On-Line Recursive Decomposition of Intramuscular EMG Signals Using GPU-Implemented Bayesian Filtering

Tianyi Yu, Konstantin Akhmadeev, Eric Le Carpentier, Yannick Aoustin, and Dario Farina, *Fellow, IEEE*

Abstract*—*Objective: **Real-time intramuscular electromyography (iEMG) decomposition, which is needed in biofeedback studies and interfacing applications, is a complex procedure that involves identifying the motor neuron spike trains from a streaming iEMG recording.** ***Methods:*** **We have previously proposed a sequential decomposition algorithm based on a Hidden Markov Model of EMG, which used Bayesian filter to estimate unknown parameters of motor unit (MU) spike trains, as well as their action potentials (MUAPs). Here, we present a modification of this original model in order to achieve a real-time performance of the algorithm as well as a parallel computation implementation of the algorithm on Graphics Processing Unit (GPU). Specifically, the Kalman filter previously used to estimate the MUAPs, is replaced by a least-mean-square filter. Additionally, we introduce a number of heuristics that help to omit the most improbable decomposition scenarios while searching for the best solution. Then, a GPU-implementation of the proposed algorithm is presented.** ***Results:*** **Simulated iEMG signals containing up to 10 active MUs, as well as five experimental fine-wire iEMG signals acquired from the tibialis anterior muscle, were decomposed in real time. The accuracy of decompositions depended on the level of muscle activation, but in all cases exceeded 85%.** ***Conclusion:*** **The proposed method and implementation provide an accurate, real-time interface with spinal motor neurons.** ***Significance:*** **The presented real time implementation of the decomposition algorithm substantially broadens the domain of its application.**

***Index Terms*—Hidden markov models, bayes methods, recursive estimation, deconvolution, electromyography decomposition, parallel computation, real-time decomposition.**

Manuscript received May 10, 2019; revised September 4, 2019; accepted October 14, 2019. Date of publication December 6, 2019; date of current version May 20, 2020. This work was supported in part by the University of Nantes, Ecole Centrale de Nantes, China Scholarship Council under Grant 201404490033 and in part by the European Research Council Synergy Grant "Natural BionicS" Contract #810346. *(Corresponding author: Dario Farina.)*

T. Yu, K. Akhmadeev, E. Le Carpentier, and Y. Aoustin are with the Laboratoire des Sciences du Numérique de Nantes, UMR 6004, CNRS, École Centrale de Nantes, Université de Nantes.

D. Farina is with the Imperial College London, London SW7 2AZ, U.K. (e-mail: d.farina@imperial.ac.uk).

Digital Object Identifier 10.1109/TBME.2019.2948397

I. Introduction

THE electromyogram (EMG) is the recording of electrical activity of the muscle fibers, as generated during muscle contractions. This activity results from the neural excitation originating from the motor neurons (MN) in the spinal cord. The procedure of identification of MN spike trains from an EMG is termed *EMG decomposition*. Such information is important in scientific studies of the motor system as well as in neurological examinations. A real-time decomposition increases the range of applicability of EMG processing, including biofeedback studies and human-machine interfaces.

A majority of currently existing EMG decomposition algorithms [1]–[6] are fundamentally off-line. On-line decomposition was previously addressed in [7], where a multichannel surface EMG signal was decomposed using a convolution kernel compensation approach [8]. Moreover, a real-time clustering and template matching algorithm for iEMG was presented in [9]. This algorithm was designed to estimate the cumulative discharge rate of MNs but does not provide resolution of action potentials superimposed in time. Similar challenges as in iEMG decomposition are encountered in spike sorting algorithms for extracellular recordings from cortical neurons [10]–[12] and from peripheral nerves (electroneurogram) [13], [14].

Recently, we proposed a Bayesian filtering approach for single-channel iEMG decomposition [15], as well as its version adapted to a case of varying number of active MUs [16]. The proposed algorithm achieves full sequential decomposition of iEMG signals. Although the proposed method requires long computation time, it can be accelerated due to its parallel structure. In this paper, we introduce modifications in the original algorithm as well as its parallel implementation on GPU, which permit to achieve real-time decomposition.

The approach presented in this work differs from its closest analogue [7] by utilization of single-channel iEMG instead of multichannel sEMG, which entails a fundamentally different model of EMG signal. Compared to another method of iEMG decomposition [9], which uses on-line clustering in order to classify MUAPs, the proposed approach resolves superpositions, thus providing more accurate decomposition and potentially scaling up to higher efforts at which MUAPs no longer occur isolated from each other.

In the sequel, we will briefly review the Hidden Markov Model (HMM) of iEMG previously proposed in [15] (Section II). A Bayesian filtering procedure estimating the parameters of MUs will be presented in Section III. Further, we will introduce methods to reduce the complexity of the original algorithm (Section IV). Then we will present its parallel implementation (Section V). Simulated and experimental iEMG signals used to assess the proposed approach are described in Section VI. Finally, results of experimental signal decomposition will be shown and analyzed in Section VII.

II. Hidden Markov Model

A. Generation of the EMG Signal

The *motor unit* (MU) is the elementary entity of the human neuromuscular system. A MU comprises a MN in the spinal cord and the muscle fibers it innervates. The MNs receive inputs from supraspinal levels of the motor system and from muscle afferents. While active, a MN exhibits a firing activity in the form of a train of *action potentials* (spikes) that reachs the muscle fibers via the MN axon and causes their contraction. Thus, muscle fibers belonging to the same MU are excited almost simultaneously, producing a short variation of electric potential in a nearby electrode, called *motor unit action potential* (MUAP). Multiple MUs located in the vicinity of the electrode simultaneously contribute to the overall signal, mixing their MUAP trains in one channel. The inter-spike intervals (ISI) of the trains exhibit a certain regularity and have a physiologically-inherent minimal value, called *refractory period*.

B. Observation Model of HMM

Based on the EMG generation principles, we have proposed a Hidden Markov Model (HMM) of EMG in [15], [16]. In the following, we will provide a short overview of this model since this description is needed to introduce the modifications made for real-time implementation..

The observation equation of HMM was derived from the linear model of EMG [17], [18]:

$$Y[n] = \sum_{i\in\Omega} \varphi_i(S[n])\, H_i[n] + W[n] \tag{1}$$

- n is the discrete time index;
- i is the index of MU;
- Ω is the set of both active and inactive MUs' indexes shown in Table I;
- Y is the observed iEMG signal;
- S is the activation scenario, composed of two elements: $S[n] = (A[n], (T_j[n])_{j\in A[n]})$. $A[n]$ is the set of indexes of MUs that are active at time n. $(T_j[n])_{j\in A[n]}$ is a discrete sequence that, for each active MU j, characterizes the time passed since its previous spike;
- H is the MUAP waveform with finite length ℓ_{IR};
- W is the independent identically distributed white noise sequence with unknown variance v;
- $\varphi_i(s)$ is, for each realization $s = (a, (t_i)_{i\in a})$, is a row vector of size ℓ_{IR} with all components equal to zero; except, if $i \in a$ and $t_i < \ell_{\text{IR}}$, the component in position $t_i + 1$ is equal to 1.

C. State Vectors and Transition Laws of HMM

In order to model the ISI statistics of a MU, we have used the discrete Weibull distribution [15], which is defined by the vector $\Theta_i[n]$ containing two parameters: a location parameter $t_{0i}[n]$ and a shape parameter $\beta_i[n]$.

Thus, the state vectors of HMM are formulated as follows:

- $S[n] = (A[n], (T_i[n])_{i\in A[n]})$ the activation scenario,
- $H[n] = (H_i[n])_{i\in\Omega}$ the MUAP shapes,
- $\Theta[n] = (\Theta_i[n])_{i\in\Omega}$ the inter-spike law parameters.

At first, for simplicity, we assume that $H_i[n]$ and $\Theta_i[n]$ are constant in time. Thus, their transition laws are as follows:

$$H_i[n+1] = H_i[n] \tag{2}$$

$$\Theta_i[n+1] = \Theta_i[n] \tag{3}$$

In practice, $H_i[n]$ and $\Theta_i[n]$ are not constant. An adaptation to their steady changes will be introduced later in Section III-D. The transition laws for $S[n] = (A[n], (T_i[n])_{i\in A[n]})$ are presented in the following two subsections, respectively for the two components $T_i[n]$ and $A[n]$.

1) Renewal Model: As shown in [15], the process $(T_i[n])_{n\in A[n]}$ is Markovian. For each $i \in A[n+1] \cap A[n]$, given $\Theta_i[n]$, its transition distribution is:

$$T_i[n+1] = \begin{cases} 0 & \text{w.p. } r(T_i[n]+1, \Theta_i[n]) \\ T_i[n]+1 & \text{w.p. } 1 - r(T_i[n]+1, \Theta_i[n]) \end{cases} \tag{4}$$

where $r(\cdot)$ is the hazard rate function of the Discrete Weibull distribution [19].

Moreover, as we described previously in Section II-A, ISIs have a lower bound termed *refractory period* t_{R}. We choose $t_{\text{R}} = 30$ ms, based on physiological findings [20]. Thus, we have:

$$r(t, \Theta_i[n]) = 0, \text{ if } t < t_{\text{R}} \tag{5}$$

2) Recruitment Model: Regulation of muscle contraction force is achieved by concurrent modulation of MN firing frequencies and recruitment of additional MUs. The recruitment mechanism is modelled as the variation of $A[n]$, which contains the indexes of all active MUs. Given $S[n]$, it has the following transition law:

$$A[n+1] = \begin{cases} A[n] \setminus i & \text{w.p. 1, if } T_i[n] = t_{\text{I}} \\ A[n] \cup i & \text{w.p. } \frac{\lambda}{\text{card}(\bar{A}[n])}, \text{ if i } \notin A[n] \\ A[n] & \text{w.p. } 1-\lambda \end{cases} \tag{6}$$

where $\text{card}(\bar{A}[n])$ denotes the number of inactive MUs. An i-th active MU is considered to be derecruited when $T_i[n]$ reaches a predefined limit t_{I}, usually chosen as $7t_{\text{R}}$. A random inactive MU is considered recruited with predefined constant probability λ and initialized with $T[n] = 0$. Thus, $1 - \lambda$ is the probability of no MUs being activated at the instant n.

