation of the entire ECG signal, T_{max} is the max-peak value of the ECG signal in the window segment, and T_{min} is the min-peak value of the ECG signal in the window segment. In real time, we set L_{min} as 0 and L_{max} as 255. These are the limitations of the values obtained from the analog-to-digital converter. V_{mul}, the multiplier variable, and V_{add}, the addition variable, can be computed as

$$V_{Mul} = \frac{L_{Max} - L_{Min}}{T_{Max} - T_{Min}} \quad (1)$$

$$V_{Add} = \left| T_{Min} \times V_{Mul} \right| \quad (2)$$

The scaled value Vt can be calculated as

$$V_t = S_t V_{Mul} - V_{Add} \quad (3)$$

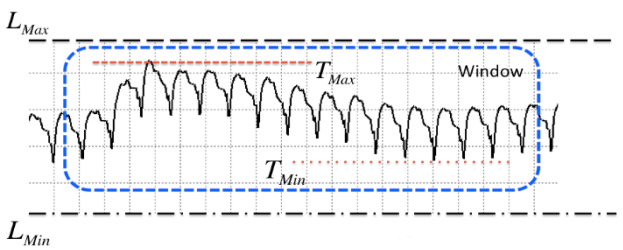


FIGURE 2. Sketch of scaling process figure number, followed by two spaces. It is good practice to explain the significance of the figure in the caption.

C. FEATURE EXTRACTION AND DETECTION

Extracting a sensor signal for VF detection often involves dividing sequential data into window segments (in seconds), followed by calculating the data features from each segment. The sequential data is filtered using different filtering methods. In most studies, the VF detection algorithm uses 8 s as the length for window segmentation [3–6]. Although Amann et al. [3] empirically selected an optimal window length of 8 s as the best result, this study explored and compared the impact of using a shorter window size such as 4 s. Based on the filtering methods and window sizes, extracted feature values are generated by the detection algorithms.

Many algorithms have been proposed and evaluated in the literature of VF detection. In real-time monitoring, it is likely that detection algorithms must be deployed in mobile devices or even in a microcontroller chip. Questions such as whether existing algorithms would work without modification (owing to limited available resources) and which algorithm performs better in a real-time environment remain unanswered. In this study, we selected five lightweight algorithms (TCI, TCSC, TD, VFF, and TOMP) for evaluation for the following reasons: 1) these algorithms performed well compared with other methods used in previous studies, and 2) they are simpler yet effective to be implemented in microcontrollers or mobile devices.

The TCI algorithm [28] checks the number of crossing points (C) of the ECG signal above or below a given threshold (T) within 1-s time intervals (I). It then applies three consecutive segments to calculate a TCI value. If the prescribed threshold value TCI_0 is greater than the TCI value, VF is declared. The main advantages of the TCI algorithm are that (1) the algorithm can be used to detect both QRS complexes and VF events with acceptable results, (2) the algorithm can be easily implemented in a single microprocessor without concern for resource constraints, and (3) it is quite energy-efficient owing to its simplicity.

The main drawback of the algorithm lies in its use of 1-s segments for analysis, because if the patient's heart rate is lower than 60 bpm, the algorithm may trigger an incorrect alarm. In addition, because the threshold line is fixed in 1-s segments, any sudden baseline noise may affect the value of the crossing threshold. Moreover, the accuracy of the algorithm depends significantly on the threshold value TCI, set as 400, and the threshold line, set as 20% of the maximum peak. If the threshold value is set close to the baseline P or the T wave of the ECG signal, many crossing values will be generated in a normal operation, so to obtain a good performance it is essential to use a good filtering process to remove high-frequency and baseline wander noises. Baseline tracking is also important when setting reasonable threshold values.

TCI may also encounter another problem because it uses only a positive threshold, and if only negative QRS complexes appear, the positive threshold is unable to detect the QRS. Moreover, if the VF wave crosses both the positive and negative thresholds, the number of crossing threshold impulses may become much higher than TCI, so TCSC [6] is proposed to improve the weaknesses of the TCI algorithm in three aspects:

1. TCSC uses a 3-s (instead of 1-s) time segment to check the number of crossing lines so that the misdetection problem for patients with a low heart rate (below 60 bpm) can be prevented.
2. TCSC uses both positive and negative thresholds to count the number of crossing signals. Since the number of samples that cross thresholds can be correctly computed, TCSC prevents possible misdetection.
3. The ECG counts signals above thresholds instead of counting the pulses (crossing above and below thresholds).

Since the VF signal is similar to a sine-wave form that oscillates up and down from the center baseline, the two thresholds cause the TCI value to increase by about twice as much as the TCI value obtained using only one threshold. Therefore, TCSC improves VF detection accuracy and

stability over TCI. However, the performance of the algorithm also depends significantly on the threshold values used, so choosing the filtering process is an essential factor for obtaining high performance. In the proposed implementation, to save battery power, this study further eliminates the multiplication of the *cosine* window.

The TD algorithm [5] uses a 40 × 40 square grid of a two-dimensional phase space diagram to analyze ECG signals for identifying dynamic low or random behavior. By plotting the signal data $X(t)$ on the x-axis against τ time delay data $X(t + \tau)$ on the y-axis (where $\tau = 0.5$ s), the result is a plot showing the number of visited boxes. The patterns of the filled boxes differentiate the VF signal from a normal SR signal. In the cases of a normal signal, the plot shows two lines in the phase space plot. In the VF signal case, the boxes visited are shown as uniformly distributed boxes over the phase space plot. To objectively distinguish the QRS complex from the VF cases, TD counts the density of boxes visited (N_B) and compares it with a threshold value (N_{TH}). If $N_B < N_{TH}$, the ECG signal is considered to be a normal QRS; otherwise, it will be classified as a VF case.

In general, while the TD algorithm exhibits high performance in detecting VF patterns, when a VF event with variable peak-to-peak values occurs, differences in the peak-to-peak values in a window block can generate errors. Moreover, TD must check 1600 boxes at every window segment, so it is more computationally time-intensive.

The VFF algorithm [29] uses a narrow-band elimination filter to reject a specific range of frequencies, with the mean period of a fixed length of data computed using a discrete Fast Fourier Transform. Once the VF event occurs, the data segment is shifted by half a period. If, like VF, the data is similar to that of a periodic signal, the algorithm cancels it so that the output value of the VF-filter leakage becomes small. The signal amplitude of QRS complexes affects the threshold of the VFF algorithm. If the signal is higher than the peak of the QRS complex from the previous window segment, the threshold is set as 0.406. Otherwise, the threshold is set as 0.625.

One major weakness of the VFF algorithm is that it can only focus on VF rhythms that are similar to *sine* or *cosine* waves. Since this algorithm does not include any QRS or sinus rhythm detection methods, the algorithm requires help from another detection algorithm to identify whether there is a sinus rhythm. Although the VFF algorithm requires only little computational power, the detection performance is overall lower than for other VF methods for the reason mentioned.

TOMP [30] applies a squaring function and moving-window integration in a window segment to detect QRS complexes. Specifically, the sliding integration window sums the absolute values of the difference between the current data at *i* and the previous data at *i* - 1 in a width of 150 *ms*. The short period of the moving window therefore captures the high-peak QRS complexes and sudden changes in the ECG data.

