Smart and Robust Biomedical Interfaces

Andy Zhou

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2023-29
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-29.html

May 1, 2023

Smart and Robust Biomedical Interfaces

by

Andy J Zhou

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering - Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Summer 2021

Smart and Robust Biomedical Interfaces

Abstract

Smart and Robust Biomedical Interfaces

by

Andy J Zhou

Doctor of Philosophy in Engineering - Electrical Engineering and Computer Sciences

University of California, Berkeley

Professor Jan M. Rabaey, Co-chair

Professor Rikky Muller, Co-chair

Wearable biosensing devices are becoming more and more ubiquitous, and new medical devices are being used to relieve previously untreatable diseases. Smart biomedical interfaces such as these depend on some form of device intelligence, enabling local processing of recorded biosignals to perform tasks such as hand gesture recognition or detection of neurological disease states. The devices and the algorithms they employ, however, must be robust to a variety of interferers and confounding factors that often prevent their practical use.

This thesis descries methods to improve robustness in implantables and wearables through a variety of means, demonstrating them in two different applications. The first part presents the Wireless Artifact-free Neuromodulation Device, or WAND, which enables closed-loop neuromodulation by cancelling self-interference artifacts that normally prevent simultaneous neural recording and stimulation. The second part then describes the FlexEMG system for hand gesture recognition using electromyography, which employs a hyperdimensional computing algorithm capable of incrementally learning hand gestures in a variety of situational contexts such as changing limb positions. These systems have been deployed in realistic animal and human subject experiments that better approximate real-world use to demonstrate the hardware and algorithmic improvements towards robustness.

To my parents.

Contents

List of Figures

List of Tables

Acknowledgments

Pursuing my PhD has been an incredibly rewarding experience, and there are so many people I need to thank. Family, friends, colleagues, and mentors have all taught me so much and made this journey unforgettable.

My warmest thanks go to my advisors, Jan Rabaey and Rikky Muller. I've always told anyone who has asked (and plenty of people who haven't) that I feel extremely lucky with my situation being co-advised by Jan and Rikky throughout my PhD. Jan was one of the first professors that I talked to in my grad school selection process, and I remember being amazed by his vision. It was not only the things that he and his group had already accomplished, but also the plans he had for the future, and I really wanted to be a part of that. Right as I started at Berkeley in Jan's group, Rikky joined as a new professor. I found myself very organically working with her more and more, learning so much about research, academia, and in general how to create work that I can really be proud of. Jan and Rikky's guidance has immensely shaped how I've approached the tough, yet fun, challenges of research, and I can't be thankful enough for how wonderful and supportive both of them have been.

I would like to thank the additional members of my qualifying exam and dissertation committees, Jose Carmena and Bruno Olshausen. Collaborating with Jose and his group opened my eyes to the huge field of neuroscience. His practical advice was always spot-on, and his enthusiasm always motivating. Bruno has been an awesome guide to the world of HD computing and computational neuroscience, as well as a fantastic model for expressing complex ideas thoughtfully and eloquently. I'd also like to thank Ana Arias for her guidance and collaboration in merging technologies within a large system and also effectively communicating the results we've achieved.

So much of the work in this thesis was done in collaboration with a huge case of amazing collaborators. Ali Moin has basically been there with me every step of the way, and working with him has always felt natural. We've always had so much fun in and out of the lab, and I can't think of a better mentor and teammate to have accomplished this with. George Alexandrov has also been an amazing mentor, collaborator, and friend. I've learned from him as a fresh grad student, and I'm thrilled to continue doing so in the next stage of our careers. I owe a debt of gratitude to Samantha Santacruz and Ben Johnson, who helped me out so much early on in a huge project where it would have been so easy to get lost. It was also a pleasure to work with Alisha, Abbas, Simone, Yasser, Ting, Justin, Ryan, and all of my other collaborators.

Outside of school, I have to shout out the Tribe, or the Trash Kings and Queens, or the Princes of the Pristine Palace, or whatever very serious name we've given to ourselves now. I've had such a great time with this huge group of people over the last six years, including Chris, my fellow Hometown Hero who I've known now for the majority of my life. I've been so fortunate to be around so many people from my home in Louisiana and from Caltech, and also to have met so many amazing friends here in Berkeley.

One of the people I met during my first ever visit to Berkeley is my incredible girlfriend, Keertana Settaluri. We've been through everything together, from navigating basically being deskmates at BWRC to adopting our kittens, Max and Pascal. Keertana, I've been so lucky to have you by my side, sharing so many similar experiences while offering a fresh perspective on things. You're my best friend, and I've learned so much from you on how to just be a better human. I can't wait to see what's next in our lives together.

Finally, I would like to thank my parents, Mei and Jing, who have literally been with me through everything. I owe everything to you, and nothing gives me more pleasure than knowing I've made you proud. You both have supported me every step of the way, including my decision early on not to try and become that *other* kind of doctor. I still have so much to learn from you, and so many ways to be inspired by you.