III. Bayes Filter

A. Principles

The state vectors of HMM $H[n], \Theta[n]$ and $S[n]$ are recursively estimated by Bayes filter. In the following, the exponent $|^n$ means "given the data Y^n". The posterior probability functions of the state vectors are:

- The probability density function (PDF) of $\Theta[n]$ given S^n, H and Y^n. Clearly, H and Y^n are not necessary for the estimation of $\Theta[n]$. Moreover, due to the assumption of independence in MU activity (which is approximated in reality [15]), this PDF is the product of the PDF of $\Theta_i[n]$ given S^n. The expected value of Θ_i given S^n, noted $\hat{\theta}_{i,S^n}$, is approximated by a recursive maximum likelihood (RML) estimation. Details and mathematical derivation of this procedure can be found in [16].
- The PDF of $H[n]$ given S^n and Y^n. With the marginalization principle [21], this PDF is gaussian and is estimated by a Kalman filter as described in Section III-B. The mean and the variance of this PDF will be denoted $\hat{H}^{|n}_{S^n}$ and P_{S^n}. Furthermore, the Kalman filter provides the observation prediction noted as $\hat{Y}^{|n-1}_{S^n}$ and its variance noted as v_{S^n}. To simplify the calculation complexity, a least-mean-square (LMS) filter is proposed to replace the Kalman filter in Section III-B.
- The probability mass function (PMF) of S^n given Y^n (see part III-C).

B. Estimation of Impulse Responses

1) Kalman Filter: Given S^n, the Markov model for impulse responses reduces, for all $n \geq 1$:

$$\begin{cases} H[n+1] = H[n] \\ Y[n] = \sum_{i \in \Omega} \varphi_i(S[n])\, H_i[n] + W[n] \end{cases} \tag{7}$$

If $H[1]$ is Gaussian, formula (7) is a standard linear Gaussian model. $H[n]|S^n, Y^n$ is Gaussian with mean $\hat{H}^{|n}_{S^n}$ and covariance matrix P_{S^n}, $Y[n]|S^n, Y^{n-1}$ is Gaussian with mean $\hat{Y}^{|n-1}_{S^n}$ and variance v_{S^n}. These means and variances are estimated recursively by the Kalman filter. With the initial prior $\hat{H}^{|0}_{S^0}$ and P_{S^0}, we have, for all $n \geq 1$:

- Prediction of observation:

$$\begin{aligned} \hat{Y}^{|n-1}_{S^n} &= \psi(S[n])\, \hat{H}^{|n-1}_{S^{n-1}} \\ v_{S^n} &= \psi(S[n])\, P_{S^{n-1}}\, \psi(S[n])^\top + v \end{aligned} \tag{8}$$

- Estimation of state:

$$\begin{aligned} K_{S^n} &= P_{S^{n-1}}\, \psi(S[n])^\top\, v^{-1}_{S^n} \\ \hat{H}^{|n}_{S^n} &= \hat{H}^{|n-1}_{S^{n-1}} + K_{S^n}\, (Y[n] - \hat{Y}^{|n-1}_{S^n}) \\ P_{S^n} &= P_{S^{n-1}} - K_{S^n}\, v_{S^n}\, K^\top_{S^n} \end{aligned} \tag{9}$$

where $\psi(s) = [\varphi_1(s), \ldots, \varphi_{\text{card}(\Omega)}(s)]$, $\text{card}(\Omega)$ denotes the number of MUs.

The variance v of the noise is unknown. A heuristic approach is proposed to estimate it with the square of the estimation error $Y[n] - \psi(S[n])\, \hat{H}^{|n}_{S^n}$.

$$\hat{V}^{|n}_{S^n} = \left(1 - \frac{1}{n}\right) \hat{V}^{|n-1}_{S^{n-1}} + \frac{1}{n} (Y[n] - \psi(S[n])\, \hat{H}^{|n}_{S^n})^2 \tag{10}$$

And its global estimation is:

$$\hat{V}^{|n} = \sum_{S^n} \hat{V}^{|n}_{S^n}\, \mathsf{Pr}^{|n}(S^n = s^n) \tag{11}$$

where $\hat{V}^{|n}$ replaces v in the formula (8).

2) Least Mean Square Filter: Due to the size of matrix P_{S^n}, which is $(\text{card}(\Omega) \times \ell_{\text{IR}}) \times (\text{card}(\Omega) \times \ell_{\text{IR}})$, the Kalman filter requires a large computational power. Here, we propose the least-mean-square filter (LMS) to replace the Kalman filter to accelerate the estimation.

The derivation procedure from Kalman filter to the LMS filter is justified in Appendix A. With the rough initial prior $\hat{H}^{|0}_{S^0}$, for all $n \geq 1$, we have the formula of the LMS filter:

$$\begin{aligned} \epsilon[n] &= Y[n] - \psi(S[n])\, \hat{H}^{|n-1}_{S^{n-1}} \\ m_{\Delta,i}[n] &= \frac{\sum_j \Delta_i[j]}{\text{card}(\Delta_i)} \\ \tilde{v}[n] &= 1 + \frac{1}{n} \sum_i m_{\Delta,i}[n]\, \varphi_i(S[n]) \varphi_i(S[n])^\top \\ \hat{H}^{|n}_{i,S^n} &= \hat{H}^{|n-1}_{i,S^{n-1}} + \frac{m_{\Delta,i}[n] \varphi_i(S[n]) \epsilon[n]}{n\, \tilde{v}[n]} \end{aligned} \tag{12}$$

where $\Delta_i[j]$ denotes the j-th inter-spike interval of the i-th MU; $\text{card}(\Delta_i)$ is the number of inter-spike intervals of the i-th MU; $m_{\Delta,i}[n]$ is the expectation value of the inter-spike intervals of the i-th MU at the time index n; and $\tilde{v}[n]$ represents the ratio of the variance of innovation v_{S^n} to the variance of noise $\hat{V}^{|n}$.

TABLE I
Main Notations

Y	The iEMG signal
Ω	The set of indexes of all MUs
A	The set of indexes of active MUs
U	Spike trains
W	White noise
H	The vector of MU action potentials shapes
ℓ_{IR}	The maximum MUAPs length
T	The sawtooth sequences
S	The activation scenario
$\Theta = [t_0, \beta]$	The vector containing discrete Weibull distribution parameters: the location parameter and the concentration parameter
t_R	The shifting parameter of discrete Weibull distribution, that is the refractory period
Pr	Probability
w.p.	with probability
$Y[n]$	The iEMG signal at time index n
Y^n	The vector containing the signal from time index 1 to n
$\|^n$	Given Y^n
$\mathsf{Pr}(T[n] = t[n])$	The probability of the sawtooth sequences at time index n being equal to a value $t[n]$. For all elements of the state vector, the uppercase symbols denote random variables, while the lowercase ones stand for their values.

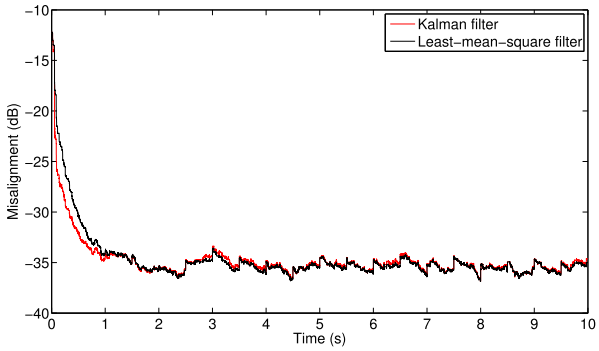

Fig. 1. Misalignment of the Kalman filter algorithm and least-mean-square filter algorithm.

The prediction of observation $\hat{Y}_{S^n}^{|n-1}$ is the same as the formula (8) and the prediction of the variance of innovation v_{S^n} is:

$$v_{S^n} = \tilde{v}[n]\ \hat{V}^{|n}. \tag{13}$$

The performances of the Kalman filter and the LMS filter were compared as follows. A simulated signal with the activity of five MUs was generated by the HMM model with the time varying impulse responses $H[n]$. Given the scenario S^n and rough initial impulse responses $\hat{H}_{S^0}^{|0}$, the two filters were used to identify $H[n]$. The measure of performance was the normalized misalignment (in dB), defined as $20\log_{10}[\|H[n] - \hat{H}_{S^n}^{|n}\|_2/\|H[n]\|_2]$. Fig. 1 shows the misalignment for the two filters. These results indicate that LMS provides almost the same estimates as the Kalman filter, and, thus, is preferred due to the computational time gain.

C. Posterior Probability of Scenario

As proposed in our previous work [16], the posterior probability recursion was derived by means of an update-prediction scheme. As follows from the Bayes' theorem, for all possible realizations s^n of S^n, the update step is:

$$\mathsf{Pr}^{|n}(S^n = s^n) \propto \mathsf{Pr}^{|n-1}(S^n = s^n)\ g(Y[n] - \hat{Y}_{s^n}^{|n-1}, v_{s^n}) \tag{14}$$

where $g(.\,, v)$ is a zero-mean and variance v Gaussian PDF.