Using this method, even positive or negative QRS complexes can be easily detected by this sliding integration window. The TOMP algorithm sets two thresholds for the number of QRS complexes, such as $l_0 = 2$ and $l_1 = 32$. If the number of the QRS is smaller than l_0 or higher than l_1, the diagnosis is for VF. This algorithm is robust at baseline noise and expends relatively low computational power compared with other algorithms. Since in the TOMP algorithm, the actual detection does not focus on QRS complexes but on VF waveforms, it is very important to set the offset value for determining a VF wave. Unfortunately, since the threshold value for the sliding window varies from data set to data set, it is difficult to set the appropriate threshold value for VF detection.

D. PERFORMANCE EVALUATION

Traditionally, the relative performance of detection is measured by the accuracy rate (*Ac*). To address the class imbalance problem of VF data sets where there exist significantly more normal QRS signals than VF signals [32], this study used three additional metrics for proper evaluation: sensitivity (*Sn*), specificity (*Sp*), and positive predictivity (*Pp*). These measures have been proposed and are commonly used in evaluating VF detection research. To calculate them, four types of values are collected: false positives (*FP*), false negatives (*FN*), true positives (*TP*), and true negatives (*TN*). Using these values, various performance measures can be derived. The metrics (*Sn*, *Sp*, *Pp*, and *Ac*) can be calculated as follows:

$$Sn = TP / (TP + FN), \quad (4)$$
$$Sp = TN / (TN + FP), \quad (5)$$
$$Pp = TP / (TP + FP), \quad (6)$$
$$Ac = (TP + TN) / (TP + FP + TN + FN). \quad (7)$$

Sn is the fraction of the VF signal being correctly detected, *Sp* is the fraction of the No-VF signal being correctly detected that can be used to assess the capability of detecting the Sinus Rhythm (SR), and *Pp* measures the fraction of detecting VF signals based on classified VF cases. Therefore, even if the sensitivity is near 100%, if *Pp* is below 30%, the VF detection algorithm classifies VF cases that are actually SR signals.

Ac is the probability of capturing all correct decisions. It would seem that *Ac* represents the effectiveness of VF detection, but since a VF database often consists mostly of SR (about 80%) and less of VF (about 20%), if an algorithm classifies every ECG signal as SR, then *Ac* becomes simply 80% of *Ac* with 0% of *Sn*. We should refer therefore to all four metrics when evaluating the relative performance of the VF algorithms; this could be a major challenge for real-world cases as there is a tradeoff between different measures for different threshold values used.

One way of aggregating these measures is to use an ROC curve, obtained by plotting sensitivity value against the (1 - specificity) value at various threshold settings. Each point on the ROC curve represents a pair of sensitivity and (1 - specificity) values corresponding to a particular decision threshold. Figure 3 shows the ROC curves of two detection results (R1 and R2) vs. a random guessing result (R3). The closer the ROC curve is to the upper left corner, the higher the overall accuracy of the detection [32], so the R3 curve exhibits the worst results, and the results for the R1 curve are better than those for the R2 curve.

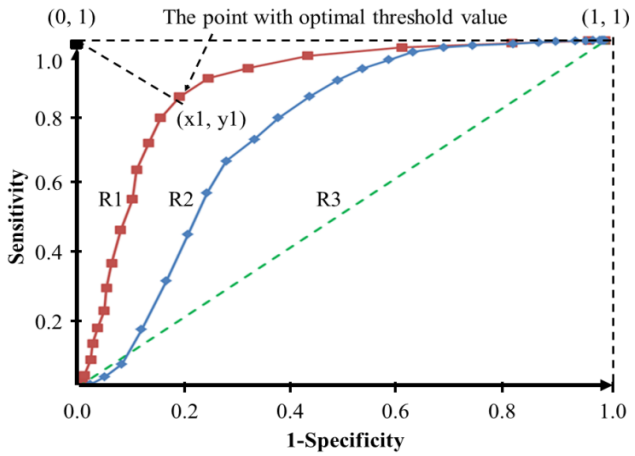


FIGURE 3. ROC curves of two classification models.

Computational time *Ct* is measured in milliseconds to assess computational efficiency. The area under the ROC curve (*AUC*) is a single objective value reflecting the performance of the detection algorithms. *AUC* is the probability of identifying which of two stimuli is noise and which is a signal plus noise [32]. *AUC* can be calculated as

$$AUC = (1 + (TP/(TP + FN)) - (FP/(FP + TN))/2. \quad (8)$$

E. THRESHOLD VALUE

Determining the best threshold value for a detection algorithm is a challenging task because changing threshold values will have a tradeoff impact on sensitivity and specificity measures. This study proposed use to the ROC curve to obtain the optimal threshold value. Referring to Figure 3, we determined the optimal threshold value based on the smallest Euclidian distance between the upper-left corner point (0, 1) and a point on the ROC curve. The distance value D_{th} can be calculated as

$$D_{th} = \sqrt{(x_{i\pm u})^2 + ((y_{i\pm u}) - 1)^2} = \sqrt{x_1^2 + (y_1 - 1)^2} \quad (9)$$

where *i* is the index of class variables, and *u* is the unit value of the index. If the ROC curve is closer to point (0, 1), then the overall performance of the threshold value is better than for the other threshold value. Using Equation (4), we can calculate the distance between the points of ROC curves and the point of the top left (0, 1) and identify the point with the shortest distance.

IV. EXPERIMENTS

To conduct the experiments, three main factors were considered: filtering methods (F), window size (W) of data extraction, and detection algorithm (D). We controlled four other factors common to all monitoring: data capture, data transformation, data scaling, and threshold values. Because it must be specially optimized, we also controlled the threshold value.

A. TESTING DATA SETS

To evaluate the potential impacts of these factors, this study conducted a thorough evaluation using the Creighton University (CU) ventricular tachyarrhythmia database [31], a database has been widely used for evaluating almost all VF detection algorithms (see Section 2). The CU database contains data for 35 patients who have had VF attacks recorded with normal ECG signals. The description of CUDB indicates that they record one ECG signal, meaning ECG capture by at least 3 leads at a gain of 400 adu/mV. Second, since the sampling frequency of ECG in CUDB is 250Hz, in this study, we converted the sampling from 250Hz to 60Hz for efficient data processing using resampling. Third, ECG signal capture in the CUDB uses 12 bits of ADC, so we converted 12 bits to 8 bits to minimize computational time. For comparative analysis, experienced cardiologists carefully annotated these data sets. Compared with other databases such as BIH-MIT and AHA, the CU database provides a clearer annotation of VF, and since almost every normal beat is also annotated in each QRS complex, this study used only the CU database for testing.

B. EXPERIMENTAL DESIGN

We developed a comprehensive 3 × 2 × 5 factorial experiment to examine the impact and relative performance of 1) without (F1) and with *filtering.m* (F2) or real-time filtering (F3) methods, 2) window sizes of 4 s (W1) and 8 s (W2), and 3) detection algorithms TCI (D1), TCSC (D2), TD (D3), VFF (D4), and TOMP (D5). Hence, an experimental design with 30 cells (3 × 2 × 5) was used to represent the combinations of all factors. For each cell, 35 data sets were used. In total, this study used 1050 data points for the experiment (30 cells with 35 data points in each cell).

The following null hypotheses were formulated for the proposed experiment:

- H1: The mean value of a) *Sn*, b) *Sp*, c) *Pp*, d) *Ac*, and e) *Ct* is the same for the five detection methods.
- H2: The mean value of a) *Sn*, b) *Sp*, c) *Pp*, d) *Ac*, and e) *Ct* is the same for the two window sizes.
- H3: The mean value of a) *Sn*, b) *Sp*, c) *Pp*, d) *Ac*, and e) *Ct* is the same for the three filtering situations.