Chapter 1

Introduction

1.1 Smart wearable and medical devices

With the rise of the Internet of Things, computers have been incorporated into our cities, into our homes, and onto, as well as into, our bodies. Wearable computers provide the ability to continuously track our health and fitness through vital signs and activity monitoring; provide new and inventive modalities for interpersonal as well as human-computer interaction (HCI); and enable monitoring of, treatment against, and rehabilitation from diseases that affect large portions of our population (Figure 1.1).

Consumer smart watches and fitness bands, such as the Apple Watch or Fitbit, can measure physiological signals like heart rate, electrocardiogram (ECG), and blood oxygenation level, while also tracking activity such as exercise and sleep. There is growing interest, as well, in smart glasses and hearable technologies worn on or in the ears, which can correct or augment a user's senses and potentially allow non-invasive access to neurological signals [1]. The number of adult smart wearable users in the US is expected to eclipse a quarter of the population by 2021 and continue to grow through 2024, with 80% of studied consumers saying they would use wearable fitness devices [2].

The rise in popularity of augmented and virtual reality (AR/VR) applications has brought with it a need for better methods for user input, which may also be provided by smart devices. For truly immersive experiences, consumer electronics must be able accurately measure user intent with high throughput, something that has been limited by traditional mechanisms relying on text, speech, or physical controls with limited input channels. Wearable, non-invasive neuromuscular interfaces show promise in providing more natural and direct control for HCI, relying on recording and learning from biosignals to interpret the wearer's intentions [3, 4].

In the medical field, smart wearable and implantable sensors have enabled 24-hour, in-home vital signs monitoring, giving physicians continuous access to better assessments of baseline health, improved data for making diagnoses, and the ability to provide fast emergency response or prevention. Through these means, better at-home healthcare can also reduce the length and frequency of hospital visits. Smart medical devices can range from simple monitoring systems that unobtrusively measure vital signs much like consumer wearables [5], to complex, closed-loop systems that must be invasively implanted to deliver therapy in response to the patient's measured state of health [6, 7].