The prediction step is:

$$\begin{aligned}\mathsf{Pr}^{|n}(S^{n+1} = s^{n+1}) &= \mathsf{Pr}^{|n}(S^n = s^n) \\ &\times \mathsf{Pr}(A[n+1] = a[n+1]|S[n] = s[n]) \\ &\times \prod_{i \in A[n+1]} \mathsf{Pr}(T_i[n+1] = t_i[n+1]|S^n = s^n)\end{aligned} \tag{15}$$

where $\mathsf{Pr}(A[n+1] = a[n+1]|S[n])$ is the transition probability of the recruitment model presented in Section II-C2; $\mathsf{Pr}(T_i[n+1] = t_i[n+1]|S^n)$ depends on the renewal model presented in Section II-C1.

For the i-th MU, the possible bifurcations of the sawtooth sequence are $t_i^{n+1} = \{t_i^n, t_i[n]+1\}$ and $t_i^{n+1} = \{t_i^n, 0\}$ if $t_i^n > t_{\text{R}}$. The sawtooth sequence is $t_i^{n+1} = \{t_i^n, t_i[n]+1\}$ if $t_i^n \leq t_{\text{R}}$. Therefore, the total number of possible bifurcations from one scenario varies from 1 to $2^{\text{card}(A[n+1])}$, where $\text{card}(A[n+1])$ denotes the number of elements in the $A[n+1]$.

D. Tracking

To make the algorithm adaptive to non-stationary inter-spike laws parameters Θ and impulse responses H, we introduce a window length sequence $\ell[n]$ [22] growing from 1 to the maximum window length ℓ_∞ related to the desired adaptivity:

$$\begin{cases}\ell[1] = 1 \\ \ell[n+1] = (1 - \frac{1}{\ell_\infty})\ \ell[n] + 1\end{cases} \tag{16}$$

The formula of the estimated impulse response (12) becomes:

$$\hat{H}_{i,S^n}^{|n} = \left(1 - \frac{1}{\ell[n]}\right) \hat{H}_{i,S^{n-1}}^{|n-1} + \frac{m_{\Delta,i}[n]\varphi_i(S[n])\epsilon[n]}{\ell[n]\ \tilde{v}[n]} \tag{17}$$

And the formula of the estimated variance of noise (10) is rewritten as:

$$\hat{V}_{S^n}^{|n} = \left(1 - \frac{1}{\ell[n]}\right) \hat{V}_{S^{n-1}}^{|n-1} + \frac{1}{\ell[n]}(Y[n] - \psi(S[n])\ \hat{H}_{S^n}^{|n})^2 \tag{18}$$

The window length sequence was also used in the estimation of inter-spike law parameters [16]. We do not repeat the formula here.

E. Initialisation

At the beginning of the decomposition, we assume that there are no active MUs. Therefore, the set of active MUs indexes $A[1]$ and the sawtooth sequence $T[1]$ are empty. Initial rough estimates of impulse responses $\hat{H}_{S^1}^{|0}$ are manually or automatically extracted using other techniques, e.g. proposed in [23]–[25]. An initial estimation of the noise variance $\hat{V}_{s^0}^{|0}$ is made by using a signal extract containing no spikes. The initial ISI distribution law parameters of active MUs $\hat{\theta}_{i,S^0}$ are composed of t_0 (typically $3t_{\text{R}} \sim 4t_{\text{R}}$) and β (typically $2 \sim 4$) according to the our experience. Finally, n_{path} initial S^1 are weighted with the same initial probability $\mathsf{Pr}^{|0}(S^1 = s^1)$.

IV. Path Pruning

As it was previously shown in Section III-C, the number of possible scenarios for S^n grows exponentially with time, due to its bifurcation. Thus, an exhaustive search for the optimal scenario is impossible. In this section we propose original methods to discard unnecessary scenarios.

A. Limiting the Number of Kept Paths

A conventional measure to deal with the large number of scenarios is to limit the number of kept paths. The n_{path} most probable scenarios are kept at every time index, where n_{path} is chosen as a trade-off between the computational complexity and the sub-optimality of the solution.

B. Pruning Based on Activity Detection

An iEMG signal, especially during low-force contractions, comprises short prominent action potentials separated by relatively long segments of background noise. It is possible to avoid

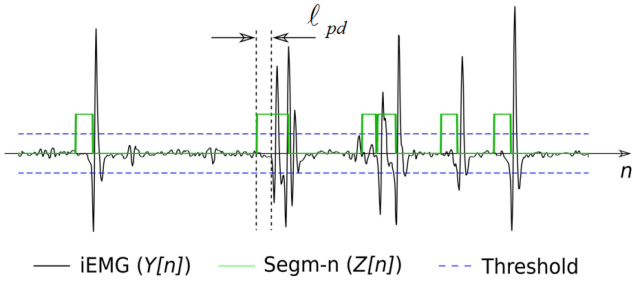

Fig. 2. Example of iEMG segmentation. Segments are detected using a threshold and shifted in time to the left by ℓ_{PD} due to the use of future samples. Bifurcations containing impulses are forbidden while $Z[n] = 0$.

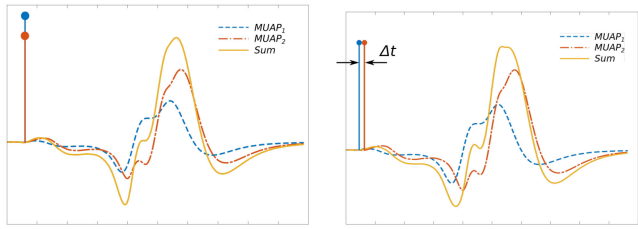

Fig. 3. Two close cases of MUAP superposition: (a) - exact superposition of two spikes, a case considered rare and thus excluded from the search; (b) - a close superposition case (Δt denotes the sampling period).

performing the bifurcations of S^n during these inactive segments in order to reduce the computational time.

We adopted a measure similar to the signal segmentation presented in [23], [24]. Peaks in EMG that exceed a certain predefined threshold, are considered as segments of signal containing MUAPs. In our algorithm, we introduce $Z[n]$ which represents the output of a pre-detection function $z(Y[n+1 : n+\ell_{\text{PD}}])$, where ℓ_{PD} denotes the length of pre-detection and is typically set to $\ell_{\text{IR}}/2$ where ℓ_{IR} is the length of MUAPs. If a MUAP or a superposition is detected in the upcoming signal, the pre-detection function returns "1" authorising S^n to bifurcate; otherwise, it returns "0" and prevents S^n from bifurcating. An example is provided in Fig. 2.

We also note that this approach introduces a delay of $\ell_{\text{IR}}/2$ samples in the decomposition process. Generally, it can vary between 2.5 and 5 ms, which can be considered as a negligible delay in most applications.

An exact implementation of the function $z(Y[n+1 : n+\ell_{\text{IR}}])$ is beyond the scope of this paper. Here, we only note that any convenient EMG segmentation method can be used. In our implementation, an adaptive spike-detection threshold from [24] was used.

C. Simultaneous Spikes Interdiction

The simultaneous occurrence of two or more spikes at exactly the same time instant is highly improbable. As an example, considering a sampling frequency of 10 kHz and ten active MUs with mean ISIs of 100 ms, the probability of having spikes occurring at a specific instant of time, given that one spike already occurs at the same instant, is $1-(1-1/1000)(1-2/1000)\ldots(1-9/1000) = 0.044$.

Furthermore, we consider the negative impact of this heuristic on the solution to be negligible compared to the gain in computational speed. The impact is illustrated in Fig. 3 where an exact superposition (a) can be anyway resolved as its closest possible version (b). Since the superposition shapes in both cases are almost identical, especially for high sampling frequencies, the effect of this heuristic on the MUAP estimates can be neglected. The gain in computation speed is reached due to the fact that the maximal number of possible bifurcation at step n reduces from $n_{\text{path}} \times 2^{\text{card}(A[n+1])}$ to $n_{\text{path}} \times (\text{card}(A[n+1])+1)$.

V. Parallelism Analysis

In the last ten years, we have entered the epoch of GPU computing. The GPU computation is taking a relatively important place in the field of high-performance computing and is applied in a large number of applications in order to achieve superior efficiency. In this section, we analyze the parallelism of the iEMG signal decomposition model and then implement it into the GPU parallel computation.

Based on the HMM model and Bayes filter established in Sections II and III, the structure of iEMG signal decomposition at the time index n, for all $n \geq 1$, is:

1) Data transmission: the iEMG signal $Y[n]$.
2) Calculation of posterior probabilities $\mathsf{Pr}^{|n}(S^n = s^n)$ of scenarios with formula (14).
3) Sorting the posterior probabilities of scenarios and keeping the n_{path} most probable scenarios.
4) Update of the inter-spike law parameters $(\hat{\theta}_{i,S^n})_{i\in\omega}$ with the RML estimator (see Section III-A).
5) Update of the impulse responses $(\hat{H}^{|n}_{S^n})_{i\in\omega}$ and the variance of noise $\hat{V}^{|n}$ with formulas (12), (17), (18), and (11).
6) Activation and inactivation of MUs with respect to the recruitment model in Section II-C2.
7) Bifurcation of the scenarios and calculation of the priori probabilities $\mathsf{Pr}^{|n}(S^{n+1} = s^{n+1})$ of the scenarios with formulas (15).
8) Prediction of the observed signal $\hat{Y}^{|n}_{S^{n+1}}$ and of the variance of the innovation $v_{S^{n+1}}$ with formulas (8) and (13)
9) Data transmission: the state vector at time index n.