C. SYSTEM IMPLEMENTATION AND VALIDATION

We developed an integrated simulated environment to emulate real-time monitoring and detection, using the CU database as inputs to a virtual patient. This environment transforms a VF data set into electrical ECG signals representing a virtual patient, using them as real-time data for detection processing in a microcontroller, and analyzes the extracted features from the real-time detection system in the evaluation simulator. Figure 4 depicts the framework of the research environment that consists of three major modules: 1) a data bank, that hosts the VF database and annotated results as a ground truth; 2) an evaluation simulator, that includes virtual patients that generate ECG signals and performs evaluations comparing the detection results from the detection system with the annotated results from the data bank; and 3) a real-time detection system, that emulates real-time monitoring and VF detection. The module includes several key processes: filtering, scaling, extraction, threshold optimization, and detection. All these processes are performed online.

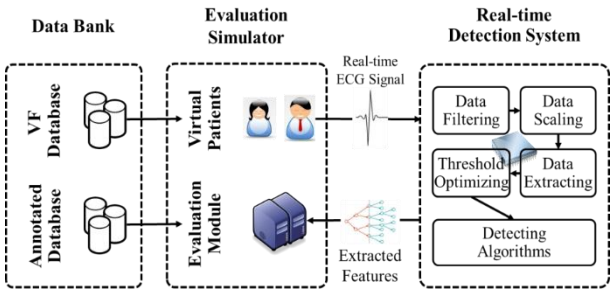


FIGURE 4. Real-time VF detection simulated environment.

TABLE II
THRESHOLD VALUES FOR EXPERIMENT

Detection Algorithm	Window Size	Filtering Method		
		No Filtering	*filtering.m* [36]	Real time
TCI	4s	385	285	290
	8s	185	190	145
TCSC	4s	7	9	8
	8s	15	15	17
TD	4s	132	123	132
	8s	218	200	214
VFF	4s	0.88	0.71	0.615
	8s	0.86	0.72	0.61
TOMP	4s	12	12	12
	8s	20	21	20

The simulator that provides ECG signal data to the connected microcontroller was programmed in Visual C# 2008 and ran on a Windows 7 operating system with a 1.6-GHz CPU and a real-time detection system based on the MSP430-f2274 microcontroller that processes the ECG signal, generates the features of VF detection, and sends the features to a simulator through a serial communication link. We generated and optimized the thresholds for each possible combination (see Table 2). The General Linear Model of ANOVA running in MINITAB was used to analyze the significance of the factor effects and their interactions.

Because VF detection has a known tradeoff problem in which the sensitivity and specificity values may have a tradeoff for the particular threshold value used, we can only apply paired t-tests to compare the computational times. To compensate, this study used the ROC curve to assess the relative solution quality of different pairs of levels for each factor.

V. COMPUTATIONAL RESULTS AND ANALYSIS

A. OVERALL AND RELATIVE IMPACT

The results from the experiment are summarized in Table 3 in which the columns present the measured results for various combinations of the experimental factors and the rows show the performance of each combination of factors. The values in each cell are the average values for the 35 data sets corresponding to that cell. The cells highlighted in boldface represent factor combinations where all four quality metrics perform better than average. The cells marked in italics (or in red color) are those combinations that have a quality performance below 50%. The algorithm combinations with the shortest computational times are highlighted in boldface, and the cells marked in italics (or in red) are those algorithm combinations exhibiting the worst time performance.

TABLE III
PERFORMANCE FOR COMBINATIONS OF DETECTION ALGORITHM, WINDOW SIZE AND FILTERING METHOD

	Factors	*Sn* (%)	*Sp* (%)	*Pp* (%)	*Ac* (%)	*Ct* (ms)
TCI	D1W1F1	67.44	53.76	45.40	57.73	151.88
	D1W1F2	53.94	74.96	59.85	68.10	738.73
	D1W1F3	91.50	22.36	32.27	43.07	191.80
	D1W2F1	55.95	67.79	50.68	65.18	75.31
	D1W2F2	48.45	74.73	59.41	67.57	688.99
	D1W2F3	79.53	43.91	41.46	57.15	122.27
TCSC	D2W1F1	71.17	54.13	42.74	57.70	98.94
	D2W1F2	66.29	64.30	52.53	65.59	752.77
	D2W1F3	31.27	89.07	53.97	73.36	235.00
	D2W2F1	60.12	67.34	53.09	66.55	58.68
	D2W2F2	70.24	58.04	52.03	62.22	675.18
	D2W2F3	20.84	95.89	57.19	75.48	110.92
TD	D3W1F1	75.29	83.49	68.97	81.97	453.24
	D3W1F2	76.86	82.09	64.30	81.21	1458.75
	D3W1F3	74.30	82.42	65.74	81.34	668.25
	D3W2F1	69.82	82.53	68.59	80.29	273.77
	D3W2F2	72.78	81.83	65.71	79.89	1294.53
	D3W2F3	69.18	82.96	65.14	80.00	447.70
VFF	D4W1F1	79.50	31.46	34.50	44.37	244.24
	D4W1F2	47.58	79.36	52.47	71.20	978.87
	D4W1F3	66.50	83.70	66.12	79.39	337.59
	D4W2F1	76.94	35.70	37.84	47.63	126.58
	D4W2F2	47.65	78.03	54.55	70.12	867.33
	D4W2F3	59.32	86.13	68.97	80.08	214.50
TOMP	D5W1F1	67.81	66.50	53.40	67.13	491.57
	D5W1F2	69.56	71.95	58.74	69.59	1388.23
	D5W1F3	62.72	69.74	55.38	68.46	516.96
	D5W2F1	52.13	87.85	66.30	79.52	139.47
	D5W2F2	47.10	92.81	73.78	80.68	1174.13
	D5W2F3	46.41	90.82	70.33	79.96	319.90
Average		62.61	71.19	56.38	69.42	

In terms of solution quality, note that the TD algorithm performed the best overall in all factor combinations; they have relatively high values in all quality measures, followed by the TCSC and TOMP algorithms. VFF and TCI did not perform well; about 30% of their cell values perform below 50%. However, in terms of computational efficiency, TCSC and TCI are the most computationally-efficient algorithms, while TD is the worst in that respect. In general, the *filtering.m* method takes a much longer time to perform than the average.

Table 4 depicts the impact results in terms of *AUC* values from a combination of window sizes, filtering methods, and detection algorithms. As shown, the type of detection algorithm had an obvious impact on the *AUC* results, while filtering methods and window sizes had only a slight impact. Among the detection methods, the TD algorithm is the clear winner with the highest and most consistent *AUC* values ranging from 0.82 to 0.83 with a minimum standard deviation (0.69). TCSC is the next-best algorithm (values ranging from 0.62 to 0.68), followed by VFF (values ranging from 0.61 to 0.69), and then TOMP (values ranging from 0.44 to 0.68). The performance of TCI is unacceptably low (below 0.41) despite its small standard deviation (1.62).

TABLE IV
AUC RESULTS: IMPACT OF WINDOW SIZE, FILTERING METHOD AND DETECTION ALGORITHM BASED ON THRESHOLD VALUES.