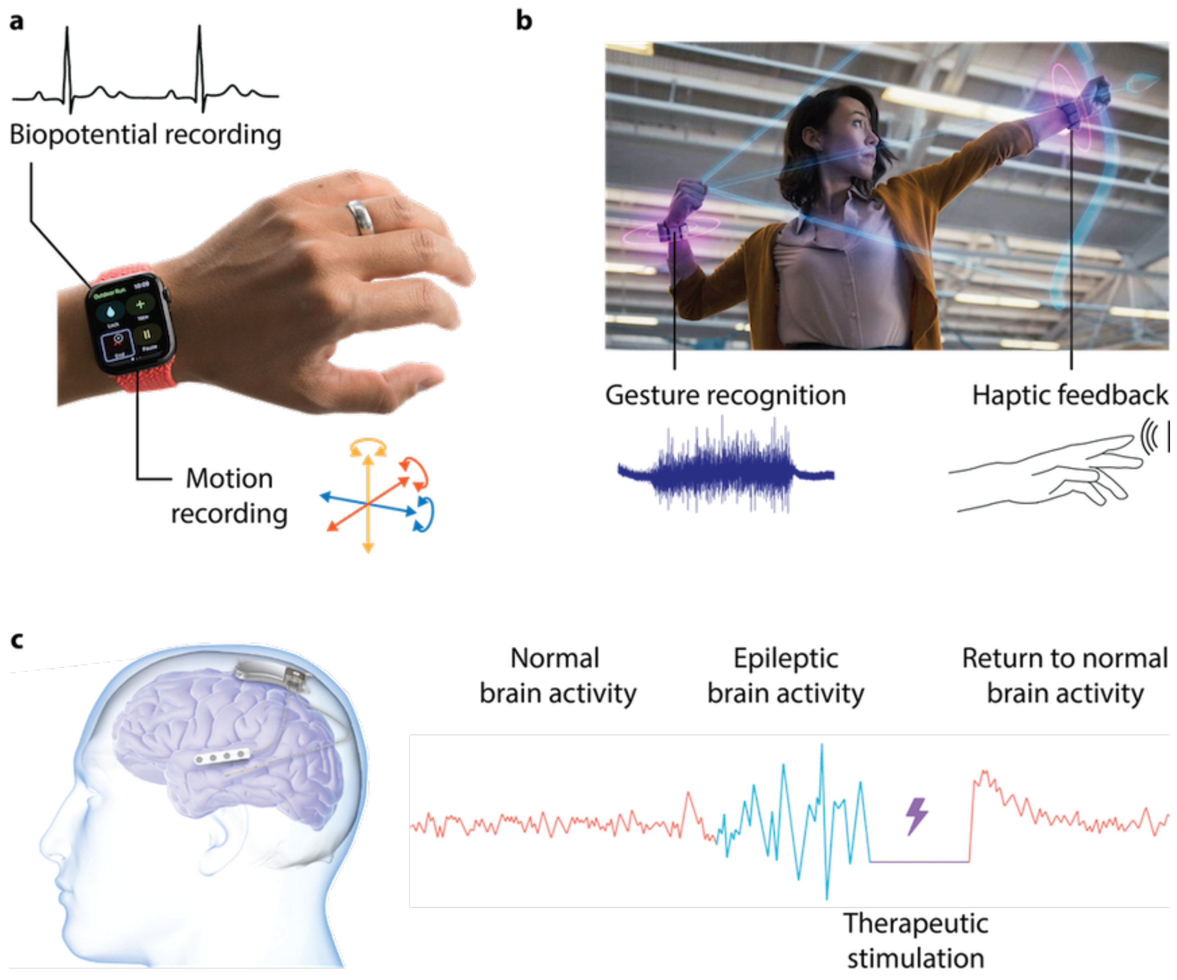


Figure 1.1: **Examples of smart wearables and medical devices. a)** Consumer smart watch with capability to record biopotentials for health monitoring, as well as motion sensors for activity recognition. Adapted from [8]. **b)** Augmented/virtual reality wearable devices for tracking user movements, recognizing gestures, and providing haptic feedback in virtual scenarios. Adapted from [9]. **c)** Implantable medical device for seizure detection and responsive therapy. Adapted from [10].

Wearable and implantable devices interact with the human body either through sensing some biological phenomenon or through actuation to produce a biological effect. Across a broad range of applications, device complexity levels, and levels of invasiveness, most devices share a few basic building blocks (Figure 1.2).

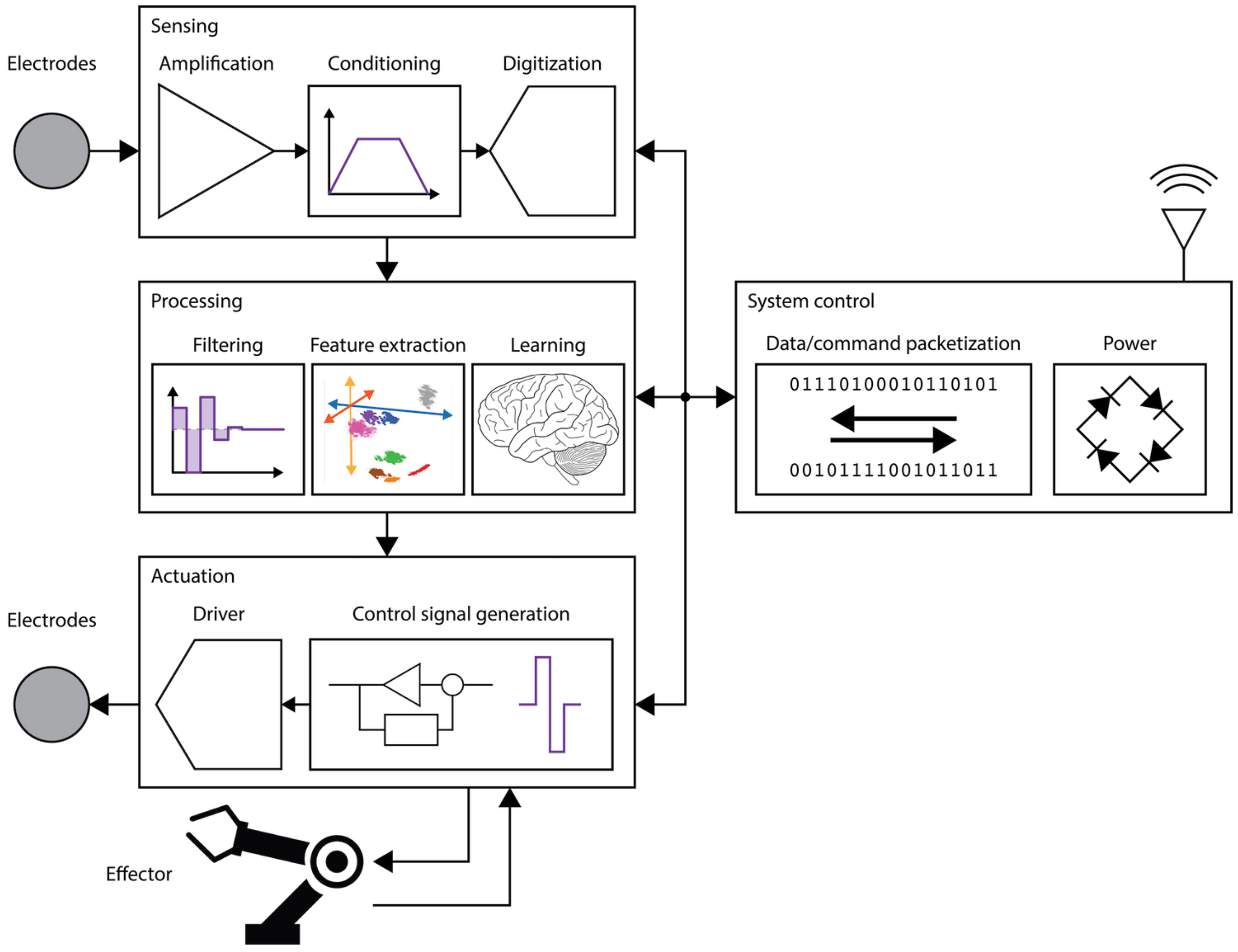


Figure 1.2: **General block diagram for smart wearable or implant.** Main components include a sensing module for recording physiological signals, a processing module consisting of the "brains" of the device, an actuation module for effecting physiological change or controlling additional systems, and a control module for powering and communication.

