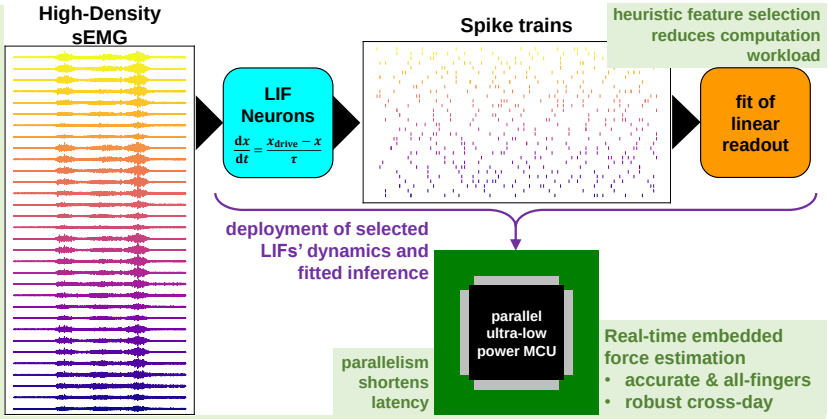
Event-based Estimation of Hand Forces from High-Density Surface EMG on a Parallel Ultra-Low-Power Microcontroller

Marcello Zanghieri*, *Graduate Student Member, IEEE*, Pierangelo Maria Rapa, Mattia Orlandi, *Graduate Student Member, IEEE*, Elisa Donati, *Member, IEEE*, Luca Benini, *Fellow, IEEE*, Simone Benatti, *Member, IEEE*

***Abstract*— Modeling hand kinematics and dynamics is a key goal for research on Human-Machine Interfaces, with surface electromyography (sEMG) being the most commonly used sensing modality. Though under-researched, sEMG regression-based modeling of hand movements and forces is promising for finer control than allowed by mapping to fixed gestures. We present an event-based sEMG encoding for multi-finger force estimation implemented on a microcontroller unit (MCU). We are the first to target the HYSER High-Density (HD)-sEMG dataset in multi-day conditions closest to a real scenario without a fixed force pattern. Our Mean Absolute Error of (8.4 ± 2.8)% of the Maximum Voluntary Contraction (MVC) is on par with State-of-the-Art (SoA) works on easier settings such as within-day, single-finger, or fixed-exercise. We deploy our solution for HYSER's hardest task on a parallel ultra-low power MCU, getting an energy consumption below 6.5 uJ per sample, 2.8× to 11× more energy-efficient than SoA single-core solutions, and a latency below 280 us per sample, shorter than HYSER's HD-sEMG sampling period, thus compatible with real-time operation on embedded devices.**



***Index Terms*— Edge computing, EMG, Event-based, Force estimation, Hand kinematics, High-Density sEMG, Human-Machine Interfaces, Low-power, Machine Learning, Microcontroller, Open source, Parallel Ultra-Low-Power, Regression.**

Submitted on 7 October 2023; revised on 13 December 2023.

This manuscript is submitted as an invited extension of the conference paper M. Zanghieri *et al.*, "Event-based low-power and low-latency regression method for hand kinematics from surface EMG," in *2023 9th IEEE International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2023, pp. 293-298. IEEE Xplore URL: https://ieeexplore.ieee.org/abstract/document/10164372. DOI: 10.1109/IWASI58316.2023.10164372.

*Corresponding author.

M. Zanghieri, P. M. Rapa, M. Orlandi, L. Benini, and S. Benatti are with the Department of Electrical, Electronic, and Information Engineering, University of Bologna, 40136 Bologna, IT (e-mail: {marcello.zanghieri2, pierangelomaria.rapa, mattia.orlandi, luca.benini, simone.benatti}@unibo.it).

E. Donati is with the Institute of Neuroinformatics, University of Zürich and ETH Zürich, 8057 Zürich, CH (e-mail: eldonati@ini.uzh.ch).

L. Benini is also with the Integrated Systems Laboratory, Department of Information Technology and Electrical Engineering, ETH Zürich, 8092 Zürich, CH (e-mail: lbenini@iis.ee.ethz.ch).

S. Benatti is also with the Department of Engineering, University of Modena and Reggio Emilia, 41125 Modena, IT (e-mail: simone.benatti@unimore.it).

This research was supported in part by the EU Horizon Europe project IntelliMan (g.a. 101070136) and by the ETH Zürich's Future Computing Laboratory funded by a donation from Huawei Technologies.

I. Introduction

SURFACE electromyography (sEMG) is nowadays one of the most promising sensing modalities for the control of Human-Machine Interfaces (HMIs) in consumer, industry, and health applications involving wearable control systems for robotics, augmented reality, and prosthetics [1], [2]. State-of-the-Art (SoA) approaches for intuitive control delegate the challenging sEMG-to-command mapping to Machine Learning (ML) and Deep Learning (DL) classification [3], designating discrete gestures for classification [4] or continuous variables for regression [5]. On the one hand, DL classification generally achieves high accuracy thanks to automated learning of data-driven features more informative than their handcrafted counterpart [3], [6]; on the other hand, regression of continuous target variables avoids fixed hand positions, enabling a more versatile and natural control.

Most research so far has focused on classification [7]–[10], and regression studies are still a niche. Just like for classification, the ML revolution in sEMG regression has gathered

momentum from the release of open-access datasets; two major examples are the Non-Invasive Adaptive Prosthetics Database 8 (NinaPro DB8) [11] for hand kinematics based on joint angles, and the High-densitY Surface Electromyogram Recordings (HYSER) dataset for multi-finger forces [12]. Regression approaches have targeted finger forces and hand kinematics, modeling joint angles or velocities. As to kinematics, [11] and [13] have tackled NinaPro DB8, reconstructing the joint angle measurements of a dataglove; in contrast, [14] has targeted joint velocity with a hybrid recognition-regression stratagem that thresholds speed into three levels. As to dynamics, multiple Degrees-of-Freedom (DoF) force estimation was addressed by [2].

The major limitation shared by all the mentioned works is the lack of focus on the execution on resource-constrained computational platforms, which is a fundamental requirement for wearable embedded control systems. This shortcoming is especially relevant for works focusing on DL aiming at a low regression error [15], [16] at the cost of increased memory and computation requirements compared to non-deep ML.

This work proposes a lean, bio-inspired strategy for an event-based encoding of the sEMG for force estimation, implemented and validated on an ultra-low-power microcontroller suitable for embedded control systems. The approach is motivated by our long-term research interest in validating the accuracy and profiling the execution of event-based techniques for future implementation onto event-based computing platforms as an alternative to the dominant DL models relying on matrix multiplication on temporal data buffers [3], [5]. In contrast, event-based computing promises reduced computation latency and energy consumption [17]–[19].

An example of energy-efficient and extreme-low-latency event-based computing is represented by Spiking Neural Networks (SNNs), a class of artificial neural networks able to process spike trains, i.e., binary time series characterized by high sparsity. The spike format is closer to biological neuronal signals than classical DL's activation tensors, moving a step further in the brain-inspired paradigm for intelligent data processing. SNNs' neurons imitate the biological membrane potential, raising their inner state when they receive input events, decaying over time if short of inputs, and *firing* a *spike* upon crossing a threshold, i.e., creating an output event forwarded to the connected neurons [20]. The optimal substrate for executing SNNs are digital [17], [18] or mixed-signal [19] neuromorphic processors, specialized for the sequential emulation of time-varying potentials. In particular, neuromorphic processors implement event-proportional computing [17], [18], which benefits latency and energy consumption by only executing computation in correspondence of events (regardless of any structure of the sparsity), in contrast with ordinary DL networks that always compute full activation maps.