Detection Algorithms	**TD**		**VFF**		**TCI**		**TCSC**		**TOMP**	
Filtering Methods	8s	4s	8s	4s	8s	4s	8s	4s	8s	4s
Without Filtering	0.82	0.82	0.61	0.69	0.39	0.41	0.65	0.62	0.67	0.44
Filtering.m	0.82	0.83	0.68	0.61	0.39	0.39	0.66	0.68	0.64	0.68
Real-time Filtering	0.82	0.83	0.62	0.61	0.36	0.38	0.68	0.65	0.65	0.67
Standard Deviation	0. 69		3.83		1.62		2.39		9.26	

Table 5 presents the results of the analysis of variance (ANOVA) using the three main factors: detection methods, window lengths, and filtering methods. The results indicate that 1) all three factors are significant at $P < 0.001$ for sensitivity, specificity, and computational time measures, thereby rejecting the null hypotheses H1 a) b) d), H2 a) b) d), and H3 a) b) d); 2) the detection method is significant at $P < 0.001$ for positive predictivity and accuracy measures, thereby rejecting the null hypotheses H1 c) and H1 d); and 3) the filtering factor is significant at $P < 0.001$ for the accuracy measurement, thereby rejecting the null hypothesis H3 d).

This study also evaluated two-way interactions between factors. The interaction between the detection algorithm and filtering method was significant for all measures. The interaction between the window size and filtering method was significant only for the computational time measure. The other interactions were relatively small.

TABLE V
ANOVA – IMPACT OF WINDOW SIZE, FILTERING METHOD AND DETECTION ALGORITHM

(a) Sensitivity

Factor	**DF**	**Adj SS**	**Adj MS**	**F Value**	**P (Sig)**
Detection Algorithm (D)	4	48815	12204	17.76	0.000
Window Size (W)	1	18310	18310	26.65	0.000
Filtering Method (F)	2	13181	6591	9.59	0.000
2-way Interactions:					
D*W	4	7470	1868	2.72	0.029
D*F	8	142628	17829	25.95	0.000
W*F	2	1031	516	0.75	0.472
Error	1028	706297	687		
Total	1049	937734			

(b) Specificity

Factor	**DF**	**Adj SS**	**Adj MS**	**F Value**	**P (Sig)**
Detection Algorithm (D)	4	96344	24086	25.91	0.000
Window Size (W)	1	15991	15991	17.20	0.000
Filtering Method (F)	2	34940	17470	18.79	0.000
2-way Interactions:					
D*W	4	15942	3985	4.29	0.002
D*F	8	185361	23170	24.92	0.000
W*F	2	3619	1809	1.95	0.143
Error	1028	955757	930		
Total	1049	1307954			

(c) Positive Predictivity

Factor	**DF**	**Adj SS**	**Adj MS**	**F Value**	**P (Sig)**
Detection Algorithm (D)	4	51894	12973	10.58	0.000
Window Size (W)	1	7225	7225	5.89	0.015
Filtering Method (F)	2	9891	4945	4.03	0.018
2-way Interactions:					
D*W	4	6052	1513	1.23	0.295
D*F	8	46928	5866	4.79	0.000
W*F	2	397	199	0.16	0.850
Error	1028	1260210	1226		
Total	1049	1382597			

(d) Accuracy

Factor	**DF**	**Adj SS**	**Adj MS**	**F Value**	**P (Sig)**
Detection Algorithm (D)	4	56105.2	14026.3	26.46	0.000

Window Size (W)	1	4501	4501	8.49	0.004
Filtering Method (F)	2	11168.4	5584.2	10.53	0.000
2-way Interactions:					
D*W	4	5702.1	1425.5	2.69	0.030
D*F	8	49087.8	6136	11.58	0.000
W*F	2	1347.5	673.7	1.27	0.281
Error	1028	544934.7	530.1		
Total	1049	672846.6			

(e) Computational Time(ms)

Factor	DF	Adj SS	Adj MS	F Value	P (Sig)
Detection Algorithm (D)	4	34124070	8531017	5005.89	0.000
Window Size (W)	1	5231464	5231464	3069.75	0.000
Filtering Method (F)	2	128957105	64478553	37835.14	0.000
2-way Interactions:					
D*W	4	1313248	328312	192.65	0.000
D*F	8	5682625	710328	416.81	0.000
W*F	2	43032	21516	12.63	0.000
Error	1028	1751915	1704		
Total	1049	177103459			

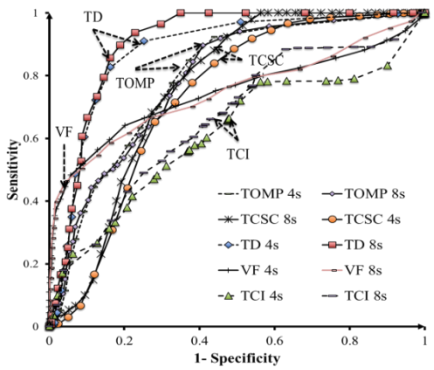
In summary, among the three factors examined, the detection algorithm had the strongest impact, followed by the filtering method (except for the sensitivity measure, for which the window size had a higher impact).

Choosing the right algorithm and filtering method is critical. In general, a good algorithm exhibits more robust behaviour, experiencing less impact by other factors. Each detection algorithm has unique characteristics that may also have an impact on the selection of the filtering method. For example, using a shorter window segment is important for mobile devices because of resource constraints. However, this may affect solution quality and increase the computational time.

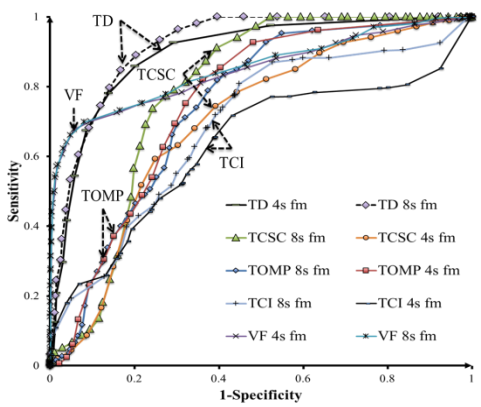
While the ANOVA results generally indicate which factors have a significant impact on performance, they do not indicate at which level each of these factors exhibits better performance. A careful examination of the mean values in Table 3 indicates that three combinations, D3W1F1, D3W1F2, and D3W1F3, are close to being tied for best performance. Since it is less efficient to apply a filtering method, we can infer that using the TD algorithm with a 4-s data extraction window and without using a filtering method generates the best solution quality and requires less time to detect a heart attack.

B. COMPARISON OF DETECTION METHODS

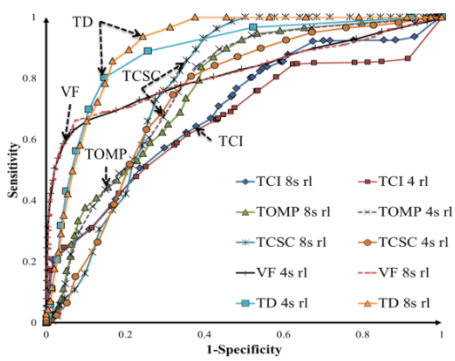
According to the results shown in Table 3, TD, TCSC, and TOMP have higher detection rates for all four (or most) metrics. Because VFF shows a higher accuracy rate but has a very low sensitivity rate, it cannot be considered as a good method. Therefore, the overall results indicate that TD, TCSC, and TOMP are more suitable algorithms for VF detection.



(a)



(b)



(c)

FIGURE 5. ROC curves of detection algorithms (a) without filtering, (b) with *filtering.m*, and (c) with real-time filtering.