First, there is the interface to the body, which can occur through transduction of a variety of modalities (e.g., chemical, motion, or magnetic) into electrical signals. In this thesis I will focus on biopotentials, or physiological signals that are already electrical by nature. These can be measured using microelectronic circuits through electrodes that serve as the interface between the device and biological matter. As with many other applications in which sensing electrical signals is required, the sensing block usually consists of signal amplification, conditioning, and digitization. This converts the analog biopotential into digital values which are more suited for processing and computation.

In the other direction, devices may also have actuation blocks which allow them to effect change, either through physiological channels or by controlling additional devices like a robotic arm. Many medical implants, such as cardiac pacemakers and deep brain stimulators, are capable of actuation only, delivering electrical stimulation at a constant rate through electrodes back to the tissue until the device is turned off or reconfigured. In these cases, the actuation sub-system is directly controlled by the user or physician, relying on a communication channel for data up-link and command down-link. The system control block handles this communication, as well as powering the device. Ideally powering and communication are wireless, enabling safer implantation and more mobility when using the device.

Finally, the devices can have some form of on-board processing, which is what makes them "smart" and can tie the sensing and actuating blocks together. It is often beneficial for wearable and implantable devices to have more complex on-board processing abilities in addition to data streaming or logging. Performing computation locally precludes the need to stream out large amounts of raw data, which in turn can increase privacy and security, lower processing latency, and potentially reduce power consumption. Recently, improvements in integrated circuits have enabled unprecedented amounts of computation to be performed with low power requirements and in small form factors. Processing steps may include extraction of useful features from the recorded data, as well as machine learning for interpretation of the data. Decisions made by more complex processing blocks can be used to identify certain patterns from the sensed biosignals and adjust actuation controls in response. An example of this is the Neuropace RNS system [6] for responsive seizure detection and therapeutic stimulation, depicted in Figure 1.1c. Recorded neural activity is analyzed for patterns related to seizures. When detected, therapeutic stimulation is triggered, relieving the patient's symptoms and "correcting" the neural state. This is what is meant by closing the loop in neural interface devices.

1.2 Challenges facing robust performance

Much attention has been given to building smart devices and developing the algorithms that they employ, with the goal of accurately identifying or predicting patterns in the recorded signals. However, the human body can be a noisy environment for deploying these algorithms. Performance of a biosignal processing algorithm or machine learning model can be degraded due to external interferers and confounding factors, which reduce signal quality and inhibit accurate extraction of features on which the algorithms rely. Interferers can include random or correlated noise from the recording interface or background biological activity; large, environmental signals like power-line interference; or artifacts arising from non-ideal conditions such as motion and self-interference. Problems with confounding factors arise due to incomplete or biased data that is used for algorithm development. For example, initial training of a machine learning model may be limited to data that isn't representative of real-world usage, especially when training is specific to a user or patient and must be done in a time efficient manner. There exist a variety of methods to address interfering and confounding factors, which I broadly split into two categories: 1) modeling and anticipating them, and 2) learning and adapting to them.

1.2.1 Anticipating and modeling interferers

Because many interferers are known to designers at the outset, they can be characterized and then mitigated with targeted strategies based on their known properties. Typical interferers that can be modeled and dealt with in this way include electromagnetic interference, signal drift, motion artifact, and self-interference. Strategies for cancelling the interference can be very basic, such as using a 50 or 60 Hz notch filter to cancel out power-line interference or using AC-coupled recording front-ends to reject large DC offsets and drift caused by unstable electrode-tissue interfaces. On the other hand, when interferers have more complex characteristics, algorithm designers must first understand their key properties and how they originate to design effective interference cancellation. Once characteristics that differentiate them from the signals of interest have been found, strategies for preventing or mitigating the interference can be built into multiple parts of the signal chain.

1.2.2 Learning and adapting to confounders

The above method relies on the ability to know beforehand what kind of interfering and confounding factors will be present during device operation. While some confounding factors can be modeled and handled in a way similar to interference cancellation, it is often more practical to instead rely on statistical machine learning models to capture their effects. This is because it can be difficult or even impossible to hand-design very accurate models for a confounding effect, which would rely on *a priori* knowledge of how the confounder is produced and how it interacts with the signals or features of interest. Large amounts of data must first be collected to learn this information, and new confounders may be discovered after deployment of a strategy, requiring re-engineering of the model.

Machine learning can enable design of algorithms that handle confounding factors without being explicitly instructed by the implementer on how to do so. These algorithms often rely on statistical methods, which means a large amount of data must be gathered for the algorithm to fully "understand" the problem. Because of this, continuous or incremental learning has become quite popular recently for addressing problems where new information must be continuously incorporated into classification models as the amount of relevant data grows or changes over time.