The key to integrating sEMG-based control and event-based computing is to devise a strategy for bringing the sEMG to the spiking domain while preserving as much information as possible. For instance, a popular technique is the delta-modulator analog-to-digital converter [21], [22]. Previous solutions for processing the sEMG on event-driven hardware have demonstrated that SNNs can successfully extract informative features consuming $< 1\,\mathrm{nJ}$ per spike, totaling a power draw as small as $0.05\,\mathrm{mW}$ (hundredths of a milliwatt) [23] since the proposed nets have a size of the order of 100 neurons with a bio-plausible maximum individual firing rate of the order of $100\,\mathrm{spike/s}$. However, this vein has only tackled classification so far [23], [24], yielding inspiring insight into the dynamics of activation patterns [25] without tackling regression.

This work proposes a technique for encoding the High-Density sEMG (HD-sEMG) signal into an event format that successfully preserves the information content required for the multi-finger force estimation regression task. Our contribution is three-fold:

- we propose an event-based method that processes the HD-sEMG samples one-by-one in streaming, updating its state and generating spike trains;
- we tune the parameters of our method on the real HD-sEMG regression dataset HYSER, obtaining a Mean Absolute Error (MAE) of $(8.42 \pm 2.80)\%$ of the Maximum Voluntary Contraction (MVC) in a multi-day, multi-finger scenario, on a par with the literature that addresses easier settings;
- we deploy and profile our setup on a parallel ultra-low power Microcontroller Unit (MCU), getting a power consumption $\leq 23.1\,\mathrm{mW}$, an energy draw $\leq 6.37\,\mu\mathrm{J}$ per sample ($2.8\times$ to $11\times$ more energy-efficient than the reference SoA single-core baseline [26], and a latency $\leq 280\,\mu\mathrm{s}$ per sample, shorter than HYSER's HD-sEMG sampling period, thus compatible with real-time processing.

This work is an extension of our previous paper [26] and expands its heuristic findings as outlined in Table I; namely, here we show that our event-based encoding method remains accurate and versatile if ported from kinematics regression based on sparse sEMG to force estimation based on HD-sEMG, always within strict latency and resource limits.

We release the code developed for this work open-source.[1]

II. Materials & Methods

A. sEMG and HYSER Dataset

The EMG originates from the motor unit action potential trains in the muscular fibers, and it is one of the most relevant indicators of the activity of muscles [30], [31]. The EMG signal has an amplitude of $10\,\mu\mathrm{V}$–$1\,\mathrm{mV}$ and a bandwidth up to $2\,\mathrm{kHz}$. Surface EMG is the technique of sensing the EMG from the skin relying on surface electrodes that enable non-invasiveness, which is essential for the acceptance of EMG-based health or consumer HMIs. The disadvantages of sEMG are represented by signal variability across subjects and time [32], motion artifacts, and noise sources such as floating ground, crosstalk, and power line interference [33]. In the field of automated learning on sEMG, SoA countermeasures to the inherent variability are multi-session training [4], [6], [32] and retuning of either non-deep [34], [35] or DL models [36], [37].

In this work, we target the High-densitY Surface Electromyogram Recordings (HYSER)[2] [12], an open-access HD-

[1] https://github.com/pulp-bio/hdsemg-force-regression

[2] https://www.physionet.org/content/hd-semg/1.0.0/

TABLE I

OUTLINE OF THE PRESENT WORK AS AN EXTENSION OF OUR PREVIOUS PUBLICATION [26]. FOR A FAIR COMPARISON, THIS WORK'S PROFILING SHOWN HERE REFERS TO 64 LEAKY INTEGRATE-&-FIRE NEURONS AS [26]; THE COMPLETE PROFILING RESULTS ARE EXPOSED IN III-C.

WORK	Regression target	Approach	sEMG type: dataset	SoA baseline	MCU: processor(s)	Results	
						MAE (average ± std)	profiling
Zanghieri et al. [26] (extended here)	kinematics: joint angles	**event-based encoding**	sparse sEMG: NinaPro DB8	[5]	STM32 F401: ARM Cortex-M4F	(8.84 ± 2.28) degrees	latency: 448 µs energy: 18.2 µJ
This work	dynamics: forces		HD-sEMG: HYSER `RANDOM`	[12] [27]–[29]	GWT GAP9: 8 RISC-V cores	$(8.42 \pm 2.80)\%$ MVC	latency: 69.6 µs energy: 1.55 µJ

sEMG dataset realized for research on hand gesture recognition and force estimation. The dataset was collected from 20 healthy participants, each undergoing two sessions at a distance of 3 to 25 days (8.5 ± 6.7 days on average). The HD-sEMG data were acquired with four 8×8 HD-sEMG arrays (256 channels in total) placed two on each side of the forearm on the extensor and flexor muscles, using an OT Bioelettronica Quattrocento system and sampling at $2048\,\mathrm{samples/s}$. Force signals were acquired during isometric contractions, with a sensor-amplifier pair for each finger, using Huatran SAS sensors and Huatran HSGA amplifiers, sampling at $100\,\mathrm{samples/s}$.

The HYSER dataset is composed of 5 sub-datasets:

1 `PR`: pattern recognition on 34 hand gestures;
2 `MVC`: trials for determining the MVC of every finger's flexion and extension;
3 `1-DoF`: single-finger contractions, for 1-Degree of Freedom (DoF) force estimation;
4 `N-DoF`: multi-finger contraction following prescribed combinations and trajectories, for 5-DoF force estimation in controlled conditions;
5 `RANDOM`: with multi-finger contractions performed in a fashion defined *random task*, i.e. with no prescribed protocol of combinations or trajectories.

Datasets 2 to 5 contain the forces of individual fingers for research on force estimation. In particular, we focus on the `RANDOM` dataset, which consists of 5 trials per subject, each lasting 25 s, performed with a 5 s inter-trial rest to prevent muscle fatigue.

Most literature on HYSER focuses on discrete gesture recognition on the `PR` dataset, whereas few works to date have addressed continuous force estimation on `1-DoF` and `N-DoF`. Moreover, the `RANDOM` dataset is only dealt with in the basic benchmarking of the first HYSER paper [12]. Table II reports the SoA works on the HYSER regression datasets. To the best of our knowledge, we are (*i*) the first to tackle HYSER's `RANDOM` dataset in a multi-day setting; (*ii*) the first to deploy and profile our regression algorithm for the HYSER task on a hardware platform suitable for low-power, low-latency wearable HMIs. Thus, we address the working conditions closest to reality, where the force ranges and trajectories are not predefined and can differ from training to test. Moreover, we propose a regressor that is explicitly designed to be hardware-friendly, considering the power, energy, and latency constraints of wearable real-time HMIs.

B. Event-based Encoding

We encode the raw sEMG to events with a power-based approach inspired by how the mammalian cochlea transduces various frequencies into neural spike trains. The principle of using a bank of filters combined with neural integration has been a model for many designers of bio-inspired hardware to implement circuits that imitate the event encoding happening in nature, either in the digital [38] or analog domain [39], [40].

We execute the conversion to events by implementing a set of Leaky Integrate-and-Fire (LIF) neurons, each associated with one of the processed sEMG signals. The LIF is a very parsimonious model of the biological neuron, characterized by an inner membrane potential $V_{\mathrm{mem}}(t)$ that follows the electrical law

$$\frac{\mathrm{d}V_{\mathrm{mem}}}{\mathrm{d}t} = -\frac{(V_{\mathrm{mem}} - \mathcal{E}_{\mathrm{leak}}) - \frac{I_{\mathrm{inj}}(t)}{g_{\mathrm{leak}}}}{\tau} \tag{1}$$

where τ is the membrane relaxation time, $\mathcal{E}_{\mathrm{leak}}$ is the constant leak reversal potential, I_{inj} is the injected current, and g_{leak} is the constant leak conductance. When V_{mem} surpasses a fixed threshold level V_{thr}, the LIF creates a spike, which acts as an emitted event associated with the time of crossing t_{spike}. Then, the LIF is subject to a refractory time t_{refr}, defined as a segment of time $[t_{\mathrm{spike}}, t_{\mathrm{spike}} + t_{\mathrm{refr}}]$ where the LIF is forced to a reset value V_{reset}:

$$V_{\mathrm{mem}}(t) \equiv V_{\mathrm{reset}} \qquad t \in [t_{\mathrm{spike}}, t_{\mathrm{spike}} + t_{\mathrm{refr}}]\,, \tag{2}$$

also pausing the response to the inhomogeneous driving term $I_{\mathrm{inj}}(t)/g_{\mathrm{leak}}$.