Figure 5(a) shows ROC curves for all algorithms, using both 4- and 8-s window sizes but without using a filtering method (raw ECG data). Corresponding sets of ROC curves using either *filtering.m* or the real-time filtering method are depicted in Figure 5(b) and 5(c), respectively. Figure 5

indicates that 1) the TD algorithm outperforms the other four algorithms, and its performance also has less impact in terms of different window sizes; 2) the performances of the other four algorithms are unacceptable due to the shape of ROC curves, and their performances are significantly impacted by the window size; 3) without using a filtering method, TOMP performs slightly better than TCSC, but when using either filtering method, the performances of TOMP and TCSC are closer; and 4) the performance of TCI is among the worst, while VFF performs slightly better than TCI.

Table 6 depicts the results of a paired t-test among different detection algorithms in terms of computation time. Table 6(a) shows the average results with a 4-s window segment (W1) under three different filtering strategies. Table 6(b) shows the average results with an 8-s window segment (W2) under three different filtering strategies.

The results show the following: 1) An 8-s window segment is obviously more computationally-efficient than a 4-s window segment because it cuts the number of times to extract segment signals in half, but it also takes 4 s longer to provide warnings; in an emergency situation where every second matters, this could be a disadvantage. In addition, it will be more difficult to implement 8-s windows within devices that have limited memory resources. 2) Among the five algorithms this study evaluated, TCSC and TCI were computationally more efficient (highlighted in boldface), followed by VFF and then TOMP. 3) The TCSC algorithm performed significantly faster than the other algorithms, except that in two cases (W1F2 and W1F3) its performance is slightly slower than that of TCI. This indicates that TCI performs better than TCSC when filtering methods with 4-s window segments are used. 4) In general, TD performed much more slowly than the other algorithms, with the differences being statistically significant.

TABLE VI
PAIRED-T TEST: THE EFFECT OF DETECTION ALGORITHM ON COMPUTATIONAL TIME (MS)

(a) Window size 4 seconds

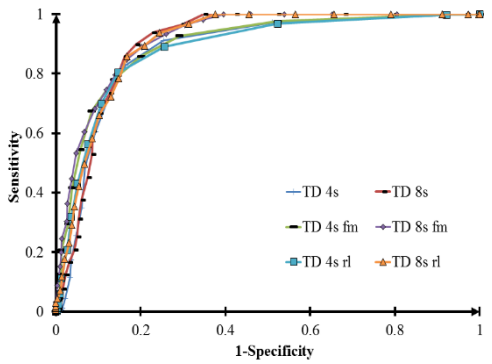
Filtering	Algorithm	Mean	STDEV	Sample	TCI	TCSC	TD	VFF	TOMP
F1	TCI	151.88	32.99	35	-	9.68	-28.27	-12.03	-37.93
	TCSC	98.94	13.50	35	0.000	-	-40.01	-25.89	-68.66
	TD	453.24	52.13	35	0.000	0.000	-	17.44	-3.96
	VFF	244.23	32.02	35	0.000	0.000	0.000	-	-32.63
	TOMP	491.57	29.34	35	0.000	0.000	0.000	0.000	-
F2	TCI	738.73	20.21	35	-	-2.51	-80.47	-15.20	-130.02
	TCSC	752.76	27.29	35	0.017	-	-77.36	-13.14	-113.64
	TD	1458.75	46.55	35	0.000	0.000	-	25.63	8.05
	VFF	978.9	94.5	35	0.000	0.000	0.000	-	-23.68
	TOMP	1388.23	25.76	35	0.000	0.000	0.000	0.000	-
F3	TCI	191.80	30.44	35	-	-6.05	-38.49	-24.47	-44.75
	TCSC	235.00	29.14	35	0.000	-	-37.82	-16.11	-34.73
	TD	668.3	60.6	35	0.000	0.000	-	32.33	12.94
	VFF	337.59	19.99	35	0.000	0.000	0.000	-	-25.03
	TOMP	516.96	36.53	35	0.000	0.000	0.000	0.000	-

(b) Window size 8 seconds

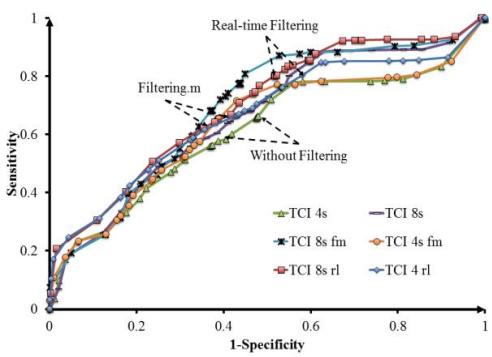
Filtering	Algorithm	Mean	STDEV	Sample	TCI	TCSC	TD	VFF	TOMP
F1	TCI	75.31	10.26	35	-	6.87	-27.31	-16.04	-6.01
	TCSC	58.68	11.23	35	0.000	-	-27.77	-19.59	-6.94
	TD	273.77	42.26	35	0.000	0.000	-	18.42	9.37
	VFF	126.58	16.26	35	0.000	0.000	0.000	-	-1.11
	TOMP	139.5	66.2	35	0.000	0.000	0.000	0.277	-
F2	TCI	688.99	11.55	35	-	5.72	-68.41	-55.91	-109.37
	TCSC	675.19	10.37	35	0.000	-	-69.29	-59.74	-112.92
	TD	1294.53	55.40	35	0.000	0.000	-	43.26	12.02
	VFF	867.33	17.08	35	0.000	0.000	0.000	-	-64. 81
	TOMP	1174.13	23.38	35	0.000	0.000	0.000	0.000	-
F3	TCI	122.27	10.93	35	-	3.61	-33.43	-33.79	-29.61
	TCSC	110.92	15.33	35	0.001	-	-34.05	-34.15	-26.92
	TD	447.69	57.21	35	0.000	0.000	-	24.15	12.23
	VFF	214.50	13.93	35	0.000	0.000	0.000	-	-14.93
	TOMP	319.90	42.65	35	0.000	0.000	0.000	0.000	-

C. IMPACT OF FILTERING METHOD AND WINDOW SIZE

From (Figure 5), based on this investigation, we can see that TD is overall the best method while TCI is the worst in terms of solution quality, but the situation reverses when they are considered in terms of computational efficiency. We will now explore their relative performance and impact of various factors by focusing on the best and worst cases. Figures 6(a) and (b) show the ROC curves for the TD and TCI algorithms, respectively, for different combinations of window sizes and filtering methods.



(a)



(b)

FIGURE 6. ROC curves of (a) TD algorithm and (b) TCI algorithm.

Clearly, the TD algorithm performed better than the TCI algorithm in term of aggregated quality performance, since all ROC curves of the different factor combinations are almost the same shape and close to to the left-top corner overall, indicating that TD is quite robust and achieves high accuracy. From Figure 6(a), we can see that the performance of TD was not impacted much by the filtering method, and by contrast, as shown in Figure 6(b), the performance of TCI improved when filtering methods were used. However, it is difficult to

compare the relative performances of the two filtering methods when they are close to one another. From Figure 6(a), we can also see that the performance of TD was not impacted significantly by the window size. In contrast, as shown in Figure 6(b), the performance of TCI using 8-s window segments was better than when using 4-s window segments.

D. TRAINING AND TESTING OF DATA SETS

Although the detailed factor analysis in this study uses all 35 patient data sets from CUDB, generalizability is somewhat lacking. To ensure repeatability, this study trained the threshold values based on 80% of the entire data set, and tested the algorithm based on 20% of the data set. Since comparison and analysis showed that the TD algorithm outperformed the other four algorithms, we conducted a training and testing experiment using TD only. Table 7 shows the impact results in terms of sensitivity, specificity, and *AUC* values from a combination of window sizes and filtering methods. As shown, the overall results for the *AUC* values based on a testing data set of 20% are slightly lower than when testing all data sets in Table 7 (82–83%).