The estimation of the state vector can be roughly interpreted as a loop-based pattern [26], whose performance in the parallel computing structure varies in terms of the dependencies between loop iterations and the work partition between the available processors. However, this is never the case since the Bayes filter is a recursive estimation and therefore it is impossible to remove the dependencies between loop iterations. We must calculate them in strictly sequential manner. Therefore, we need to analyze the parallelism in each iteration.

In each iteration, the decomposition process can be separated into a number of single tasks (kernel functions) executed in parallel. In each task, the data can be processed in parallel. In the following sections, we will analyze the structure of the

decomposition algorithm to minimize communication between processors and to maximize the use of on-chip resources.

A. Data Parallelism

Data parallelism is a form of parallelization based on data. It focuses on the distribution of data in the different processors that execute the same operation in parallel [26]:

- Paths (or scenarios) on parallel: Before the bifurcation of sawtooth sequences $T[n]$, there are n_{path} paths, which are mutually independent. After the bifurcation, all new paths remain independent. So calculations in all paths could be implemented in the parallel structure with less communication between them.
- MUs on parallel: According to the hypothesis of the Markov model, there is no dependency between any two MUs. Therefore, in every path, the calculation of all MUs can be executed simultaneously.
- Operation in parallel: In every single task, such as: the estimation of inter-spike law parameters and impulse responses, several operations as sum of vector or matrix multiplication can be calculated in parallel.

B. Task Parallelism

Task parallelism is another parallelization that contrasts data parallelism [26]. Rather than simultaneously computing the same function on several data elements in data parallelism, task parallelism consists in performing two or more completely different tasks in parallel. In the structure of iEMG signal decomposition, the simultaneous execution of tasks is limited by the dependences between them.

In each iteration, the data transfer takes place twice: data transfer of observed signal $Y[n]$ from host (CPU) to device (GPU) (task 1) and data transfer of state vector from device to host (task 9). The overlap of two types of memory copy and the computation on GPU can be achieved. As a result, the time for data transfer is covered by the execution time of other kernel functions.

Furthermore, some parallel computing architectures support concurrent kernel execution [27], [28], where different small kernels of the same application context can be executed at the same time to ensure the full use of the GPU resources. According to the structure of the Bayes filter presented in Section III-A, the PDFs of $\Theta[n]$ and $H[n]$ do not depend on each other. Therefore, in every loop, the tasks related to the estimate of the inter-spike law parameters $\hat{\theta}_{i,S^n}$ can be executed simultaneously with the ones related to impulse responses $\hat{H}_{S^n}^{|n}$. Thus, tasks 4 and 5, as well as tasks 7 and 8, can be calculated at the same time.

C. Task Analysis

Some of these decomposition tasks need to be analyzed in the parallel environment: Task 3 is related to a classic parallel sorting problem, analyzed in Section V-C1; Task 7 (bifurcation of sawtooth sequences), which changes the size of parallel structure, also deserve more consideration, as shown in Section V-C2.

1) Parallel Sorting: After the bifurcation of sawtooth sequences, with respect to the transition distribution presented in Sections II-C1 and II-C2, there are usually at most $n_{\text{path}} \times 2^{\text{card}(A[n+1])}$ paths. The size of parallel sorting problem varies from n_{path} to $n_{\text{path}} \times 2^{\text{card}(A[n+1])}$. With the interdiction of simultaneous spikes presented in Section IV-C, the maximum number of bifurcations reduces to $n_{\text{path}} \times (\text{card}(A[n+1])+1)$.

For small sequences, bitonic sorting is usually considered as one of the fastest traditional parallel sorting algorithms [29], [30]. The time complexity of bitonic sorting is $O(n\ \log_2^2 n)$, while in the parallel environment, it's $O(\log_2^2 n)$ [31].

The most important operation of the bitonic sorting is the arrangement of a bitonic sequence, comprising an ascending sequence and a descending one, into a sorted sequence. In task 3, the final objective is to keep the n_{path} most probable scenarios. Therefore, in the bitonic sequence, if the size of the ascending one and the descending one are more than n_{path}, we only keep the n_{path} biggest values in the two sequences to form the bitonic sequence. This measure can remove parts of unnecessary sorting.

2) Indexes of Bifurcation: Path $S[n]$ bifurcates in at most $A[n+1]+1$ different ways giving an overall number of $n_{\text{path}} \times (A[n+1]+1)$ of new paths. After the parallel sorting, we only keep the n_{path} most probable new paths at time index $n+1$. To avoid the memory allocation and initialization of each bifurcation originated from one path, indexing is used.

Here is an example for two active motor neurons, which gives a two-dimensional vector $\mathbf{T}[n]$ and three possible bifurcations (the used values are arbitrary):

$$\text{if } \mathbf{T}[n] = \begin{bmatrix} 450 \\ 635 \end{bmatrix}, \quad \mathbf{T}[n+1] \in \left\{ \begin{bmatrix} 451 \\ 636 \end{bmatrix}, \begin{bmatrix} 0 \\ 636 \end{bmatrix}, \begin{bmatrix} 451 \\ 0 \end{bmatrix} \right\} \tag{19}$$

Each i-th motor unit can either not fire at time $n+1$ ($T_i[n+1] = T_i[n]+1$) or fire if ready ($T_i[n+1]=0$). Therefore, a binary code can be associated to each configuration in $\boldsymbol{T}$.

$$\mathbf{T}[n+1] \mapsto \begin{bmatrix} 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}; \tag{20}$$

This code is unique for each bifurcation within a scenario.

Therefore, in task 7, we initialize the indexes instead of the bifurcation. After sorting the bifurcations and keeping the n_{path} most probable paths at time index $n+1$, according to the unique index of every bifurcation kept, we initialize the new scenarios.

D. Parallel Structure

As presented in Section IV-B, $Z[n]$ is the indication of the bifurcation of S^n. If $Z[n]=0$, S^n does not bifurcate, means that $t[n]=t[n-1]+1$ and $\hat{Y}_{S^n}^{|n-1}=0$. Hence, we do not need to bifurcate scenarios (task 7) and predict $\hat{Y}_{S^n}^{|n-1}$ (task 8) at time index $n-1$. At the next time index, sorting the posterior probabilities of scenarios and keeping the n_{path} most probable scenarios (Task 3) are skipped, because after the bifurcation, the number of scenarios does not change. Moreover, the update of impulse responses (Task 5) is not needed.

With the parallelism analysis presented above, the parallel structure is illustrated in Fig. 4.

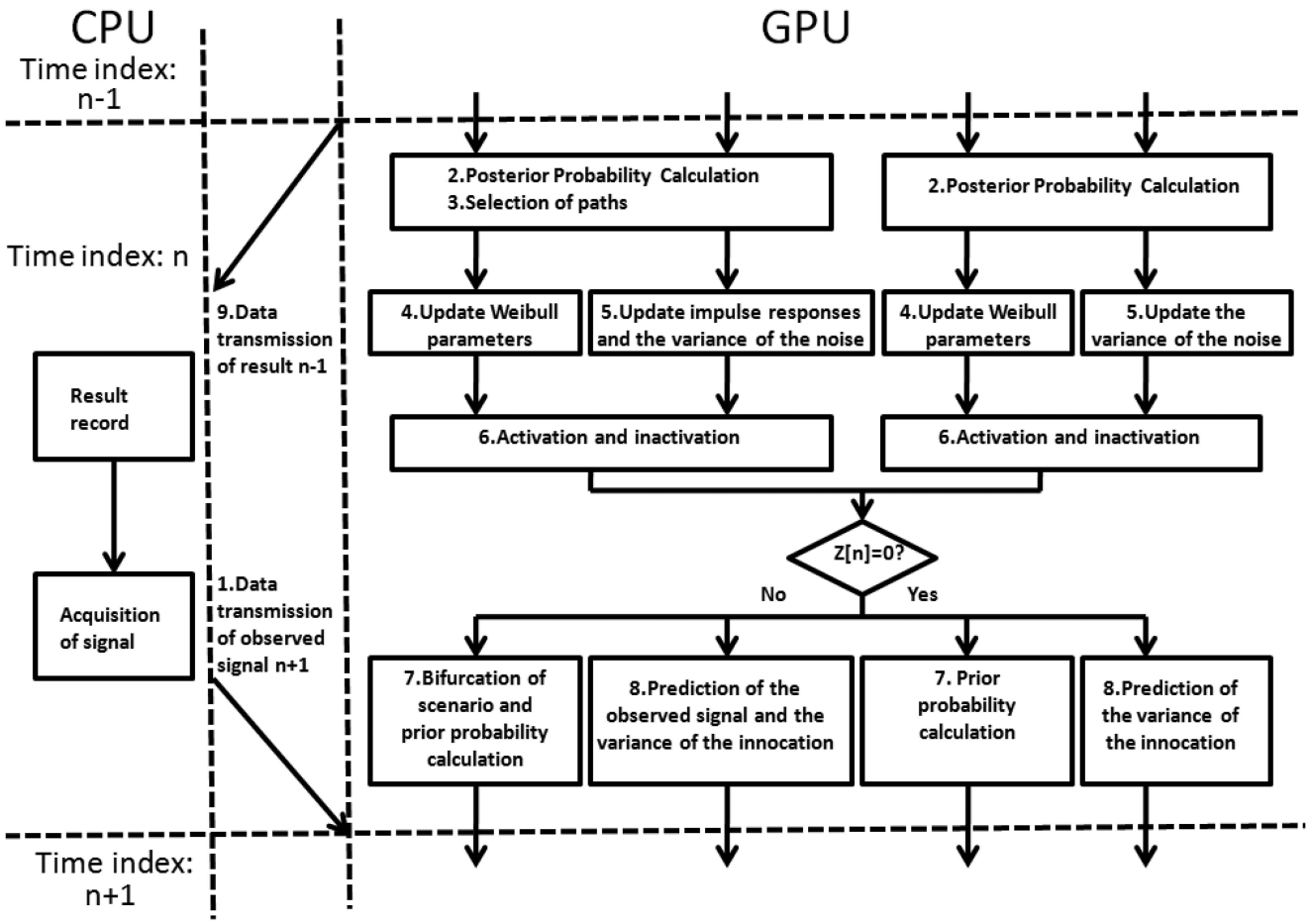

Fig. 4. Parallel structure of iEMG signal decomposition algorithm.