Our numerical LIF emulation for accuracy-oriented regressions does not need to account for the electrical nature of the bio-inspired model. This makes it convenient to change variables to remove the electrical quantities and simplify the notation:

$$x(t) \triangleq \frac{V_{\mathrm{mem}}(t) - \mathcal{E}_{\mathrm{leak}}}{V_{\mathrm{thr}} - \mathcal{E}_{\mathrm{leak}}} \tag{3}$$

$$x_{\mathrm{drive}}(t) \triangleq \frac{1}{V_{\mathrm{thr}} - \mathcal{E}_{\mathrm{leak}}} \cdot \frac{I_{\mathrm{inj}}(t)}{g_{\mathrm{leak}}}. \tag{4}$$

The physical sense of this transformation is to refer the membrane voltage to the constant $\mathcal{E}_{\mathrm{leak}}$, and measure the membrane voltage and $I_{\mathrm{inj}}(t)/g_{\mathrm{leak}}$ (dimensionally a tension) as a fraction of of $V_{\mathrm{thr}} - \mathcal{E}_{\mathrm{leak}}$, which is the dynamic range of

TABLE II
OVERVIEW OF THE LITERATURE OF FORCE REGRESSION ON THE HYSER DATASETS.

WORK	Interest	Approach	HYSER sub-dataset	Data split	Numerical results (average ± std)	HW?
Jiang et al. [12] (2021)	dataset presentation	FIR kernel	**RANDOM**	within-day, leave-1-trial-out	RMSE $= (8.57 \pm 5.27)\%$ MVC	✗
Jiang et al. [27] (2022)	channel selection	FIR kernel + random masks	N-DoF	**cross-day**, leave-1-subject-out	RMSE $= (8.66 \pm 0.96)\%$ MVC	✗
Jiang et al. [28] (2023)	robustness vs. noise, physiological explainability	deep forests	1-DoF	**cross-day: train on day 1, test on day 2**	RMSE $= (8.0 \pm 2.3)\%$ MVC $r_{\text{Pearson}} = 0.900 \pm 0.101$ $R^2 = 0.631 \pm 0.172$	✗
Wu et al. [29] (2023)	extraction of motor units	gCKC BSS + cumulative spike train + linear regression	1-DoF	no ML-style validation	$r_{\text{Pearson}} = 0.908 \pm$ n.a.	✗
This work	event-based ML embedded on parallel ultra-low power MCU	encoding as events + linear regression	**RANDOM**	**cross-day: train on day 1, test on day 2**	MAE $= (8.42 \pm 2.80)\%$ MVC	✓

the system. This yields a dimensionless state $x(t)$ with firing threshold $x_{\text{thr}} = 1$ and law

$$\frac{\mathrm{d}x}{\mathrm{d}t} = -\frac{x - x_{\text{drive}}}{\tau}. \tag{5}$$

This form of the law makes it more evident that the injected current plays the role of an external driving term. We encode the sEMG as events using each channel to drive an independent LIF unit. In our experiments, we set the relaxation time to $\tau = 10\,\text{ms}$ and the refractory time to $t_{\text{refr}} = 2\,\text{ms}$. To create the driving term from the raw sEMG data, we multiply the dataset values $\text{sEMG}(t)$ by an empirical gain g_{data}, converting the arbitrary-units into an $x_{\text{drive}}(t)$:

$$x_{\text{drive}}(t) = g_{\text{data}} \cdot |\text{sEMG}(t)|\,, \tag{6}$$

experimenting different values of g_{data}. Since the signals of the HYSER dataset are available as voltage values, g_{data} has dimensions V^{-1}.

After each LIF, we simulate a post-synaptic potential $x_{\text{post}}(t)$ driven by the spikes generated by the corresponding source LIF. This potential also undergoes relaxation, with a relaxation time τ_{post} that causes decay toward 0 if no spikes are received. More formally, x_{post} obeys the law

$$\frac{\mathrm{d}x_{\text{post}}}{\mathrm{d}t} = -\frac{x_{\text{post}}}{\tau_{\text{post}}} + \sum_{t_{\text{spike}}} \delta(t - t_{\text{spike}}) \tag{7}$$

where δ denotes the Dirac delta and represents the fact that x_{post} is raised by $+1$ (dimensionless) increments at each received spike. The relaxation regulated by τ_{post} has the effect of a causal exponential decay kernel [41], a form of rate encoding. This rate encoding is equivalent to an event count that, at the present time t, weights every spike in the past as $\exp(-(t - t_{\text{spike}})/\tau_{\text{post}}) \leq 1$. This causal exponential-kernel rate always takes values in the range

$$0 \leq x_{\text{post}} < \frac{1}{1 - e^{-t_{\text{refr}}/\tau_{\text{post}}}} \tag{8}$$

In our experiments, we set $\tau_{\text{post}} = 250\,\text{ms}$, thus getting x_{post} values in the range $[0, 125.5)$. Finally, we use all LIFs' x_{post} values as the input regression features for force estimation.

Numerically, the simulations of the LIF neurons can be implemented as discrete updates of x and x_{post}:

$$x \leftarrow x \cdot e^{-\frac{\Delta t}{\tau}} + x_{\text{drive}} \cdot (1 - e^{-\frac{\Delta t}{\tau}}) \tag{9}$$

$$x_{\text{post}} \leftarrow x_{\text{post}} \cdot e^{-\frac{\Delta t}{\tau_{\text{post}}}} + 1\,[\text{if spike}] \tag{10}$$

where Δt is the discrete time step of the simulation; Eq. 9 is skipped during refractories. A key feature of (9) and (10) is that they update *online*, i.e., they consume one single sEMG input for each channel at a time, computing x_{drive} and then the new x_{pre} and x_{post}; at the next sampling period, the new sEMG input data overwrite the old ones. Hence, the size of the input data stored at each sampling period never exceeds $N_{\text{ch}} \times 4\,\text{bytes} = 256 \times 4\,\text{bytes} = 1\,\text{KiB}$ for `float32` data on the $N_{\text{ch}} = 256$-channel HYSER dataset. In addition, we apply L_1-regularization for a data-driven channel selection, further reducing the size of inputs and computation, as explained in II-C.

We developed two implementations of the event-based encoding and the inference:

- in Python (v. 3.8), we directly configured the `NeuronGroup` and `Synapses` classes native to the simulator Brian2[3] [42] v. 2.5;
- in C, for deployment on the MCU, we implemented (9) and (10) and refractories, parallelizing as detailed in Section II-D.

We used the Python and C implementations to compute the regression error statistics in offline experiments on a PC and online experiments on the GAP9 MCU (II-D), respectively. In the Python offline experiments, the whole output time series of each HYSER `RANDOM`'s 25-second recording is available for taking the error statistics. In the online experiments, a PC sends the HYSER's samples to the MCU via a serial interface in streaming; the MCU consumes them to update the neurons' states and the inference; finally, the MCU sends the inference output back to the PC. Figure 1 shows the setup used for the online experiments. The GAP9 MCU repeats the reception-processing-transmission loop online for each sample of every

[3]https://github.com/brian-team/brian2

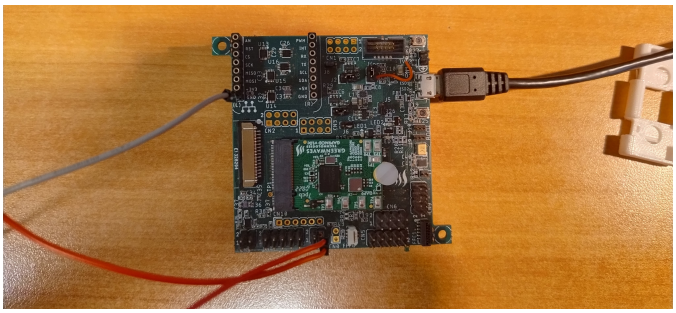


Fig. 1. Setup of the online experiments: board mounting the GAP9 MCU with serial connection to a PC. (For pictures of the placing of HD-sEMG arrays on the forearm, please refer to the original HYSER paper [12], to which the sensors setup pictures belong.)