1.3 Scope of this thesis

This introduction gave a broad overview of smart wearables and implantables, as well as of the problems affecting the robustness of the algorithms that they employ. In the rest of this thesis, I will present the work I've done to address this challenge in various ways. Almost all of the results discussed here involve a low-power device platform developed in a joint effort between numerous collaborators from various research groups and industry partners. In my research, we've used this platform for the medical application of closed-loop neuromodulation, and it has also been adapted for non-invasive electromyography (EMG)-based gesture recognition, which has applications in both consumer wearables for AR/VR and prosthetics control for rehabilitation.

In Chapter 2, I cover closed-loop neuromodulation, where the main challenge to robustness that I address is self-interference due to stimulation artifact. Stimulation artifact is a known source of signal degradation that arises when simultaneously recording neural activity and delivering electrical stimulation to the brain, hindering the extraction of neural biomarkers. Because the source of the artifact is the device itself, we can model and build an understanding of how and when artifacts arise, allowing us to remove them through system co-design of the different device building blocks. Only then can we have truly concurrent sensing and stimulation for closed-loop neuromodulation.

Chapters 3 and 4 describe a different approach for improving robustness in EMG-based gesture recognition, where machine learning techniques are used to identify which gestures a user is performing based on their measured neuromuscular activity. Typically, gesture classifiers are trained in controlled settings that don't fully match what they would be deployed in, limiting the accuracy of the classification models during real-world situations. Variations due to everyday use of an EMG-based gesture recognition device introduce confounding factors from changing the upper limb position, instability of the electrode interface, and electrode shift, all of which must be learned by the model for generalization to a larger number of situational contexts. This can be achieved by introducing data from additional sensors, providing more information about both the gestures and the confounders. Additionally, as the amount of required training data grows, the training process can be split into multiple, smaller steps for incrementally updating the model only when it is necessary.

The following chapters are organized as follows:

- Chapter 2 begins with a review of techniques for stimulation artifact prevention, mitigation, and cancellation in closed-loop neuromodulation systems. I then present WAND, the Wireless Artifact-free Neuromodulation Device, which incorporates a number of strategies in the co-design of multiple system blocks to achieve artifact-free recording during stimulation. WAND's capabilities are demonstrated through *in vivo* in experiments with a non-human primate, showing the ability to run simple closed-loop neuromodulation algorithms.

- Chapter 3 describes the system adapted for real-time, in-sensor classification of hand gestures using electromyography. Newly dubbed the FlexEMG system, the WAND platform was re-purposed for non-invasive EMG acquisition and processing, leveraging a custom screen-printed electrode array that could be worn around the forearm. Gesture classification was performed using hyperdimensional (HD) computing, a neuro-inspired computing paradigm capable of fast, incremental learning. Results from a full in-sensor implementation of the HD algorithm show that the classification model can be updated using solely on-board computations to improve generalization in new situational contexts.

- Chapter 4 extends the work from Chapter 3 to demonstrate classification improvements using HD computing in a larger number of situational contexts. For this, I conducted gesture recognition experiments in 8 different limb positions to demonstrate the negative impact of limb position variation on classification accuracy, as well as how to counteract that degradation. Key results include an efficient way to incorporate limb position information into the classification model through sensor fusion, as well as an analysis of different configurations for training the HD classification model *incrementally*.

- Chapter 5 concludes the thesis and discusses possible future directions for the work.

Chapter 2

Artifact-Free Closed-Loop Neuromodulation

Electronic neural interfaces allow us to record brain activity, as well as modulate it through electrical stimulation. Algorithms for recognizing patterns and brain states from recorded neural signals can be used to drive closed-loop neuromodulation for optimized therapy in treating neurological disorders. However, self-interference from stimulation artifact during simultaneous sensing and stimulation is a major issue for closed-loop neuromodulation systems. In this chapter, I first describe the origin of stimulation artifacts and review methodologies for cancelling them or reducing their effects. I then present the Wireless Artifact-Free Neuromodulation Device (WAND), with which my collaborators and I achieve effective artifact cancellation using system-level co-design of the neural stimulator, neural recording front-end, and signal processing blocks. WAND provides a platform enabling research applications which require high-throughput and high channel count data streaming, low-latency biosignal processing, and simultaneous sensing and stimulation. We demonstrate its abilities through *in vivo* recording, open-loop stimulation, and closed-loop experiments with a rhesus macaque.