E. Performance Analysis

In this section, we analyse the performance of parallel algorithm implementation. As a test signal, we use ten seconds of simulated iEMG with 6 active MUs, generated at sampling frequency of 5 kHz. This signal was decomposed by the proposed algorithm using 256 paths. The parallel decomposition algorithm was programmed in C++ CUDA language. The parallel computation was implemented in the software VISUAL STUDIO 2013 and CUDA 8.0 on a laptop with Central Processing Unit: Intel(R) Core(TM) i7-4700MQ CPU @ 2.40 GHz and Memory Installed RAM 16.00 Go; Graphics Processing Unit: Intel(R) HD Graphics 4600 and NVIDIA GeForce GTX 780 M. And the performance analysis was provided by the software Nsight Visual Studio Edition 5.2.

Due to the fact that the GPU that was used to produce the experimental results presented below (NVIDIA Tesla K80) is located in a remote data center, it was not possible for us to collect the analysis profile information with Nsight elipse on it. However, our implementation still demonstrates the required parallelism characteristics when run on an older laptop NVIDIA GeForce GTX 780 M.

An extract of the timeline provided by Nsight is shown in Fig. 5. There are four streams in this figure. Stream 1 is the default one, in which all kernel functions are completely synchronous, while the other three streams were designed for asynchronous operations. As presented in Fig. 4, Stream 1 contains three synchronous kernel functions: posterior probability calculation (Task 2), initialization of new scenarios (part of Task 3, see Section V-C2), and activation-inactivation of MUs (Task 6). Stream 2 is composed of bitonic sorting (part of Task 3, see Section V-C1), update of Weibull parameters (Task 4), bifurcation of scenarios and prior probability calculation (Task 7). Stream 3 is designed for device-to-host and host-to-device memory copies (Tasks 1 and 9). Update of the MUAP shapes and of the variance of noise (Task 5), prediction of the observed signal and calculation of the variance of innovation (Task 8) are located in Stream 4. As shown in Fig. 5, all the task parallelisms proposed in Section V-B, including the overlap of two types of memory copy and the computation on GPU, as well as the concurrent execution of different tasks, are achieved.

As shown in Fig. 5, different tasks contain different numbers of kernel functions. Task 3 is composed of six kernel functions: three for bitonic sorting (one chosen in function of sorting problem complexity), one for scenario selection, and two for scenario initialization (one for $S[n]$ and $\Theta[n]$, one for $H[n]$). Task 7 contains two kernel functions: one for the calculation of the hazard rate r (see function (4)) and the other for the calculation

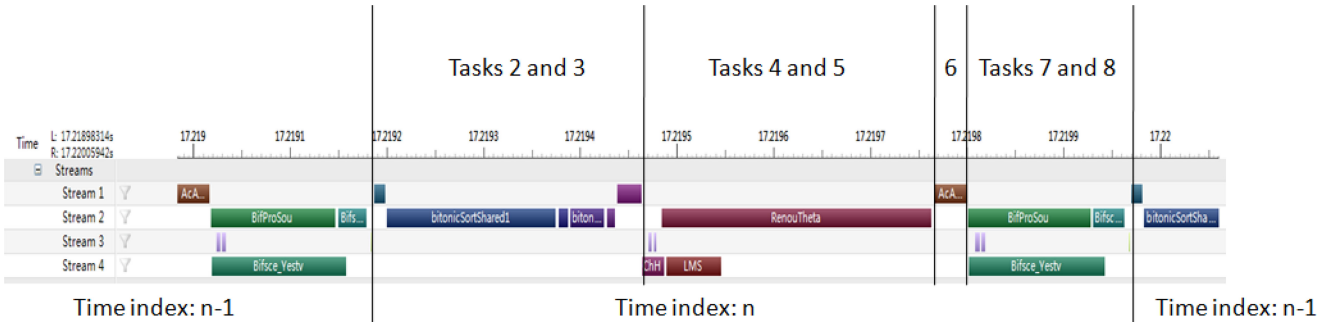

Fig. 5. Extract from the performance analysis timeline provided by Nsight profiler.

TABLE II
WORKLOAD OF DIFFERENT TASKS IN GPU

Task Index	Workload(%)
Task 2	1.40
Task 3	14.38
Task 4	42.49
Task 5	6.01
Task 6	5.36
Task 7	20.6
Task 8	8.15

of the prior probability of the new bifurcated scenario. The rest of the tasks contain only one kernel function each.

The workloads of different tasks in GPU are shown in Table II. A workload is influenced by the complexity of the task and the number of calls. Update of Weibull parameters (Task 4) takes the largest workload in all tasks. It is called in every iteration, disregarding the pre-detection value $Z[n]$. Other tasks, such as selection of paths (Task 3) and so on, are simplified in the case of $Z[n] = 0$. Moreover, as presented in Section III-A, Task 4 is based on the Weibull parameters estimation method presented in [16], which shows a relatively high complexity compared to other tasks.

Since Task 4 takes the highest workload in the decomposition, we have studied its profile, including the instruction statistic, issue efficiency and achieved occupancy. Its grid dimension and block dimension are respectively $\{8, 1, 1\}$ and $\{192, 1, 1\}$. Every block is allocated 4608 byte shared memories and 9216 byte registers. In the instruction statistic, all stream multiprocessors are active during 93.5% duration of the kernel launch. All stream multiprocessors reach the uniform value of instructions per warp and launch the same number of warps.

However, this kernel function shows low achieved occupancy (9.25%) and low percentage of eligible warp issue efficiency (6.83%), which means that it is difficult to hide latency between dependent instructions. Such low profile indexes are determined by the Weibull parameters estimation method [16]. In this method, the formula of estimation changes with respect to the different cases $T[n]$ and $T[n-1]$. Thus, several *if-else* statements used in the kernel function make the workload between and across blocks unbalanced, causing the issue stall. In order to address this issue and further accelerate the decomposition, the mathematical model of algorithm should be accordingly adapted, which can be one of the perspectives.

The execution time for this 10-s signal is 25.85 s on NVIDIA GeForce GTX 780 M card (computational power 3.0), which cannot reach the real-time decomposition. Execution time on Nvidia Tesla K80 GPU card (computational power 3.7) for the same signal is 6.34 s.

VI. EXPERIMENTAL AND SIMULATION PROTOCOLS

A. Signals

Three groups of simulated signals were generated by the described Markov model with respectively 6, 8 and 10 active MUs. There were 10 signals in each group. The sampling frequency was set to 5 kHz and the duration was 20 s. MUAP shapes extracted from the experimental iEMG signals were used to make the simulated signals more realistic. For the statistic parameters of ISI, the refractory period was chosen to be 30 ms; the location parameter t_0 ranged from 60 ms to 90 ms; and the concentration parameter β ranged from 2 to 6. The SNR (Signal to Noise Ratio) was set to 10 dB.

Five experimental signals were acquired from the tibialis anterior (TA) muscle of a 26 years-old healthy man. The subject performed five trials of an isometric force by tracking a trapezoidal profile with target force set to 20% or 30% of the maximal voluntary contraction (MVC). The duration of each trail was 24 s. The wire electrodes used for these recordings were made of Teflon coated stainless steel (50 um diameter; A-M Systems, Carlsborg, WA, USA) and inserted into the muscle with 25 G needles. The signals were amplified, band-pass filtered between 100 Hz and 4.4 kHz and sampled at a frequency of 10 kHz (OTBioelettronica MEBA amplifier). Then they were subsequently down-sampled to 5 kHz.

Parallel computation algorithm was applied to decode the simulated and the experimental signals. The activation probability λ and the maximum time t_I were respectively set to 0.03 and $7t_R$; The window length corresponding to the adaptivity was 1.4 s. The number of selected paths was set to 128, 256, 384, and 512.

B. Indexes of Performance and Task Complexity

Results of automatic decomposition were evaluated in terms of similarity between the reference spike trains and those obtained by the algorithm. In the case of experimental signals, the reference was a manual decomposition provided off-line by

TABLE III
DECOMPOSITION PERFORMANCE OF SIMULATED SIGNALS: "NB MUS" IS THE MAXIMAL NUMBER OF MUS CONCURRENTLY ACTIVE IN THE SIGNAL; "NB SUP-SPIKES" REPRESENTS THE NUMBER OF SPIKES INVOLVED IN SUPERPOSITIONS; "NB SPIKES" DENOTES THE OVERALL NUMBER OF SPIKES IN THE SIGNAL; "SUP." IS THE PERCENTAGE OF SUPERPOSITION; "NB PATHS" IS THE NUMBER OF PATHS USED IN THE ALGORITHM; "SENS.' DENOTES THE GLOBAL SENSITIVITY; "PRED." IS THE GLOBAL PREDICTIVITY

Nb MUs	Nb sup-spikes	Nb spikes	Sup.(%)	Nb paths	Sens. (%)	Pred. (%)	Time(s)
10	645.20±39.90	2093.10±80.59	30.83±1.91	384	95.15±2.47	93.38±2.40	20.60±0.82
				256	93.38±2.88	88.40±2.89	17.23±0.66
				128	80.85±7.02	80.78±5.63	14.43±0.48
8	446.40±28.52	1769.80±59.44	25.22±1.61	384	97.32±1.79	96.74±2.15	16.13±0.57
				256	96.55±2.17	95.13±2.15	15.29±0.50
6	291.70±15.27	1469.80±52.49	19.85±1.04	384	99.05±1.08	98.63±1.44	14.75±0.76
				256	98.78±1.19	98.23±1.70	13.26±0.61