HYSER `RANDOM`'s 25-second recording. On the PC side, a script reads the regression outputs and saves them into an array. This series is used to check the numerical match of the results of the Python and the C implementations and to compute the regression error statistics.

C. Regression

We tackle force estimation with linear regression to ensure reduced memory and computation requirements, prioritizing the low embedded resource budget. With the $N_{\text{ch}} = 256$ HD-sEMG channels and the $N_{\text{DoFs}} = 5$ of the HYSER dataset, the force estimation task is framed as a

$$\mathbf{x} \in \mathbb{R}^{N_{\text{ch}}} \longmapsto \mathbf{y} \in \mathbb{R}^{N_{\text{DoFs}}} \tag{11}$$

multivariate, multi-target regression, parameterized by a $\mathbf{W} \in \mathbb{R}^{N_{\text{ch}} \times N_{\text{DoF}}} = \mathbb{R}^{5 \times 256}$ coefficients matrix and an intercept $\overline{\mathbf{y}}_{\text{train}} \in \mathbb{R}^{N_{\text{DoFs}}} = \mathbb{R}^5$ equal to the sample mean of the training set values. In `float32` format, each inference amounts to a memory footprint of 6184 bytes (including input and output) and 1285 FLOP.

To keep the processing hardware-friendly, we reduce the resource budget by applying a strong L_1 regularization. The target function of the regression is thus

$$\frac{1}{N_{\text{train}}} \sum_{i}^{N_{\text{train}}} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|_2^2 + \alpha \|\mathbf{W}\|_1 \tag{12}$$

where $\mathbf{y}_i, \hat{\mathbf{y}}_i = \mathbf{W}\mathbf{x} \in \mathbb{R}^5$ are the multivariate ground truth and estimation, respectively, corresponding to the i-th inference, N_{train} is the total number of inferences (equivalent to the training set size, i.e., each HYSER `RANDOM` Day 1 session), $\|\cdot\|_2$ is the Euclidean norm, $\|\cdot\|_1$ is the L_1-norm, and α is the parameter governing the amount of regularization (notice that it does not get divided by N_{train}). We tune the amount of regularization by exploring different values for α.

In addition to countering overfitting, the prerogative of the L_1 regularization is to perform automatic sEMG channel selection. The reason why L_1 regularization results in automatic feature selection is that it induces sparsity, since reducing any coefficient benefits the penalty term equally, regardless of the coefficient's magnitude; in contrast, $L_{p>1}$-norms privilege reducing the larger coefficients (since the $p > 1$ exponent makes their contribution to the norm larger), thus making $L_{p>1}$-norms less likely to push coefficients to 0.

The L_1-induced data-driven feature selection on the sEMG channels results in fewer associated LIF units compared to HYSER's $N_{\text{ch}} = 256$ total sensors. Channel selection reduces the application's requirements, namely input data bandwidth, memory footprint, and computational load. This reduction makes our processing more hardware-friendly for resource-constrained computation devices such as the MCU targeted in this work (II-D). Moreover, this data-driven reduction experimentally determines the number of channels actually required for an accurate regression. So, in this work, L_1-regularization is used as the key for studying the integration of HD-sEMG acquisition setup and embedded platforms.

As a dataset split on HYSER `RANDOM`, we used the Day 1 session for training and the Day 2 session for validation; for both sessions, we used all the 5 trials. We ran training and test separately for every subject without any multi-subject training or inter-subject validation. Together with the choice of HYSER `RANDOM` itself, this dataset split is the most challenging and closest to a real test scenario; no previous work on HYSER has addressed multi-day inference of 5-finger forces (Table II).

We determine the regression accuracy using the Mean Absolute Error (MAE):

$$\text{MAE} = \frac{1}{N_{\text{infer}} N_{\text{DoF}}} \sum_{i=1}^{N_{\text{infer}}} \|\hat{\mathbf{y}}_i - \mathbf{y}_i\|_1 \tag{13}$$

where N_{infer} is the total number of inferences, i.e., the validation set size, and the rest follows the notation of Section II. We measure the error as a fraction of the MVC, as is the standard approach [12], [27]–[29]. To rescale forces to the MVC scale, we determined the MVC for each direction (i.e., flexion or extension) of each finger of each subject using the data from HYSER `MVC` following the same heuristic as suggested by the dataset authors, namely determining the MVC as the average of the 200 strongest values.[4] Assessing the control quality via the MAE is convenient because the MAE is first-order, thus more robust to outliers than quadratic statistics such as the (R)MSE or the multivariate coefficient of determination R^2. We average the MAE over time (i.e., over the 5 trials of each session) and over all 5 finger DoFs, as expressed by (13), and over all 20 participants.

D. Deployment and Profiling on a Parallel ULP MCU

We have deployed the numerical model of the LIF neurons associated with each HD-sEMG onto the commercial MCU GAP9[5] (Fig. 2), which features a Parallel Ultra-Low Power (PULP)[6] 9-core cluster accelerator based on the RISC-V Instruction Set Architecture extended with specialized DSP and ML instructions. This device is a SoA low-power processor that ranked first in latency and energy consumption on the benchmarks MLPerf Tiny v1.0.[7] In a potential complete prototype implementing our HD-sEMG-based control policy,

[4] https://www.physionet.org/content/hd-semg/1.0.0/toolbox/function
[5] https://greenwaves-technologies.com/gap9_processor/
[6] https://pulp-platform.org/
[7] https://mlcommons.org/en/inference-tiny-10/

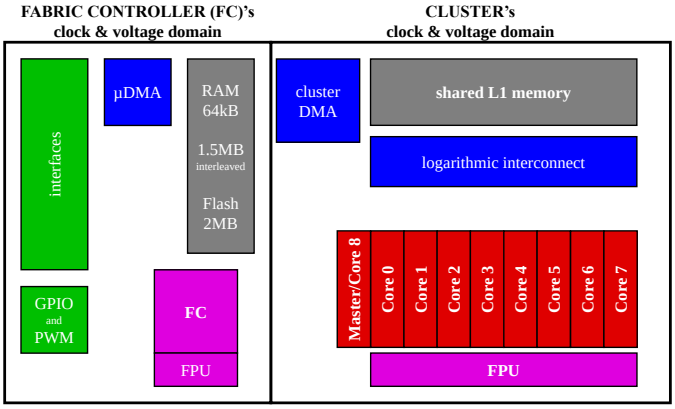


Fig. 2. Essential scheme of the GAP9 MCU and its parallel multicore cluster.

the MCU's role is computation, i.e., the data processing that consists of updating the LIF neurons' states and executing the regression inference. In contrast, the upstream functions of data sampling and transmission are to be performed by other ADC and interface modules that produce the data and convey them to the MCU.

We have programmed GAP9 implementing the LIF update steps in C with parallelization. Parallelization comes naturally since all LIFs are independent as to both state variables and operations, so LIFs were distributed evenly across the cores, with a workload difference of 1 LIF at maximum in the experiments where LIFs are not a multiple of the used cores. A core getting assigned a LIF means that the core will execute all that LIF's x and x_{post} discrete update iteration steps. We parallelize over up to 8 cores since GAP9's cluster's ninth core, referred to as Master Core or Core 8, only serves as a cluster controller and manages Direct Memory Access (DMA) memory transfers.