2.1 Introduction

Recent times have seen a surge of clinical applications for neural recording and stimulation in the treatment of neurological disorders, largely as a result of major investments into neuroscience research such as the Brain Research through Advancing Innovative Neurotechnologies (BRAIN) Initiative. For instance, deep brain stimulation (DBS) is a form of therapy widely researched and clinically prescribed for the treatment of Parkinson's disease [11–13]. Additionally, responsive neurostimulation for treating epilepsy uses recorded neural biomarkers to trigger therapeutic stimulation [6, 14–16]. These therapies depend on devices that provide an electrical interface to the brain, enabling both electrical stimulation to modulate neural activity and recording of neural biopotentials for disease state monitoring and research. Figure 2.1 shows example clinical electronic neural interfaces (including the NeuroPace RNS offering seizure detection and responsive therapy and the Medtronic Activa RC offering DBS for treatment of movement disorders) as well as how a neural interface may be implanted. The main unit, housing the pulse generator, processing unit, and power supply, is implanted in a cavity in the skull. In the case of the NeuroPace RNS device, the unit is connected to two leads: one for recording electrocorticography (ECoG) from the surface of the brain, and another penetrating the cortex for delivering stimulation to targeted anotomical regions. These devices are extremely invasive, only used in cases of severe and advanced disease. Still, they have been implanted in hundreds of thousands of patients to provide tremendous quality-of-life improvements in ameliorating symptoms that have been otherwise untreatable.

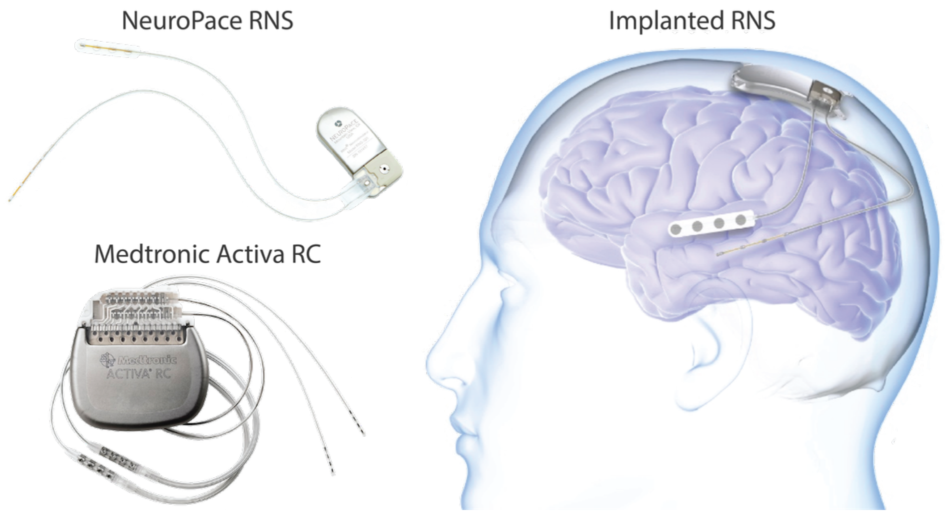


Figure 2.1: **Example clinical neural interfaces.** Some widely used electronic neural interfaces are used to treat neurological diseases and include the NeuroPace RNS (device upper left, implantation cartoon right) and the Medtronic Activa RC (lower left). These devices typically contain a pulse generation unit, stimulating leads, and, in the case of the NeuroPace RNS, a sensing ECoG lead. Image sources: [10, 17]

2.1.1 Electronic neural interfaces

The electrical nature of neural activity lends itself well to excitation and observation using modern microelectronic circuits. The basic signaling and processing unit of the nervous system is the neuron, a cell which can produce electrical activity depending on inputs from other neurons or from sensory stimuli [18] (Figure 2.2). When the input potentials to a neuron satisfy certain timing conditions and cause the voltage across the cell membrane to cross a certain threshold, the receiving neuron generates a large spike in voltage called an action potential. These spikes are characterized as brief depolarizations and repolarizations of the membrane voltage (Figure 2.2) which can withstand long-distance transmission to different regions of the nervous system and in turn drive the activity of other neurons. The spiking of an individual neuron can encode information through its rate or timing, and the collective spiking of vast, complex networks of billions of neurons encodes human function, experience, and intent. Neural interfaces include both afferent (stimulation) devices that affect neural activity and efferent (recording) devices which are described below.