TABLE IV
EXECUTION TIME AND ACCELERATION FACTOR OF SIMULATED SIGNALS

Nb MUs	Nb paths	GPU execution time (s)	CPU execution time (s)	Acceleration factor
10	384	20.60±0.82	251.53±6.02	12.21
	256	17.23±0.66	155.26±3.88	9.01
	128	14.43±0.48	70.00±2.91	4.85
8	384	16.13±0.57	195.40±2.20	12.11
	256	15.29±0.50	119.69±1.65	7.83
6	384	14.75±0.76	165.71±5.43	11.23
	256	13.26±0.61	96.50±3.06	7.28

TABLE V
DECOMPOSITION PERFORMANCE FOR EXPERIMENTAL SIGNALS. THE MEANING OF INDEXES ARE THE SAME AS TABLE III

Index	Duration (s)	Force (MVC%)	Nb MUs	Nb spikes	Sup.(%)	Sens. (%)	Pred.(%)	Time (s)		
								256(Nb paths)	384	512
1	24	20	5	873	18.10	92.67	91.72	13.47	14.30	15.16
2	24	20	5	936	18.38	94.87	93.77	13.69	14.67	15.48
3	24	20	6	933	17.15	94.21	94.41	12.63	13.25	14.20
4	24	30	7	1176	22.28	87.93	86.67	14.40	15.75	17.49
5	24	30	8	1295	28.96	88.73	86.78	14.75	15.32	18.51

an expert operator using the publicly available decomposition software EMGLAB [32]. In case of simulated signals, the exact spike trains were known from the simulation procedure

In order to characterise the complexity of the decomposition task, we used the superposition percentage as in our previous work [16]:

$$\text{Sup} = \frac{\text{Nb}_{\text{SUP}}}{\text{Nb}_{\text{SPIKES}}} \tag{21}$$

where $\text{Nb}_{\text{SPIKES}}$ is the number of spikes in the reference spike train and Nb_{SUP} is the number of spikes which action potentials are superposed with others. We considered a MUAP superimposed if there was at least one other MUAP within a margin of 3 ms (less than half of the average MUAP duration) around it.

In order to quantitatively evaluate the decomposition results, we used global sensitivity and global positive predictivity values, defined as following. A MUAP was considered correctly identified (true positive) if the reference train contained a spike from the same MU within a margin of 1 ms around it. Consequently, the global sensitivity was defined as the overall number of correctly identified MUAPs from all MUs, divided by the overall number of spikes in the reference decomposition. Global positive predictivity was the number of correctly identified spikes divided by the overall number of spikes in the decomposition under evaluation.

An individual analysis of each MUAP train was also performed, using the "classification phase" indexes proposed in [33]. These indexes included sensitivity, specificity and accuracy, as defined in [33].

VII. RESULTS

All results presented in this section were obtained using decomposition programmed in C++ CUDA language, were performed on a Nvidia Tesla K80 GPU card with CUDA 10.0 and GCC 4.9.3 using double-precision floating-point format.

A. Simulated Signals

As shown in Table III, three groups of simulated signals with 6, 8 and 10 MUs were decomposed. We note that the mean values of global sensitivity and predictivity (Table III) decreased for signals with greater number of active MUs. This is due to the increase of decomposition task complexity, quantified by the superposition percentage. Moreover, the standard deviations of the performance indexes showed proportionality to the task complexity. We also observe that greater numbers of paths n_{path} mitigate this effect.

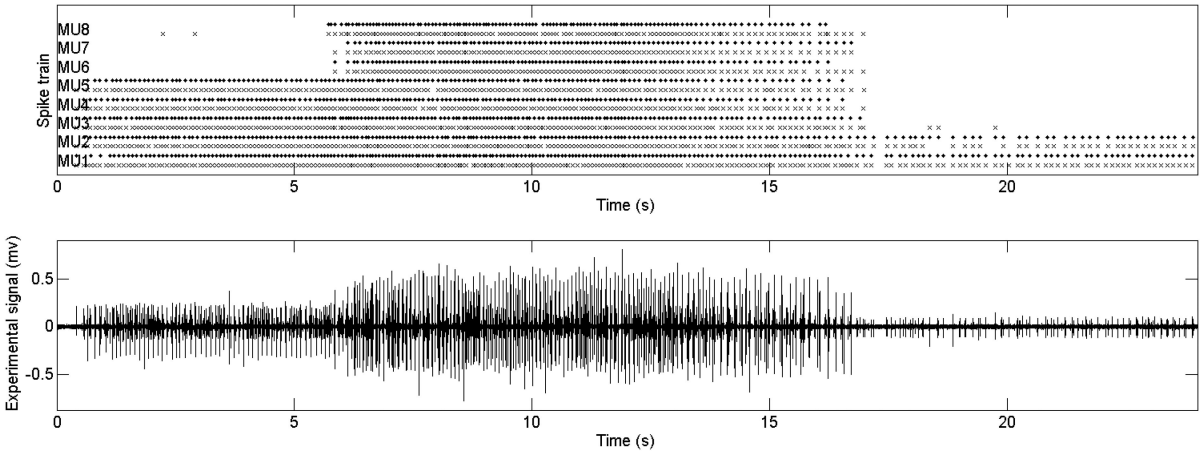

Fig. 6. Comparison of automatic (crosses, "x") and reference (points, ".") decompositions (upper panel) and the experimental signal from TA, 30% MVC (lower panel).

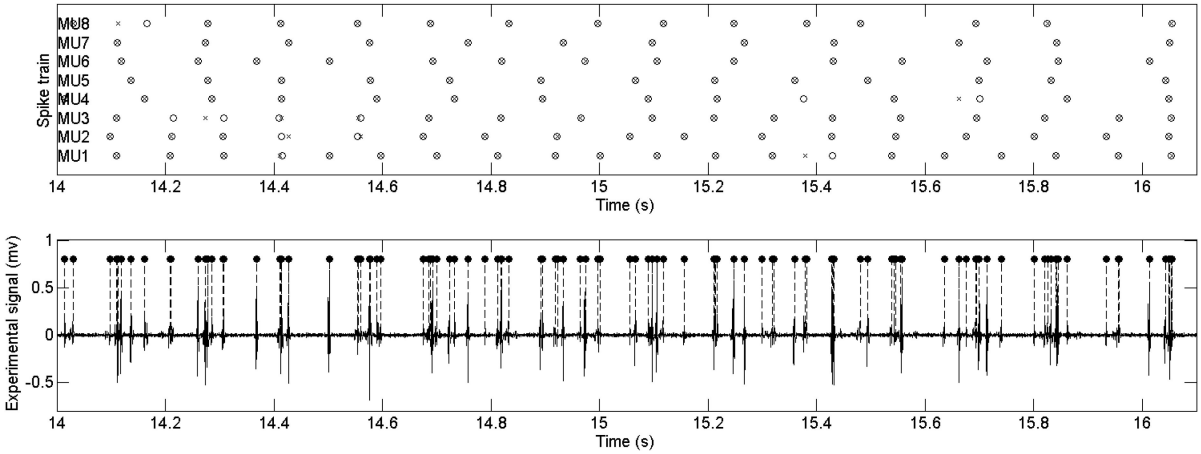

Fig. 7. An extract of the experimental signal decomposition shown in Fig. 6; circles "○" and crosses "x" represent respectively the spikes from the reference and automatic decompositions.

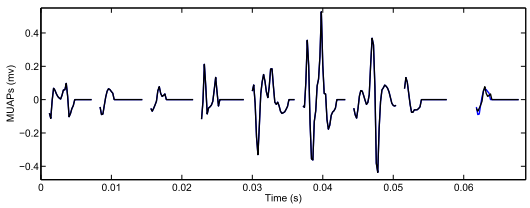

Fig. 8. Eight MUAP shapes (manually-extracted dictionary) for the signal presented in Fig. 6, and a comparison between the 2nd one and the 3rd one.

The execution time becomes larger with the increasing number of paths and active MUs. The signals with 10 MUs, 8 MUs, and 6 MUs are decomposed in real time, with respectively 256 paths, 384 and 384 paths. More complex decompositions cannot be accomplished in real time using the computational resources used in this study. However, they still can be accomplished in a relatively short time and with high accuracy. Thus, the number of paths n_{path}, as a parameter determined by the user, defines both the decomposition accuracy and speed. Its value establishes a trade-off between the computational complexity (which converts into decomposition time) and the sub-optimality of the solution.

Table IV shows the execution time of simulated signals decomposition in CPU and GPU, as well as their acceleration factor. The decomposition algorithm in CPU was programmed in C++ and performed on the 12-core Intel Xeon (Haswell) E5-2680v3 processor with GCC 4.9.3. As shown in table, the acceleration factors, defined as the ratio of execution time between CPU and GPU, increase from 4.85 to 12.21 with increasing number of MUs and paths, indicating that the parallel decomposition algorithm leads to greater accelerations for more complex decompositions. Furthermore, we point out that the parallel decomposition in this paper is more than 100 times faster than the one proposed in our previous study [16].