For profiling, we used the settings corresponding to GAP9's highest energy-efficiency, namely $V_{\text{dd core}} = 0.65\,\text{V}$ and $f_{\text{CLK}} = 240\,\text{MHz}$. We measured latency in cycles by exploiting the performance counter exposed by the API of PMSIS,[8] the open-source system layer for GAP9's operating system. We determined latency in physical time as num_cycles$/f_{\text{CLK}}$. We measured the power draw experimentally, using the GAP9's Evaluation Kit[9,10] and a Nordic Semiconductor Power Profiler Kit II (PPK2).[11] The PPK2 measured the current consumption of GAP9's core, excluding the peripherals and the off-chip memories. We used a GPIO to synchronize the current measurement with the code execution. Finally, we determined the energy consumption as power$\times$latency.

III. Experimental Results

A. Time-Domain Behavior

Fig. 3 displays a representative example of the time-domain behavior of the finger force estimation provided by our algorithm. The reported results are from HYSER's Subject 1, RANDOM dataset, Day 2 (i.e., the one never seen in training), all 5 trials, all 5 fingers. These data are chosen for display as they are representative of the general time-domain trends observed in the results.

The reported trials contain many examples of good regression quality, especially for the thumb, middle, and ring fingers. The rest position is generally well-modeled, with a good match between the ground truth and the estimation when force is in the interval $\pm 0.05\,\text{MVC}$. The timing of the falling and rising fronts, corresponding to the dynamic phases of flexions and extensions, respectively, are also accurate. In contrast, the estimation errors mainly happen in the central regions of flexions and extensions, where the estimation often stops before the force reaches its full amplitude; this happens both in some steady central segments and in some triangular peaks. Interestingly, this kind of time-domain behavior, with accurate timing in transients and a central offset error, is the same as observed in the estimation of the hand kinematics [5].

The displayed results also contain the typical estimation errors obtained with our algorithm, especially in the index finger trials. A common regressor's mistake is failing to recognize negative forces, i.e., finger flexions. This error is one of the most frequent erratic behaviors in the results. In future work, the trials with poor modeling of contractions can be addressed by adding a flexion-vs-extension(-vs-rest) detector before the regressor, using the regressor only for estimating the amount of force and the dedicated detector for recognizing the force's sign. The little finger's results are the ones showing the least accurate regression. The reason why this poor modeling does not harm the overall performance of our method is that the little finger has an MVC, and thus a force dynamic range, that is on average $(0.55 \pm 0.07)\times$ compared to the other single fingers. This means that the little finger contributes to the global dynamics of the hand forces by approximately one-half compared to the other single fingers, making errors less impactful from an end-to-end application viewpoint.

It is worth remarking that the regression issues illustrated in this section are only discussed to present an overview of the most typical errors from a time-domain point of view. These behaviors do not compromise the average regression quality. The overall competitiveness of our method compared to the SoA is shown by the regression error statistics presented in the next section.

B. Regression Error

Fig. 4 shows the regression error obtained evaluating a grid of pairs of g_{data} and α, exploring

$$g_{\text{data}}\,[\text{V}^{-1}] = 5.0,\ 10.0,\ 15.0,\ 20.0,\ 25.0,\ 30.0 \quad (14)$$

$$\alpha = 10^{-1},\ 10^{-1.5},\ 10^{-2}. \quad (15)$$

The fits with the mildest regularization $\alpha = 0.01$ do not yield reliable results: the high variability of the MAE means a high regression error for a relevant fraction of fingers

[8] https://greenwaves-technologies.com/manuals/BUILD/HOME/html/index.html

[9] https://greenwaves-technologies.com/product/gap9_evk-gap9-evaluation-kit-efused/

[10] https://greenwaves-technologies.com/product/gap9-resources/

[11] https://www.nordicsemi.com/Products/Development-hardware/Power-Profiler-Kit-2

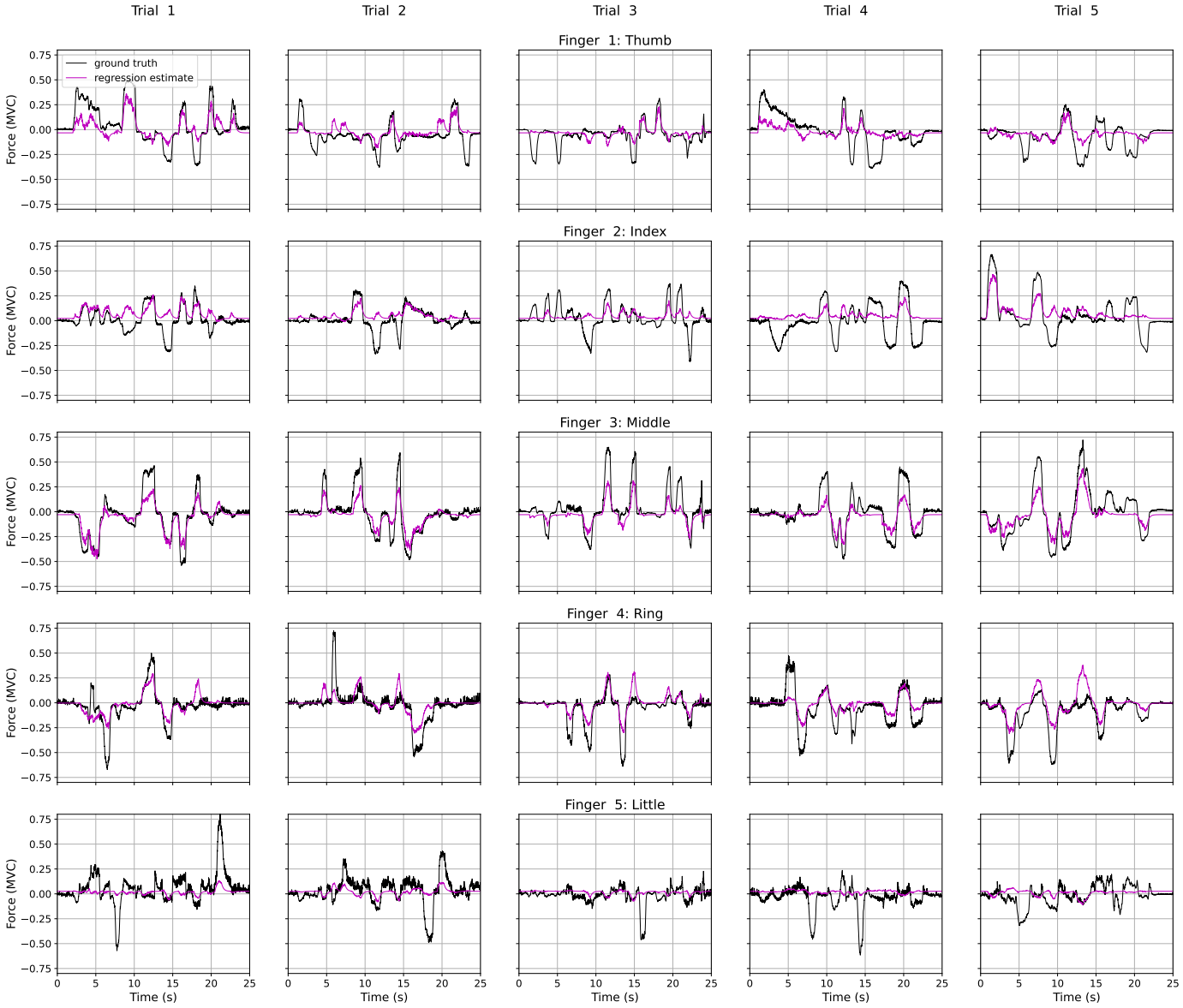


Fig. 3. Force estimation results obtained for regression inference on HYSER's Subject 1, `RANDOM` dataset, Day 2 (the one not seen in training), all 5 trials, all 5 fingers.

or subjects; $\alpha = 0.01$ proves thus poor methodologically. Pushing regularization to $\alpha = 0.032$ makes results more stable, improving both the MAE's average and variance for all values g_{data}. Tightening to $\alpha = 0.1$ further improves both averages and variances for all the explored g_{data} yielding the lowest error, i.e. $(8.42 \pm 2.80)\%$ MVC for $g_{\text{data}} = 15.0\,\text{V}^{-1}$. These results prove that our method can work in a multi-day multi-finger setup in the absence of fixed force exercises, achieving results in the same range as previous works that tackled the HYSER dataset in easier settings, namely within-day [12], with predefined force protocol [27], or single-finger [28] (as summarized in Table II), whereas our validation is closer to actual non-laboratory scenarios.