TABLE VII
SENSITIVITY, SPECIFICITY, AND AUC RESULTS: IMPACT OF WINDOW SIZE AND FILTERING METHOD BASED ON THE TRAINED THRESHOLD VALUES USING 80% OF CU DATA SET.

	Window Size	Sn (%)	Sp (%)	AUC (%)
Without Filtering	4s	86.65	71.66	79.155
	8s	81.14	68.75	74.945
Filtering.m	4s	84.99	74.73	79.86
	8s	82.89	73.69	78.29
Real-time Filtering	4s	82.46	73.66	78.06
	8s	83.89	64.22	74.055

Table 7 Sensitivity, specificity, and AUC results: Impact of window size and filtering method based on the trained threshold values using 80% of CU data set.

E. CROSS-VALIDATION

To validate greater presence of generalizability, this study applied 10-fold cross-validation. The comparison and analysis showed that the TD algorithm outperformed the other four algorithms in terms of the balance between Sn and Sp. Table 8 shows the impact results in terms of sensitivity, specificity, positive-predictivity and accuracy values from a combination of algorithms, window sizes and filtering methods. The number of data points is 3,897 for the window size of 4 seconds and 1,968 for 8 seconds and we used the logistic regression for selecting the optimal threshold of a classification. This study excluded the TCI algorithm since the TCSC is an improvement of TCI.

TABLE VIII
PERFORMANCE FOR COMBINATIONS OF DETECTION ALGORITHM, WINDOW SIZE AND FILTERING METHOD BASED ON 10-FOLD CROSS VALIDATION

	Factors	***Sn (%)***	***Sp (%)***	***Pp (%)***	***Ac (%)***
TCSC	D2 W1 F1	35.30	96.60	76.50	81.96
	D2 W1 F2	17.30	96.10	57.90	77.23
	D2 W1 F3	32.80	95.10	67.60	80.18
	D2 W2 F1	33.20	97.70	81.80	82.16
	D2 W2 F2	15.00	96.20	55.50	76.67
	D2 W2 F3	28.10	95.70	67.50	79.46
TD	D3 W1 F1	67.60	94.00	77.80	87.65
	D3 W1 F2	66.60	94.10	78.00	87.53
	D3 W1 F3	67.20	92.40	73.50	86.37
	D3 W2 F1	62.40	94.20	77.40	86.58
	D3 W2 F2	64.10	93.60	76.10	86.53
	D3 W2 F3	61.90	93.20	74.20	85.66
VFF	D4 W1 F1	31.80	98.60	87.60	82.62
	D4 W1 F2	57.60	98.10	90.50	88.42
	D4 W1 F3	51.20	97.90	88.70	86.78
	D4 W2 F1	32.10	98.80	89.40	82.77
	D4 W2 F2	56.90	98.20	90.90	88.26
	D4 W2 F3	53.10	97.70	87.80	86.93
TOMP	D5 W1 F1	39.70	95.70	74.30	82.32
	D5 W1 F2	63.16	48.74	27.89	52.18
	D5 W1 F3	32.70	96.80	76.20	81.47
	D5 W2 F1	42.70	95.90	76.50	83.07
	D5 W2 F2	20.30	97.80	74.40	79.16
	D5 W2 F3	40.80	95.90	76.00	82.66

VI. DISCUSSION

Real-time monitoring and evaluation are essential for practical VF detection, and since previous performance evaluations on VF detection were mainly conducted offline with prefiltered data sets and focused on the effectiveness of the detection algorithms, such performance results may not be applicable to real-world cases. In general, it is difficult (or impossible) to use real cases for experimentation when studying the algorithms and the impact of various factors. To address this challenge, this study developed a simulation environment for evaluation. Using a public VF database as inputs, we generated real-time signals for validating real-time VF detection processes such as data filtering, scaling, segmentation, and detection analysis.

While millisecond differences in computational time may not act an important role in the VF detection itself, in real-time wearable-device VF detection, computational time can play an important role in enabling the device to be operated for a long time using a small-capacity battery. Thus, along with VF detection, this study evaluated computational time that is also important in real-time VF detection. Table 8 lists the average computational times when running CU data sets in a prototype of a wearable ECG system. As shown, Table 8 depicts similar trends and relative performance (Table 3) among the detection algorithms. TCI and TCSC are still rated the most efficient algorithms in an embedded microcontroller, with TCI performing slightly better than TCSC. Basically, these algorithms share similar computational logic because TCSC reflects an improvement of TCI, and the differences could be a result of processing and architectural differences between the laptop-based simulator (1.6 GHz) and the microcontroller-based wearable sensor (16 MHz). The quality performance ranking results obtained from the microcontroller and simulator were identical, validating the reliability of using the proposed simulator for experimental analyses.

TABLE VIII

SUMMARY OF AVERAGE COMPUTATIONAL TIME (MS) RUNNING IN THE EMBEDDED MICROCONTROLLER

Factors	TCI	TCSC	TD	VFF	TOMP
W1 F1	28.28	30.23	102.24	53.34	97.79
W1 F2	28.45*	29.89*	99.47*	53.13*	98.47*
W1 F3	460.16	462.38	534.39	485.50	529.95
W2 F1	14.00	15.11	51.12	26.67	48.90
W2 F2	14.21*	14.95*	51.88*	27.47*	49.38*
W2 F3	447.81	448.11	484.11	459.66	481.89

* *filtering.m* cannot be run in microcontroller, so we use pre-filtered data and cannot count the time for running *filtering.m*

Consistent with the recent trend of using mobile devices for health monitoring, a wearable ECG module could be useful in daily life because of its quick response to users. The proposed ECG devices will collect data including noise directly and process VF detection. In this case, the suggested integration of preprocessing (filtering and scaling) and data extraction into the devices would play an important role in practical VF detection.

In general, because it requires the use of the entire signal to perform the first two subprocesses, it is difficult to implement the *filtering.m* method in a real-time environment, and this study adopted and modified it as a real-time filter. The real-time filtering method we have proposed exhibits an overall better performance in TD and TCSC than when applying the *filtering.m* method (prefiltered). While the best performance of all possible factor combinations used an 8-s window of TD with the *filtering.m* method, when no filtered data was used, a 4-s window of TD exhibited the best performance among all results.

Although nonfiltered data contains various noises that can interfere with accurate VF detection, our proposed scaling method mitigates the noise effects. Moreover, a shorter window size (4 s) can improve scaling accuracy without filtering. The phase-space reconstruction of the TD algorithm also makes the difference between SR and VF signals much larger than the original signals. In the SR signal, reconstructed boxes mostly overlapped as a baseline or a part of a QRS complex. On the other hand, the VF signal was distributed widely over the entire phase space. This characteristic of TD makes the adaptation of such detection robust from the point of view of various factor impacts.

Basically, the TOMP algorithm is designed for detecting QRS complexes. Although it is relatively effective at finding QRS complexes, the length of the integration window is not adjusted for VF signals similar to sinusoidal waves. Since the VF signal has a slope that is not as sharp as that of QRS complexes, although TOMP showed good performance in some test cases, it also showed weaknesses in many selected data sets.

VFF is a weak algorithm regarding its impact on adjusting the threshold values. If a normal signal (SR) is not detected by other QRS detection algorithms, the VFF algorithm selects the SR signal as being randomly-distributed. In this study, since this study analyzed the factor impacts for each algorithm, no QRS detector was applied to the VFF algorithm. TCI is a basic algorithm that uses the threshold line to check the R-peak signal. Although previous studies have evaluated TCI with reasonable results, since proposed experiments showed that the TCI results reflect large differences between SR and VF decisions, optimizing the thresholds does not impact the performance of the TCI algorithm.