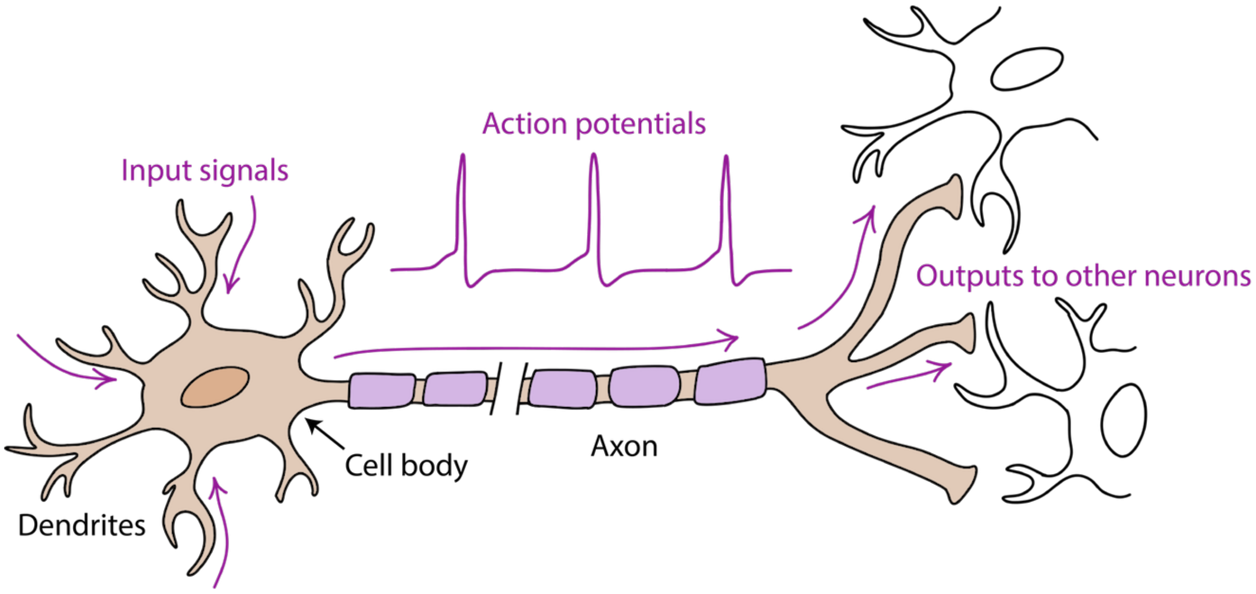


Figure 2.2: **Neuron with flow of information.** Inputs from other neurons come in at the dendrites and are integrated in the cell body. Upon integration of the right inputs, action potentials are produced and travel down the axon to other neurons.

Electrical neural stimulation

The cell membrane of a neuron can be modeled as a capacitance, and application of charge from an external source produces similar changes in polarization that can artificially induce or inhibit spiking activity. This is the fundamental idea behind electrical neural stimulation, and the required charge can be delivered using an electrode connected to a current source and placed near the body of a target neuron. The effects of electrical stimulation were observed in experiments as early as the 18th century, where dead frogs' muscles were made to contract by stimulating their nerves [19]. Modern devices such as retinal [20] and cochlear [21] implants are able to produce visual and audio percepts for the seeing and hearing impaired by stimulating sensory neurons with different patterns. Since the early 2000s, neurostimulation in deep, subcortical brain regions has been approved by the FDA to treat a wide variety of neurological diseases, including essential tremor, Parkinson's disease, and epilepsy. Although the exact mechanisms behind therapeutic stimulation are still unclear, most models suggest that electrical stimulation serves to disrupt or prevent abnormal neurological activity like excessive oscillations [22] and adverse discharges [23].

Electrical neurostimulation patterns are typically hand-chosen by physicians based on the measured efficacy of the treatment. Due to the inexact nature of electrode-tissue interface impedances, stimulation pulses are typically current-controlled rather than voltage-controlled in order to deliver more precise amounts of charge. Figure 2.3 illustrates some common configurations for delivering stimulation. The most basic stimulation waveform is a single cathodic pulse (Figure 2.3a), which is the most likely to induce action potentials. Stimulation may be delivered in a monopolar configuration (Figure 2.3c), where the return path for the current is distant and is typically the device's packaging, or in a bipolar configuration (Figure 2.3d) in which the return path is through another local electrode. It must be noted that the area of effect for stimulation can be rather large, inducing activity in a region of neurons rather than any specific neuron. This can be slightly better controlled through the use of current steering by providing multiple return paths and delivering current through multiple electrodes [24].

A single cathodic pulse may leave residual charge at the electrode interface, potentially harming neural tissue and damaging electrodes through corrosion. Therefore, a passive recharge phase may be added after each monopolar pulse to short the source electrode and return path and clear built up charge. Biphasic stimulation (Figure 2.3b), consisting of a cathodic pulse followed by an anodic pulse in the opposite direction, is more commonly used to promote balance in the total amount of charge delivered. Typical tunable parameters include amplitude or duration of each of the pulses, the interphase gap or delay between the two phases of a biphasic pulse, and the period between successive pulses in a stimulation pulse train. Bipolar stimulation configurations can also benefit from additional passive recharge through shorting electrodes.

Recording neural biopotentials

Working in the reverse direction, recording neural signals can be performed by connecting those same electrodes to an electrical amplifier and analog-to-digital converter (ADC) to capture the signals produced by nearby neurons. With the proper hardware, spikes produced by individual neurons can be recorded to decode neural activity. It is very hard to be extremely selective in terms of which neurons are being recorded due to the difficulty of fine positioning of electrodes as well as the general propagation of electrical signals. A single recording electrode will typically pick up the activity of not only the nearest neurons, but also of the larger population of neurons within a local sphere (Figure 2.4a). The superposition of this activity, which includes both spiking and other subthreshold potentials, is another type of recorded signal called the local field potential, or LFP. Recording of LFP can be more robust, as it depends less on precise electrode positioning and is effected less by local gliosis due to implantation [25]. A recorded neural trace with sufficient bandwidth will