In the perspective of pursuing real-world implementations, error variability is methodologically as essential as error average to ensure that a method is capable of uniform performance across different users. The standard deviation is due to the inherent variability across subjects and sessions, which produce sEMG data with easier or harder patterns. Tightening the screw of regularization has decreased both the average error and its dispersion, proving more beneficial than adjusting g_{data}, which only yielded plateaus with uniform variance.

As to the feature sparsity obtained from the L_1 regularization, the identified optimal solution $\alpha = 0.1$, $g_{\text{data}} = 15.0\,\text{V}^{-1}$ has a minimum of 42 sEMG channels (out of $N_{\text{ch}} = 256$) with a non-zero coefficient for at least one of the 5 fingers (i.e., maximum sparsity of 83.6%, for subject 6), and a maximum of 84 (i.e., minimum sparsity 67.2%, for subject 13), with a median of 55 (sparsity 78.5%, for subjects 10 and 18). These results are not only competitive but also interesting for the insight they provide in the perspective of integrating HD-sEMG with embedded systems. On the one hand, the heuristic range 42 – 84 is above the typical channel count of a low-density, sparse sEMG setup, confirming that the use of HD-

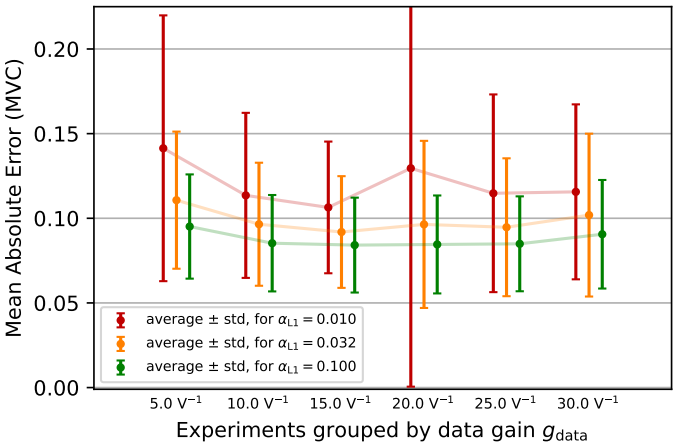


Fig. 4. Regression error results. Explored data gains shown as categorical to allow horizontal x-displacement to de-overlap bars.

sEMG is motivated. On the other hand, the resulting 42 – 84 channels are much fewer than HYSER's total $N_{\text{ch}} = 256$ (i.e., sparsity is high), informing in a data-driven way that an accuracy-oriented application does not require channel counts of the order of 100 or even 200. So, a key insight of the regression results is the empirical estimate of the useful channel count actually needed for a low regression error.

The amount of L_1-induced data-driven sparsity also shapes the conclusions of the on-device profiling results (in the next section) since we only need to implement the LIFs associated with the selected input channels. For instance, a consequence of sparsity is on input data memory footprint: sparsity lowers the size of input data from the theoretical maximum of 1 MiB (as explained in II-B) to a minimum of 168 bytes, a maximum of 336 bytes, and a median of 220 bytes across subjects, corresponding to the values of 42, 84, and 55 channels reported above. These results prove that L_1-regularization contributes to making our method hardware-friendly in the presence of a high number of input channels, such as the 256 sensors of the HYSER dataset.

C. Profiling

Figs. 5 and 6 show the profiling results regarding speedup and latency on 8 cores, respectively.

The experimental speedup on 8 cores (Fig. 5) is close to 8× and is better for a higher number of simulated LIF neurons. The speedup on 8 cores for 64 and 256 LIFs is 7.45× and 7.81× on a theoretical maximum of 8×. Since all LIFs are independent, our algorithm is fully parallelizable mathematically. The only sequential part when executing on HW is the initial DMA transfer of one `float32` value per channel from the MCU's L2 memory to the cluster's L1 memory for faster access during the subsequent computation. This DMA transfer takes < 2 cycles per `float32` datum. According to Amdahl's law, this overhead yields an ideal speedup of 7.95× on 8 cores, equal for all the profiled workloads since both the transfer and the computation are proportional to the number of LIFs. Considering the Amdahl upper bound, the obtained speedup is 97.6% and 98.3% of the Amdahl ideal for 64 and 256 LIFs, respectively.

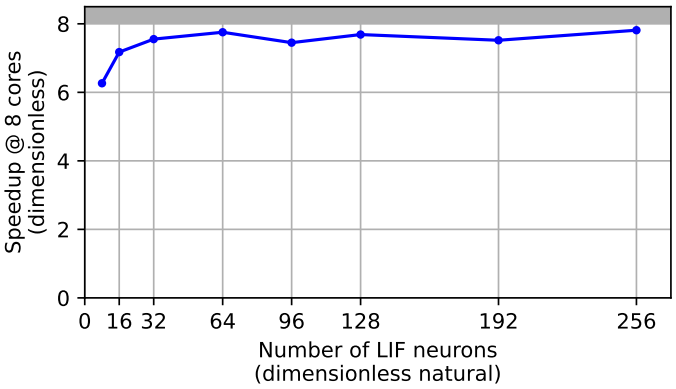


Fig. 5. Speedup on 8 cores obtained for different numbers of executed LIF neurons. The grey region is the unreachable speedup $\geq 8\times$.

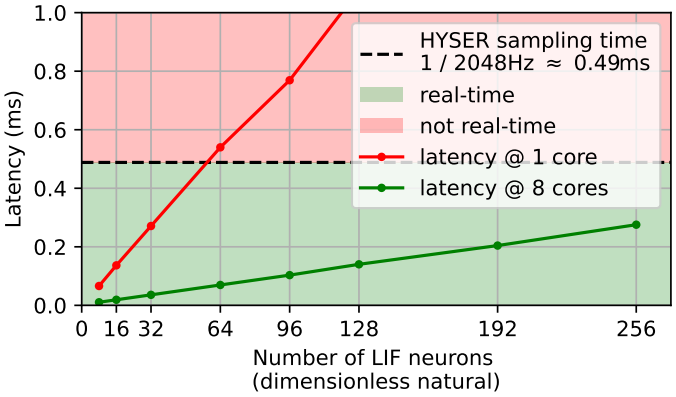


Fig. 6. Latency as a function of the number of executed LIF neurons, using 8 cores.

The latency results (Fig. 6) show that our implementation satisfies the real-time constraint since it can update all LIFs within the sampling period

$$T_{\text{sEMG HYSER}} = \frac{1}{2048\,\text{Hz}} \approx 488\,\mu\text{s}, \tag{16}$$

thus proving able to use every sEMG sample as a driving term for the corresponding LIF. The highest workload case, i.e., 256 LIFs, has a latency of 276 us. We observe that workloads between 64 and 96 LIFs have a latency of 69.6 µs to 103 µs, which is in the range of 1/7 to 1/5 of the available time. This range of workloads is similar to the range of lower-sparsity subjects identified in the regression results of the previous section, namely 55 (median) to 84 (maximum) LIFs. These results confirm that the amount of parallelism we have pursued is required by our use case, motivating our choice of the parallel platform. Overall, our solution is consistent with the HYSER dataset and with SoA sEMG applications, characterized by sampling frequencies typically > 1 kHz.