B. Experimental Signals

Five experimental signals (three recorded at 20% MVC, two recorded at 30% MVC) were automatically decomposed. As

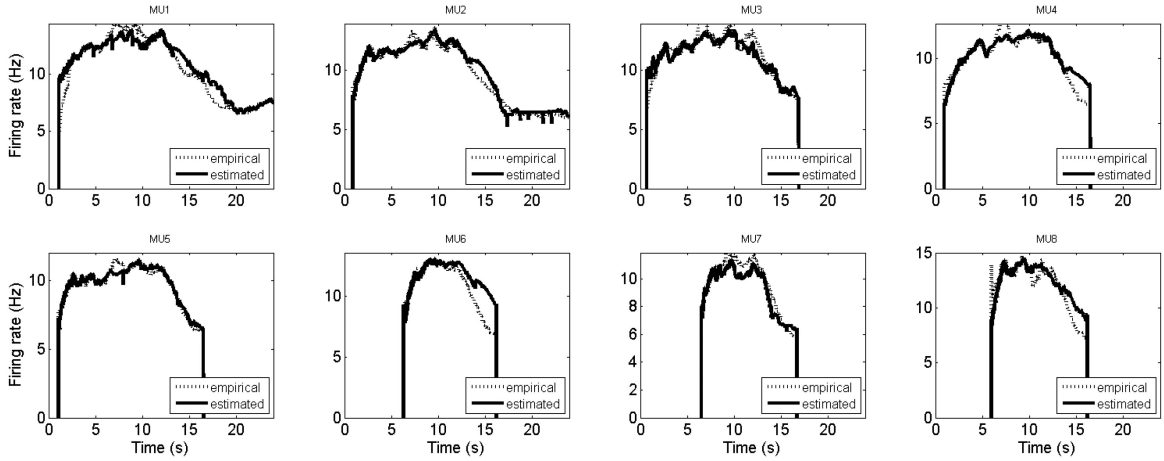

Fig. 9. Firing rates for the iEMG from TA set (see Fig. 6): the dash line (empirical) represents the firing rates estimated using reference decomposition; continuous line (estimated) represents the firing rates calculated via parameters of discrete Weibull distribution estimated as described in [15], [16].

TABLE VI
MAXIMUM DELAY OF EXPERIMENTAL SIGNALS DECOMPOSITION: THE SIGNAL INDEX CORRESPONDS TO THE SIGNALS PRESENTED IN TABLE V

Index	1	2	3	4	5
Nb paths	512	512	512	512	512
Max Delay (ms)	5.2	5.6	4.9	41.5	67.5

TABLE VII
DECOMPOSITION PERFORMANCE FOR AN EXPERIMENTAL SIGNAL DETECTED FROM THE TA WITH 8 MUS: FOR EACH MU, "SENS." DENOTES THE SENSITIVITY; "PRED." IS THE PREDICTIVITY; "ACC." REPRESENTS THE ACCURACY

MU	Sens.	Spec.	Acc.
MU1	91.13	97.36	96.07
MU2	80.54	95.67	92.96
MU3	76.88	95.63	92.81
MU4	91.87	98.33	97.46
MU5	97.35	99.40	99.14
MU6	99.07	99.43	99.39
MU7	98.99	99.53	99.48
MU8	84.43	97.57	96.23

shown in Table V, for these signals, the number of MUs ranged from five to eight and the percentage of superposition ranged from 18.10% to 28.96%. Both the global sensitivity and predictivity of the three signals recorded at 20% MVC were above 90%, while the global sensitivity and predictivity of the other two signals were above 85%. The decomposition accuracy did not vary significantly in function of number of paths. In Table V, we also notice that all the experimental signals can be decomposed automatically in real time using 256, 384 or 512 paths.

Let us denote the time delay between passing a new EMG sample to the decomposition program and receiving the corresponding decomposition result at its output as *decomposition latency*. Its value can change during decomposition as a function of the signal complexity. Table VI shows the maximal decomposition latencies for experimental signals. Their values are smaller than acceptable delay in many applications of man-machine interfacing, e.g. delays up to 250 ms are usually considered acceptable for prosthetic control [34].

Detailed results of the decomposition are illustrated and analyzed in the following for the signal with 8 MUs, the most representative and complex one. These results were obtained from decomposition with 384 paths.

Fig. 6 provides global view of the decomposition results. In the upper panel, the decomposed spike timings (crosses) of each MU are correlated with the reference (points); in the lower panel, the corresponding iEMG signal is shown. A detailed view of the decomposition results is given in Fig. 7, containing two seconds of extracted signal. The algorithm performed generally well, successfully processing several complex superpositions. Due to the high complexity of the signal, there were a few mistakes in the classification. As an example, we note two misclassification cases occurred at 14.4 s and 14.55 s (see upper panel of Fig. 7).

For the classification phase, the individual (per MU) performance indexes are shown in Table VII. Fig. 8 illustrates the MUAP waveforms of eight MUs. The last one is the comparison of MUAP waveforms between the 2nd one and the 3rd one. According to Fig. 8, we analyze the performance indexes in Table VII. The reason for the lower sensitivity of the 2nd and 3rd MU is that they have the smaller amplitudes of MUAPs, compared to the other ones. Generally, this can lead to their complete masking in the superpositions. Moreover, their MUAP waveforms are similar. Thus, decomposition algorithm mistakes occasionally between them, as shown in Fig. 6 (two cases occurred around 18 s, upper panel). With respect to these two MUs, the other MUs are well classified. Globally, the algorithm succeeded in tracking and decomposing the MUs.

The algorithm recursively estimates the parameters of the inter-spike intervals distribution, used to calculate the firing rates. Fig. 9 shows the corresponding firing rates. Empirical ones were estimated as the inverse of the moving average of subsequent inter-spike intervals in the reference decomposition. The estimated ones were calculated with the estimated parameters t_0

and β. The algorithm successfully tracked the changes in firing rates.

VIII. Conclusion and Perspectives

In our previous works [15], [16], a sequential decomposition algorithm based on a HMM of the EMG, that used Bayesian filtering to estimate the unknown parameters of discharge series of motor units was proposed. This algorithm successfully decomposes experimental iEMG signals, however, it requires long computation time.

In this paper we presented a real-time implementation for the previous algorithm, including the replacement of time-consuming Kalman filter by a more computationally efficient LMS filter, three heuristics to reduce the complexity and calculation load, and the implementation of parallel computation. Validations on simulated and experimental signals demonstrated successful performance of the algorithm. The real-time algorithm performance matched that of its offline implementation [16]. Obtained results show that the proposed approach is able of decomposing larger numbers of MUs than its closes analogue [7] (8 MUs instead of 5 for experimental signals in real-time), while providing equal or higher accuracy. This number matches that of another close method [9], however, we note that the last does not provide resolution of superimposed waveforms.

Limitations in the algorithm performance are due to large differences in amplitudes between MUAPs (masking of small action potentials) and similarity in MUAP waveforms (switching between similar MUs). These are common problems in EMG decomposition [35], not specific to the developed algorithm or to its online implementation. A multichannel version of the presented algorithm, which may resolve this problem, is a current topic of our work. Another limitation is the number of MUs that can be simultaneously tracked by the algorithm in the real-time operation (up to 10 MUs), which restricts the approach to low contraction efforts. This limit may be overcome in the future by a better hardware or further modifications of the mathematical model that reduce the calculation complexity.

Appendix A
From Kalman Filter to the Least-Mean-Square Filter

Kalman filter, originally used for MUAPs estimation [15], [16], can be replaced by an LMS filter under specific assumptions. Let's consider the state covariance matrix from (9):

$$\begin{aligned} P_{S^n} &= P_{S^{n-1}} - K_{S^n}\, v_{S^n}\, K_{S^n}^{\top} \\ &= P_{S^{n-1}} - P_{S^{n-1}}\, \psi(S[n])^{\top}\, v_{S^n}^{-1}\, v_{S^n} \\ &\quad \times (P_{S^{n-1}}\, \psi(S[n])^{\top}\, v_{S^n}^{-1})^{\top} \\ &= P_{S^{n-1}} - P_{S^{n-1}}\, \psi(S[n])^{\top}\, v_{S^n}^{-1}\, \psi(S[n]) P_{S^{n-1}} \end{aligned} \tag{22}$$

Applying the Woodbury matrix identity:

$$[A+BCD]^{-1} = A^{-1} - A^{-1}B[DA^{-1}B + C^{-1}]^{-1}DA^{-1} \tag{23}$$

to (22), we obtain:

$$\begin{aligned} P_{S^n}^{-1} = P_{S^{n-1}}^{-1} &+ \psi(S[n])^{\top}\, (v_{S^n} - \psi(S[n])\, P_{S^{n-1}}\, \psi(S[n])^{\top}) \\ &\times \psi(S[n]) \end{aligned} \tag{24}$$

This can be simplified using expression (8) for the variance of innovation:

$$P_{S^n}^{-1} = P_{S^{n-1}}^{-1} + \psi(S[n])^{\top}\, v^{-1}\, \psi(S[n]) \tag{25}$$

where v is the variance of measurement noise $\hat{V}^{|n}$ estimated using (10) and (11). Finally, we have:

$$\begin{aligned} P_{S^n} &= \frac{\hat{V}^{|n}}{n}\, R_{S^n}^{-1} \\ R_{S^n} &= \frac{1}{n} \sum_{k=1}^{n} \psi(S[k])^{\top} \psi(S[n]) \end{aligned} \tag{26}$$

where R_{S^n} can be approximated by a matrix made of $\text{card}(\Omega) \times \text{card}(\Omega)$ blocks R_{i,j,S^n} with dimension $O(\ell_{\text{IR}} \times \ell_{\text{IR}})$:

$$R_{i,i,S^n} = \begin{bmatrix} \xi_{i,S^n} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \xi_{i,S^n} \end{bmatrix} \tag{27}$$

$$R_{i,j,S^n} = \begin{bmatrix} \xi_{i,S^n}\xi_{j,S^n} & \cdots & \xi_{i,S^n}\xi_{j,S^n} \\ \vdots & \ddots & \vdots \\ \xi_{i,S^n}\xi_{j,S^n} & \cdots & \xi_{i,S^n}\xi_{j,S^n} \end{bmatrix} \tag{28}$$

where ξ_{i,S^n} is the firing rate of i-th motor unit, which is the inverse of its inter-spike interval (ISI) expected value. We can notice that $\forall i, j \in \Omega$, $\xi_{i,S^n}\xi_{j,S^n} \ll \xi_{i,S^n}$ and $\xi_{i,S^n}\xi_{j,S^n} \ll \xi_{j,S^n}$. Therefore, if $i \neq j$, R_{i,j,S^n} can be approximated by a zero-matrix, R_{S^n} can be approximated by a diagonal matrix.