This study applied 1 Hz (HPF) and 30 Hz (LPF) as real-time filtering. According to a survey paper [38], fifteen studies applied bandpass filters as preprocessing steps between 0.05 Hz and 120 Hz. Among those studies, two applied the bandpass filter at a filtering frequency between 1Hz and 30 Hz and their detection performance achieved accuracies greater than 99%. The applied frequencies of high-pass and low-pass filters were 0.1 Hz and 30 Hz, respectively, and, based on the previous studies, these frequencies are reasonable frequencies for ECG analysis, so as a real-time filter for ECG signal analysis, the proposed implementation is appropriate for VF detection.

There have recently been multiple attempts to implement a machine-learning algorithm on embedded microcontrollers for detecting ventricular arrhythmia. Chen et al. have proposed a lightweight deep-learning method on a microcontroller with a millisecond-scale inference [17]. Although the microcontroller used there is of higher performance compared to ours, it is shown that deep-learning approaches may soon be implemented into low-power microcontrollers for detecting VF in real-time. Since Tiny machine-learning approaches are also focused on this research area, our proposed methodology could be a good testing environment.

VII. CONCLUSION

There are several factors in real-world applications that affect the overall performance of real-time VF detection. This paper discussed seven such factors and examined the impacts of three of them on VF detection. Because they are critical and should be strategically considered in all experiments Four other factors were considered as control factors. This study evaluated the performance and impact using the complete Creighton University (CU) ventricular tachyarrhythmia database.

The proposed study concluded the following: 1) The scaling process is very important and required for successful VF detection. 2) All three factors evaluated (detection method, window size, and filtering method) have significant impact on VF detection, with the detection method requiring careful selection to reduce its degree of impact to a minimum level; otherwise, a filtering method should be considered. 3) TD and TCSC are two lightweight algorithms that outperform the other three methods evaluated. TD in particular is more robust in reducing the impact of other factors to insignificance, while TCSC is more computationally efficient. Considering this tradeoff, we conclude that using TD would be preferable because detection accuracy and robust are particularly critical.

4) The threshold value is an important parameter for any detection algorithm, and its value needs to be properly determined to achieve good performance.

This study contributes to the field in the following aspects: First, it proposes an integrated simulated environment enabling researchers to use existing databases for emulating real-time detection processes, identifying critical factors, and evaluating performance. Second, it proposes a simple scaling method for improving overall data quality. Third, it examined and empirically tested factors that may affect VF detection, and provided guidelines for detection-method selection. Finally, it describes a method for helping optimize determination of threshold values for detection methods using the ROC curve. It also identified the range of threshold values for the algorithms studied, and since these values were derived from 35 data sets, they can be used in general.

In this study, we used the entire data set from a database to conduct the analysis, and noticed clear and obvious variations among different data sets (patients) in terms of SR patterns, signal strengths, etc. Personalization of VF detection and its overall impact is a major area that deserves further exploration. Detection accuracy is also impacted by the particular activity that the patient is performing, so activity recognition is also an active topic for future research.

REFERENCES

[1] World Health Organization. Available online: http://www.who.int/mediacentre/factsheets/fs317/en/ index.html (accessed on 13 December 2017).

[2] E. M. Anas, S. Y. Lee, and M. K. Hasan, "Sequential algorithm for life threatening cardiac pathologies detection based on mean signal strength and EMD functions," Biomed Eng Online, vol. 9, p. 43, Sep 4 2010, doi: 10.1186/1475-925X-9-43.

[3] A. Amann, R. Tratnig, and K. Unterkofler, "Reliability of old and new ventricular fibrillation detection algorithms for automated external defibrillators," Biomed Eng Online, vol. 4, p. 60, Oct 27 2005, doi: 10.1186/1475-925X-4-60.

[4] A. Amann, R. Tratnig, and K. Unterkofler, "A New Ventricular Fibrillation Detection Algorithm for Automated External Defibrillators," Comput. Cardiol. Vol 32, p. 559–562, 2005.

[5] A. Amann, R. Tratnig, and K. Unterkofler, "Detecting ventricular fibrillation by time-delay methods," IEEE Trans Biomed Eng, vol. 54, no. 1, pp. 174-7, Jan 2007, doi: 10.1109/TBME.2006.880909.

[6] M. A. Arafat, A. W. Chowdhury, and M. K. Hasan, "A simple time domain algorithm for the detection of ventricular fibrillation in electrocardiogram," Signal, Image and Video Processing, vol. 5, no. 1, pp. 1-10, 2009, doi: 10.1007/s11760-009-0136-1.

[7] R. H. Clayton, A. Murray, and R. W. Campbell, "Comparison of four techniques for recognition of ventricular fibrillation from the surface ECG," Med Biol Eng Comput, vol. 31, no. 2, pp. 111-7, Mar 1993, doi: 10.1007/BF02446668..

[8] R. H. Clayton, A. Murray, and R. W. Campbell, "Recognition of ventricular fibrillation using neural networks," Med Biol Eng Comput, vol. 32, no. 2, pp. 217-20, Mar 1994, doi: 10.1007/BF02518922.

[9] A. H. Ismail, M. Fries, R. Rossaint and S. Leonhardt, "Validating the Reliability of Five Ventricular Fibrillation Detecting Algorithms," In Proceedings of the 4th European Conference of the International Federation for Medical and Biological Engineering, Antwerp, Belgium, pp. 26–29, 23–27 November 2008.

[10] Q. Li, C. Rajagopalan, and G. D. Clifford, "Ventricular fibrillation and tachycardia classification using a machine learning approach," IEEE Trans Biomed Eng, vol. 61, no. 6, pp. 1607-13, Jun 2014, doi: 10.1109/TBME.2013.2275000.

[11] A. Mjahad, A. Rosado-Munoz, M. Bataller-Mompean, J. V. Frances-Villora, and J. F. Guerrero-Martinez, "Ventricular Fibrillation and Tachycardia detection from surface ECG using time-frequency representation images as input dataset for machine learning," Comput Methods Programs Biomed, vol. 141, pp. 119-127, Apr 2017, doi: 10.1016/j.cmpb.2017.02.010.

[12] R. K. Tripathy, A. Zamora-Mendez, O. S. J. A. de la, M. R. A. Paternina, J. G. Arrieta, and G. R. Naik, "Detection of Life Threatening Ventricular Arrhythmia Using Digital Taylor Fourier Transform," Front Physiol, vol. 9, p. 722, 2018, doi: 10.3389/fphys.2018.00722.

[13] M. Mohanty, S. Sahoo, P. Biswal, and S. Sabut, "Efficient classification of ventricular arrhythmias using feature selection and C4.5 classifier," Biomedical Signal Processing and Control, vol. 44, pp. 200-208, 2018, doi: 10.1016/j.bspc.2018.04.005.

[14] Y. Xu, D. Wang, W. Zhang, P. Ping, and L. Feng, "Detection of ventricular tachycardia and fibrillation using adaptive variational mode decomposition and boosted-CART classifier," Biomedical Signal Processing and Control, vol. 39, pp. 219-229, 2018, doi: 10.1016/j.bspc.2017.07.031.

[15] R. Panda, S. Jain, R. K. Tripathy, and U. R. Acharya, "Detection of shockable ventricular cardiac arrhythmias from ECG signals using FFREWT filter-bank and deep convolutional neural network," Comput Biol Med, vol. 124, p. 103939, Sep 2020, doi: 10.1016/j.compbiomed.2020.103939.