The measured power consumption is 22.3 mW and 23.1 mW for 64 and 256 LIF neurons respectively, with a respective energy consumption of 1.55 µJ and 6.35 µJ. In particular, the energy draw for 64 LIFs is 11.7× lower compared to the 18.2 µJ of our previous work with 64 LIFs deployed on a single-core MCU (STM32 F401, featuring ARM Cortex-M4F) [26], denoting the improved energy-efficiency of our parallel implementation.

IV. Conclusion

We have presented a solution for estimating finger forces from the HD-sEMG exploiting an event-based signal encoding, expanding existing event-based heuristics for hand kinematics. We are the first to address the HYSER HD-sEMG dataset in a multi-day validation without fixed force sequences and the first to deploy and profile a solution for HYSER's most arduous task onto a parallel ultra-low-power MCU. Our regression error is in the same range as previous works on easier scenarios such as within-day, single-finger, and fixed force patterns. Our method proves accurate and versatile even within embedded devices' power and latency budget, proving competitive in the conditions closest to actual non-laboratory settings. In future work, we will exploit this work's insight about the optimal HD-sEMG channel number to implement an acquisition setup and realize a novel dataset for research; moreover, on the performance front, we will port our event-based processing onto neuromorphic event-driven hardware to exploit the increased energy efficiency of event-proportional computing.

Acknowledgment

We thank Prof. Chenyun Dai for providing valuable clarifications about the HYSER dataset. We thank GreenWaves Technologies for the preview access to the SDK of GAP9 and Luca Valente for the counseling on the device's behavior.

References

[1] L. Guo *et al.*, "Human-machine interaction sensing technology based on hand gesture recognition: A review," *IEEE Transactions on Human-Machine Systems*, vol. 51, no. 4, pp. 300–309, 2021. DOI: 10.1109/THMS.2021.3086003.

[2] C. Castellini and P. van der Smagt, "Surface EMG in advanced hand prosthetics," *Biological Cybernetics*, vol. 100, no. 1, pp. 35–47, Nov. 2008. DOI: 10.1007/s00422-008-0278-1.

[3] A. Phinyomark and E. Scheme, "EMG pattern recognition in the era of big data and deep learning," *Big Data and Cognitive Computing*, vol. 2, no. 3, 2018. DOI: 10.3390/bdcc2030021.

[4] M. Zanghieri *et al.*, "Temporal variability analysis in sEMG hand grasp recognition using temporal convolutional networks," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2020, pp. 228–232. DOI: 10.1109/AICAS48895.2020.9073888.

[5] M. Zanghieri *et al.*, "sEMG-based regression of hand kinematics with temporal convolutional networks on a low-power edge microcontroller," in *2021 IEEE International Conference on Omni-Layer Intelligent Systems (COINS)*, 2021, pp. 1–6. DOI: 10.1109/COINS51742.2021.9524188.

[6] M. Zanghieri *et al.*, "Robust real-time embedded EMG recognition framework using temporal convolutional networks on a multicore IoT processor," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 14, no. 2, pp. 244–256, 2020. DOI: 10.1109/TBCAS.2019.2959160.

[7] L. Nieuwoudt and C. Fisher, "Investigation of real-time control of finger movements utilizing surface EMG signals," *IEEE Sensors Journal*, vol. 23, no. 18, pp. 21 989–21 997, 2023. DOI: 10.1109/JSEN.2023.3299384.

[8] L. Tong *et al.*, "sEMG-based gesture recognition method for coal mine inspection manipulator using multistream CNN," *IEEE Sensors Journal*, vol. 23, no. 10, pp. 11 082–11 090, 2023. DOI: 10.1109/JSEN.2023.3264646.

[9] O. Kerdjidj *et al.*, "Implementing hand gesture recognition using EMG on the Zynq circuit," *IEEE Sensors Journal*, vol. 23, no. 9, pp. 10 054–10 061, 2023. DOI: 10.1109/JSEN.2023.3259150.

[10] B. Leelakittisin *et al.*, "Enhanced lightweight CNN using joint classification with averaging probability for sEMG-based subject-independent hand gesture recognition," *IEEE Sensors Journal*, vol. 23, no. 17, pp. 20 348–20 356, 2023. DOI: 10.1109/JSEN.2023.3296649.

[11] A. Krasoulis *et al.*, "Effect of user practice on prosthetic finger control with an intuitive myoelectric decoder," *Frontiers in Neuroscience*, vol. 13, Sep. 2019. DOI: 10.3389/fnins.2019.00891.

[12] X. Jiang *et al.*, "Open access dataset, toolbox and benchmark processing results of high-density surface electromyogram recordings," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 29, pp. 1035–1046, 2021. DOI: 10.1109/TNSRE.2021.3082551.

[13] P. Koch *et al.*, "Regression of hand movements from sEMG data with recurrent neural networks," in *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2020, pp. 3783–3787. DOI: 10.1109/EMBC44109.2020.9176278.

[14] A. Krasoulis and K. Nazarpour, "Myoelectric digit action decoding with multi-output, multi-class classification: An offline analysis," *Scientific Reports*, vol. 10, no. 1, Oct. 2020. DOI: 10.1038/s41598-020-72574-7.

[15] R. C. Sîmpetru *et al.*, "Accurate continuous prediction of 14 degrees of freedom of the hand from myoelectrical signals through convolutive deep learning," in *2022 44th Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*, 2022, pp. 702–706. DOI: 10.1109/EMBC48229.2022.9870937.

[16] R. C. Sîmpetru *et al.*, "Sensing the full dynamics of the human hand with a neural interface and deep learning," *bioRxiv*, 2022. DOI: 10.1101/2022.07.29.502064.

[17] A. Di Mauro *et al.*, "SNE: An energy-proportional digital accelerator for sparse event-based convolutions," in *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2022, pp. 825–830. DOI: 10.23919/DATE54114.2022.9774552.

[18] A. Di Mauro *et al.*, "Kraken: A direct event/frame-based multi-sensor fusion SoC for ultra-efficient visual processing in nano-UAVs," in *2022 IEEE Hot Chips 34 Symposium (HCS)*, 2022, pp. 1–19. DOI: 10.1109/HCS55958.2022.9895621.

[19] S. Moradi *et al.*, "A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (DYNAPs)," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 12, no. 1, pp. 106–122, 2018. DOI: 10.1109/TBCAS.2017.2759700.

[20] W. Maass, "Networks of spiking neurons: The third generation of neural network models," *Neural Networks*, vol. 10, no. 9, pp. 1659–1671, 1997. DOI: 10.1016/S0893-6080(97)00011-7.

[21] F. Corradi and G. Indiveri, "A neuromorphic event-based neural recording system for smart brain-machine-interfaces," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 9, no. 5, pp. 699–709, 2015. DOI: 10.1109/TBCAS.2015.2479256.

[22] M. Sharifshazileh *et al.*, "An electronic neuromorphic system for real-time detection of high frequency oscillations (HFO) in intracranial EEG," *Nature communications*, vol. 12, no. 1, p. 3095, 2021. DOI: 10.1038/s41467-021-23342-2.

[23] E. Donati *et al.*, "Discrimination of EMG signals using a neuromorphic implementation of a spiking neural network," *IEEE Transactions on Biomedical Circuits and Systems*, vol. 13, no. 5, pp. 795–803, 2019. DOI: 10.1109/TBCAS.2019.2925454.

[24] Y. Ma *et al.*, "Neuromorphic implementation of a recurrent neural network for EMG classification," in *2020 2nd IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS)*, 2020, pp. 69–73. DOI: 10.1109/AICAS48895.2020.9073810.

[25] E. Donati *et al.*, "Processing EMG signals using reservoir computing on an event-based neuromorphic system," in *2018 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, IEEE, Oct. 2018. DOI: 10.1109/biocas.2018.8584674.

[26] M. Zanghieri *et al.*, "Event-based low-power and low-latency regression method for hand kinematics from surface EMG," in *2023 9th International Workshop on Advances in Sensors and Interfaces (IWASI)*, 2023, pp. 293–298. DOI: 10.1109/IWASI58316.2023.10164372.