Having the approximation of P_{S^n}, we can derive directly the LMS filter from the Kalman filter (9). With a rough initial prior $\hat{H}_{S^0}^{|0}$, for all $n \geq 1$, we have:

$$\begin{aligned} \epsilon[n] &= Y[n] - \psi(S[n])\, \hat{H}_{S^{n-1}}^{|n-1} \\ m_{\Delta,i}[n] &= \frac{\sum_j \Delta_i[j]}{\text{card}(\Delta_i)} \\ \tilde{v}[n] &= 1 + \frac{1}{n} \sum_i m_{\Delta,i}[n]\, \varphi_i(S[n]) \varphi_i(S[n])^{\top} \\ \hat{H}_{i,S^n}^{|n} &= \hat{H}_{i,S^{n-1}}^{|n-1} + \frac{m_{\Delta,i}[n] \varphi_i(S[n]) \epsilon[n]}{n\, \tilde{v}[n]} \end{aligned} \tag{29}$$

where $\Delta_i[j]$ denotes the j-th ISI of the i-th motor unit; $\text{card}(\Delta_i)$ is the number of the ISIs for the i-th motor unit; $m_{\Delta,i}[n]$ is the expected ISI for i-th motor unit at time index n; and $\tilde{v}[n]$ represents the ratio of the variance of innovation v_{S^n} to the variance of noise $\hat{V}^{|n}$. And the prediction of the variance of innovation v_{S^n} is:

$$v_{S^n} = \tilde{v}[n]\, \hat{V}^{|n}. \tag{30}$$

In order to make this filter adaptive to the changes in MUAPs forms, time index n can be replaced by a forgetting factor $l[n]$.

REFERENCES

[1] D. Ge *et al.*, "Spike sorting by stochastic simulation," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 19, no. 3, pp. 249–259, Jun. 2011.

[2] J. Florestal, P. Mathieu, and K. McGill, "Automatic decomposition of multichannel intramuscular EMG signals," *J. Electromyography Kinesiology*, vol. 19, pp. 1–9, 2009.

[3] H. R. Marateb *et al.*, "Robust decomposition of single-channel intramuscular EMG signals at low force levels," *J. Neural Eng.*, vol. 8, no. 6, 2011, Art. no. 066015.

[4] S. H. Nawab, S.-S. Chang, and C. J. De Luca, "High-yield decomposition of surface EMG signals," *Clin. Neurophysiology*, vol. 121, no. 10, pp. 1602–1615, Oct. 2010.

[5] F. Negro *et al.*, "Multi-channel intramuscular and surface EMG decomposition by convolutive blind source separation," *J. Neural Eng.*, vol. 13, no. 2, 2016, Art. no. 026027.

[6] J. Roussel, P. Ravier, and M. Haritopoulos, "Decomposition of multichannel intramuscular EMG signals by cyclostationary-based blind source separation," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 25, no. 11, pp. 2035–2045, Nov. 2017.

[7] V. Glaser, A. Holobar, and D. Zazula, "Real-time motor unit identification from high-density surface EMG," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 21, no. 6, pp. 949–958, Nov. 2013.

[8] A. Holobar and D. Zazula, "Multichannel blind source separation using convolution kernel compensation," *IEEE Trans. Signal Process*, vol. 55, no. 9, pp. 4487–4496, Sep. 2007.

[9] S. Karimimehr *et al.*, "A real-time method for decoding the neural drive to muscles using single-channel intra-muscular EMG recordings," *Int. J. Neural Syst.*, vol. 27, no. 6, 2017, Art. no. 1750025.

[10] S. E. Paraskevopoulou *et al.*, " Feature extraction using first and second derivative extrema (FSDE) for real-time and hardware-efficient spike sorting," *J. Neuroscience Methods*, vol. 215, no. 1, pp. 29–37, Apr. 2013.

[11] J. Navajas *et al.*, " Minimum requirements for accurate and efficient real-time on-chip spike sorting," *J. Neuroscience Methods*, vol. 230, pp. 51–64, Jun. 2014.

[12] T. Werner *et al.*, "Spiking neural networks based on OxRAM synapses for real-time unsupervised spike sorting," *Frontiers Neuroscience*, vol. 10, p. 474, Nov. 2016.

[13] L. Citi *et al.*, "On the use of wavelet denoising and spike sorting techniques to process electroneurographic signals recorded using intraneural electrodes," *J. Neuroscience Methods*, vol. 172, no. 2, pp. 294–302, Jul. 2008.

[14] D. Pani *et al.*, "Real-time neural signals decoding onto off-the-shelf DSP processors for neuroprosthetic applications," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 24, no. 9, pp. 993–1002, Sep. 2016.

[15] J. Monsifrot *et al.*, "Sequential decoding of intramuscular EMG signals via estimation of a Markov model," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 22, no. 5, pp. 1030–40, Sep. 2014.

[16] T. Yu *et al.*, "Recursive decomposition of electromyographic signals with a varying number of active sources: Bayesian modelling and filtering," *IEEE Trans. Biomed. Eng.*, vol. 67, no. 2, pp. 428–440, Feb. 2020, doi: 10.1109/TBME.2019.2914966.

[17] D. Farina, A. Crosetti, and R. Merletti, "A model for the generation of synthetic intramuscular EMG signals to test decomposition algorithms," *IEEE Trans. Biomed. Eng.*, vol. 48, no. 1, pp. 66–77, Jan. 2001.

[18] D. Stashuk, "EMG signal decomposition: How can it be accomplished and used?" *J. Electromyography Kinesiology*, vol. 11, no. 3, pp. 151–173, 2001.

[19] V. Barbu and N. Limnios, "Reliability theory for discrete-time semi-Markov systems," in *Semi-Markov Chains and Hidden Semi-Markov Models Toward Applications* (ser. Lecture Notes in Statistics, vol. 191). New York, NY, USA: Springer, 2008, pp. 1–30.

[20] C. C. J. Heckman and R. Enoka, "Motor unit," *Comprehensive Physiol*, vol. 2, no. 4, pp. 2629–2682, 2012.

[21] T. Schon, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear nonlinear state-space models," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2279–2289, Jul. 2005.

[22] L. Ljung and T. Söderström, *Theory and Practice of Recursive Identification*. London, U.K.: The MIT Press, 1983.

[23] J. R. Florestal, P. A. Mathieu, and A. Malanda, "Automated decomposition of intramuscular electromyographic signals," *IEEE Trans. Biomed. Eng.*, vol. 53, no. 5, pp. 832–839, May 2006.

[24] K. C. Mcgill, K. L. Cummins, and L. J. Dorfman, "Automatic decomposition of the clinical electromyogram," *IEEE Trans. Biomed. Eng.*, vol. BME-32, no. 7, pp. 470–477, Jul. 1985.

[25] C. Katsis *et al.*, "A novel method for automated EMG decomposition and MUAP classification," *Artif. Intell. Medicine*, vol. 37, pp. 55–64, 2006.

[26] S. Cook, *CUDA Programming: A Developer's Guide to Parallel Computing With GPUs*. Boston, MA, USA: Morgan Kaufmann, 2013.

[27] NVIDIA Corporation, *Whitepaper NVIDIA's Next Generation CUDA Compute Architecture: Fermi*. Santa Clara, CA, USA: NVIDIA Corporation, 2009.

[28] J. Sanders and Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*. Boston, MA, USA: Addison-Wesley Professional, 2010.

[29] A. C. Dusseau *et al.*, "Fast parallel sorting under LogP: Experience with the CM-5," *IEEE Trans. Parallel Distrib. Syst.*, vol. 7, no. 8, pp. 791–805, Aug. 1996.

[30] N. Satish, M. Harris, and M. Garland, "Designing efficient sorting algorithms for Manycore GPUs," in *Proc. 23rd IEEE Int. Parallel Distrib. Process. Symp.*, 2009, pp. 1–10.

[31] A. Greb and G. Zachmann, "GPU-ABiSort: Optimal parallel sorting on stream architectures," in *Proc. 20th IEEE Int. Parallel Distrib. Process. Symp.*, 2006.

[32] K. McGill, Z. Lateva, and H. Marateb, "EMGLAB: An interactive EMG decomposition program," *J. Neuroscience Methods*, vol. 149, no. 2, pp. 121–133, 2005.

[33] D. Farina *et al.*, "Evaluation of intra-muscular EMG signal decomposition algorithms," *J. Electromyography Kinesiology*, vol. 11, pp. 175–187, 2001.

[34] L. H. Smith *et al.*, "Determining the optimal window length for pattern recognition-based myoelectric control: Balancing the competing effects of classification error and controller delay," vol. 19, no. 2, pp. 186–192.

[35] S. H. Nawab, R. P. Wotiz, and C. J. De Luca, "Decomposition of indwelling EMG signals," *J. Appl. Physiol.*, vol. 105, no. 2, pp. 700–710, 2008.