[16] D. Panigrahy, P. K. Sahu, and F. Albu, "Detection of ventricular fibrillation rhythm by using boosted support vector machine with an optimal variable combination," Computers & Electrical Engineering, vol. 91, 2021, doi: 10.1016/j.compeleceng.2021.107035.

[17] T. Chen, A. Gherardi, A. Das, H. Li, C. Xu, and W. Xu, "Short:VANet: An Intuitive Light-Weight Deep Learning Solution Towards Ventricular Arrhythmia Detection," Smart Health, vol. 28, 2023, doi: 10.1016/j.smhl.2023.100388.

[18] G. T. Taye, E. B. Shim, H. J. Hwang, and K. M. Lim, "Machine Learning Approach to Predict Ventricular Fibrillation Based on QRS Complex Shape," Front Physiol, vol. 10, p. 1193, 2019, doi: 10.3389/fphys.2019.01193.

[19] D. Lai, X. Fan, Y. Zhang, and W. Chen, "Intelligent and Efficient Detection of Life-Threatening Ventricular Arrhythmias in Short Segments of Surface ECG Signals," IEEE Sensors Journal, vol. 21, no. 13, pp. 14110-14120, 2021, doi: 10.1109/jsen.2020.3031597.

[20] I. Jekova, and V. Krasteva, "Real time detection of ventricular fibrillation and tachycardia," Physiol Meas, vol. 25, no. 5, pp. 1167-78, Oct 2004, doi: 10.1088/0967-3334/25/5/007.

[21] P. Leijdekkers, and V. Gay, "A Self-test to Detect a Heart Attack Using a Mobile Phone and Wearable Sensors," In Proceedings of the 2008 21st IEEE International Symposium on Computer-Based Medical Systems, University of Jyväskylä, Finland, 17–19 June 2008, pp. 93–98.

[22] A. Pantelopoulos, and N. G. Bourbakis, "A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 40, no. 1, pp. 1-12, 2010, doi: 10.1109/tsmcc.2009.2032660.

[23] Z. X, Zhang, X. W. Tian, and J. S. Lim, "Real-Time Algorithms for a Mobile Cardiac Monitoring System to Detect Life-Threatening Arrhythmias," In Proceedings of 2010 The 2nd International Conference on Computer and Automation Engineering, Singapore, 26–28 February 2010, pp.232–236.

[24] U. R. Acharya et al., "Automated identification of shockable and non-shockable life-threatening ventricular arrhythmias using convolutional neural network," Future Generation Computer Systems, vol. 79, pp. 952-959, 2018, doi: 10.1016/j.future.2017.08.039.

[25] J. A. Healey, "Wearable and Automotive Systems for Affect Recognition from Physiology," Ph.D. Thesis, Dept. of Electrical Engineering and Computer Science, MIT, USA, 2000.

[26] S. Fokhenrood, P. Leijdekkers, and V.Gay, "Ventricular Tachycardia/Fibrillation Detection Algorithm for 24/7 Personal Wireless Heart Monitoring," In Proceedings of 5th International Conference on Smart Homes and Health Telematics, Nara, Japan, 21–23 June 2007, pp. 110–120.

[27] Z. Jia, D. Li, C. Liu, L. Liao, X. Xu, L. Ping, and Y. Shi, "TinyML Design Contest for Life-Threatening Ventricular Arrhythmia Detection," 2023, *arXiv preprint arXiv:2305.05105*.

[28] N. V. Thakor, Y. S. Zhu, and K. Y. Pan, "Ventricular tachycardia and fibrillation detection by a sequential hypothesis testing algorithm," IEEE Trans Biomed Eng, vol. 37, no. 9, pp. 837-43, Sep 1990, doi: 10.1109/10.58594.

[29] S. Kuo, "Computer Detection of Ventricular Fibrillation," *Comput. Cardiol.* **1978**, pp. 347-349.

[30] J. Pan and W. J. Tompkins, "A real-time QRS detection algorithm," IEEE Trans Biomed Eng, vol. 32, no. 3, pp. 230-6, Mar 1985, doi: 10.1109/TBME.1985.325532.

[31] A. L. Goldberger et al., "PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals," Circulation, vol. 101, no. 23, pp. E215-20, Jun 13 2000, doi: 10.1161/01.cir.101.23.e215.

[32] M. Galar, A. Fernandez, E. Barrenechea, H. Bustince, and F. Herrera, "A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 42, no. 4, pp. 463-484, 2012, doi: 10.1109/tsmcc.2011.2161285.

[33] J. Kim, "Real-time ventricular fibrillation detection for pervasive health monitoring," Ph.D. Dissertation, Pennsylvania State University, PA, USA, May 2014

[34] E. Tragardh, H. Engblom, and O. Pahlm, "How many ECG leads do we need?," Cardiol Clin, vol. 24, no. 3, pp. 317-30, vii, Aug 2006, doi: 10.1016/j.ccl.2006.04.005.

[35] V. Raghunathan, C. Schurgers, P. Sung, and M. B. Srivastava, "Energy-aware wireless microsensor networks," IEEE Signal Processing Magazine, vol. 19, no. 2, pp. 40-50, 2002, doi: 10.1109/79.985679.

[36] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Emerging challenges: Mobile networking for "Smart Dust"," Journal of Communications and Networks, vol. 2, no. 3, pp. 188-196, 2000, doi: 10.1109/jcn.2000.6596708.

[37] Karl Unterkofler. Available online: https://homepages.fhv.at/ku/karl/VF/filtering.m (accessed on 13 December 2017).

[38] S. Kaplan Berkaya, A. K. Uysal, E. Sora Gunal, S. Ergin, S. Gunal, and M. B. Gulmezoglu, "A survey on ECG analysis," Biomedical Signal Processing and Control, vol. 43, pp. 216-235, 2018, doi: 10.1016/j.bspc.2018.03.003.

Jungyoon Kim received the B.S. degree in electronics and the M.S. degree in electrical and computer engineering from the University of Ulsan, Korea, in 2004 and 2006, respectively, and the Ph.D. degree in information sciences and technology from Pennsylvania State University, University Park, PA, USA, in 2014. He is currently an Assistant Professor of Computer Science with Kent State University, where he is also the Founding Director of the Smart Communities and IoT Laboratory His areas of experience and expertise are in smart health and wellbeing, especially in real-time cardiovascular disease and stress monitoring, physiological sensor design, and intelligent analytics for decision supports; environmental monitoring and assessment—especially in air quality monitoring and cut slope movement monitoring; and ubiquitous computing—especially in embedded system design, energy efficient processing, and programming model for networking performance.

Jaehyun Park received the B.S. and Ph.D. degrees in industrial and management engineering from Pohang University of Science and Technology (POSTECH). He worked at the User Experience Team, Samsung Electronics, as a Senior Designer, from 2013 to 2015. He is currently an Associate Professor with the Department of Industrial and Management Engineering, Incheon National University (INU). His research interests include semantic network analysis, machine/deep learning on physical behavior, and computational cognitive engineering.

Misun Kang is an assistant professor in the Department of Computer Software Engineering at Soonchunhyang University. She received the B.S., M.S., and Ph.D. degrees in computer science and engineering from Ewha Womans University, Seoul, Korea in 2007, 2012 and 2017, respectively. She was Research Fellow in the Department of Computer Science and Engineering, Ewha Womans University. She worked as a staff engineer at Samsung Display. Her research interests include AI-based bio-medical image processing.