[27] X. Jiang *et al.*, "Random channel masks for regularization of least squares-based finger EMG-force modeling to improve cross-day performance," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 30, pp. 2157–2167, 2022. DOI: 10.1109/TNSRE.2022.3194246.

[28] X. Jiang *et al.*, "Explainable and robust deep forests for EMG-force modeling," *IEEE Journal of Biomedical and Health Informatics*,

vol. 27, no. 6, pp. 2841–2852, 2023. DOI: 10.1109/JBHI.2023.3262316.

[29] W. Wu *et al.*, "A new EMG decomposition framework for upper limb prosthetic systems," *Journal of Bionic Engineering*, Jul. 2023. DOI: 10.1007/s42235-023-00407-0.

[30] C. J. D. Luca, "The use of surface electromyography in biomechanics," *Journal of Applied Biomechanics*, vol. 13, no. 2, pp. 135–163, May 1997. DOI: 10.1123/jab.13.2.135.

[31] R. M. Rangayyan, Ed., *Biomedical Signal Analysis*. Wiley, Apr. 2015. DOI: 10.1002/9781119068129.

[32] B. Milosevic *et al.*, "Exploring arm posture and temporal variability in myoelectric hand gesture recognition," in *2018 7th IEEE International Conference on Biomedical Robotics and Biomechatronics (Biorob)*, 2018, pp. 1032–1037. DOI: 10.1109/BIOROB.2018.8487838.

[33] M. Tomasini *et al.*, "Power line interference removal for high-quality continuous biosignal monitoring with low-power wearable devices," *IEEE Sensors Journal*, vol. 16, no. 10, pp. 3887–3895, 2016. DOI: 10.1109/JSEN.2016.2536363.

[34] E. Donati *et al.*, "Long-term stable electromyography classification using canonical correlation analysis," in *2023 11th International IEEE/EMBS Conference on Neural Engineering (NER)*, 2023, pp. 1–4. DOI: 10.1109/NER52421.2023.10123768.

[35] M. Zanghieri *et al.*, "Online unsupervised arm posture adaptation for sEMG-based gesture recognition on a parallel ultra-low-power microcontroller," in *2023 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, 2023, pp. 1–5. DOI: 10.1109/BioCAS58349.2023.10388902.

[36] U. Côté-Allard *et al.*, "Deep learning for electromyographic hand gesture signal classification using transfer learning," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 27, no. 4, pp. 760–771, 2019. DOI: 10.1109/TNSRE.2019.2896269.

[37] A. Burrello *et al.*, "Tackling time-variability in sEMG-based gesture recognition with on-device incremental learning and temporal convolutional networks," in *2021 IEEE Sensors Applications Symposium (SAS)*, 2021, pp. 1–6. DOI: 10.1109/SAS51076.2021.9530007.

[38] A. Jiménez-Fernández *et al.*, "A binaural neuromorphic auditory sensor for FPGA: A spike signal processing approach," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 4, pp. 804–818, 2017. DOI: 10.1109/TNNLS.2016.2583223.

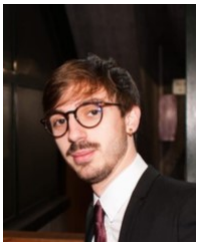
[39] S.-C. Liu *et al.*, "Event-based 64-channel binaural silicon cochlea with Q enhancement mechanisms," in *2010 IEEE International Symposium on Circuits and Systems (ISCAS)*, 2010, pp. 2027–2030. DOI: 10.1109/ISCAS.2010.5537164.

[40] M. Yang *et al.*, "A 0.5V 55µW 64×2-channel binaural silicon cochlea for event-driven stereo-audio sensing," in *2016 IEEE International Solid-State Circuits Conference (ISSCC)*, 2016, pp. 388–389. DOI: 10.1109/ISSCC.2016.7418070.

[41] I. M. Park *et al.*, "Kernel methods on spike train space for neuroscience: A tutorial," *IEEE Signal Processing Magazine*, vol. 30, no. 4, pp. 149–160, 2013. DOI: 10.1109/MSP.2013.2251072.

[42] M. Stimberg *et al.*, "Brian 2, an intuitive and efficient neural simulator," *eLife*, vol. 8, F. K. Skinner *et al.*, Eds., e47314, Aug. 2019. DOI: 10.7554/eLife.47314.

Marcello Zanghieri (Graduate Student Member, IEEE) received his M.Sc. in Physics from the University of Bologna, Italy, in 2019. He is currently working toward his Ph.D. in Data Science and Computation under the supervision of Prof. L. Benini at the Energy-Efficient Embedded Systems Laboratory (EEES Lab), Department of Electrical, Electronic, and Information Engineering, University of Bologna. His research interests focus on time series analysis with machine learning and deep learning, focusing on sEMG, ultrasounds, and EEG to advance human-machine interaction based on low-power computing platforms.

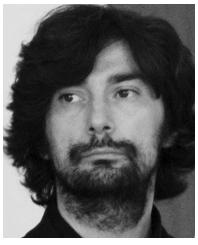


Pierangelo Maria Rapa received his M.Sc. degree in Electronic Engineering from the University of Bologna, Italy, in 2022. He is currently working on his Ph.D. in Automotive Engineering for Intelligent Mobility under the supervision of Prof. S. Benatti at DEI department, University of Bologna. His research interests include biosignals and ultra-low-power HMIs with specific emphasis on their application in the automotive industry.

Mattia Orlandi (Graduate Student Member, IEEE) received his M.Sc. degree in Artificial Intelligence from the University of Bologna, Italy, in 2022. He is currently working toward his Ph.D. in Data Science and Computation under the supervision of Prof. S. Benatti at the Energy-Efficient Embedded Systems Laboratory (EEES Lab), DEI Department, University of Bologna. His research activities involve bio-signal processing with machine learning on low-power computing platforms. He is investigating how to decode EMG signals into spike trains to develop advanced human-machine interfaces.

Elisa Donati (Member, IEEE) received the B.Sc. and M.Sc. degrees (cum laude) in biomedical engineering from the University of Pisa, Pisa, Italy, and the Ph.D. degree in biorobotics from the Sant'Anna School of Advanced Studies, Pisa. She is currently a Research Fellow with the Institute of Neuroinformatics, University of Zurich and ETH Zurich. Her research activities include the interface of the neuroscience and neuromorphic engineering. She is interested in understanding how the biological neural circuits carry out the computations and apply them in biomedical application and neurorobotics. She was the Co-Coordinator of the H2020 EU CSA Project NEUROTECH.

Luca Benini (Fellow, IEEE) holds the chair of digital Circuits and systems at ETH Zurich, Switzerland, and is Full Professor at the Università di Bologna, Italy. He received his Ph.D. from Stanford University. Dr. Benini's research interests are energy-efficient parallel computing systems, smart sensing micro-systems, and machine learning hardware. He is a Fellow of the ACM and a member of the Academia Europaea. He is the recipient of the 2016 IEEE CAS Mac Van Valkenburg Award, the 2020 EDAA Achievement Award, the 2020 ACM/IEEE A. Richard Newton Award, and the 2023 IEEE CS E.J. McCluskey Award.



Simone Benatti (Member, IEEE) received the Ph.D. degree in electrical engineering and computer science from the University of Bologna, Bologna, Italy, in 2016. Currently, he has been appointed as an Assistant Professor at the University of Modena e Reggio Emilia. He has collaborated with several international research institutes and companies. Previously, he worked for eight years as an Electronic Designer and R&D Engineer of electromedical devices. In this field, he has authored or coauthored more than 90 papers in international peer-reviewed conferences and journals. His research interests include energy-efficient embedded wearable systems, signal processing, sensor fusion, and actuation systems. This includes hardware/software codesign to address performance, as well as advanced algorithms efficiently.