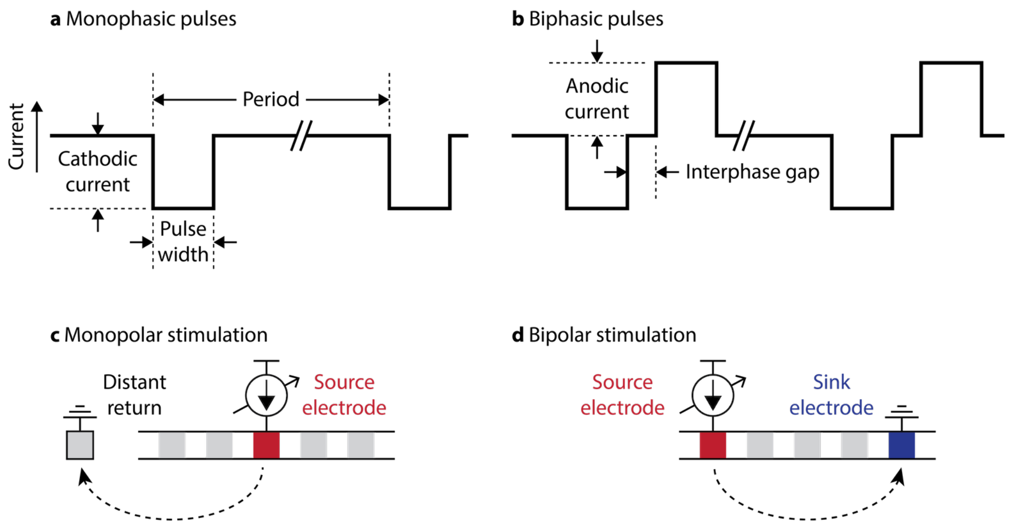
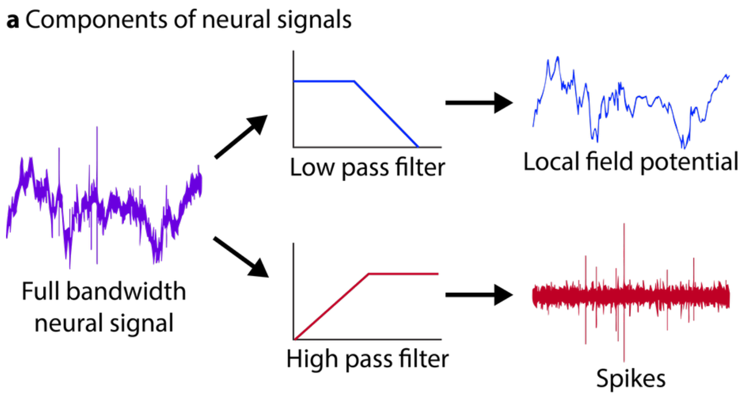


Figure 2.3: **Basic stimulation configurations. a)** The most basic stimulation waveform consists of a single cathodic pulse, which depolarizes target neuron populations. Tunable parameters include the amplitude and pulse width of the cathodic current pulse (dictating the total charge delivered), as well as the period between successive pulses. **b)** An anodic pulse can be added to balance charge delivery in biphasic pulses. Typically the cathodic and anodic pulses are charge balanced. An interphase gap may be added between the two phases to improve stimulation efficacy. **c)** Stimulation may be delivered in a monopolar configuration, where only a single electrode is active and the return path for the current (which could be the device packaging) is distant. **d)** A bipolar stimulation configuration utilizes two electrodes: one to source current, and one to sink it.

contain both spiking (higher frequency, 250 Hz - 10 kHz) and LFP (lower frequency, 1-500 Hz) components. Typical amplitudes of these signals recorded from intracortical penetrating electrodes range from millivolts for spikes down to a few microvolts for LFP.

Because action potentials from multiple neurons may be included in recordings, a typical first processing step after digitization is spike sorting. Different spike morphologies are identified as being produced by different neurons, and newly detected spikes are then assigned to different neurons for counting and timing purposes. For robust spike sorting, recorded neural data must be low noise and low distortion and have a high enough sampling rate to be able to differentiate different spikes.

For LFP recordings, we are less interested in the time domain characteristics of the signal, and we focus more on the amplitude and phase modulation of its frequency components. Oscillatory behavior, as seen in LFP, is generally healthy and allows for dynamic communication and plasticity between spatially distant neuron populations [27]. LFP is often divided into smaller frequency bands as shown in Figure 2.4b, with lower frequencies being associated with sleep and relaxation, while higher frequencies are associated with movement and



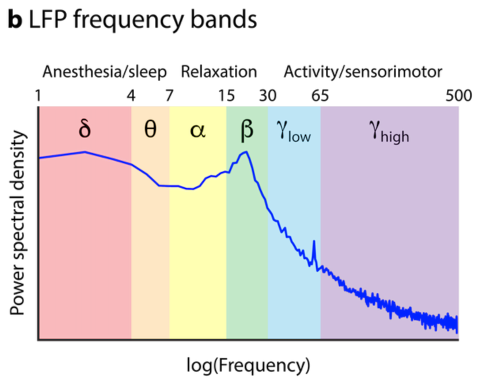


Figure 2.4: **Makeup of recorded neural signals. a)** Neural signals record activity from a large population of neurons. High frequency components consist of spikes from the most proximal cells, while lower frequency components are local field ptentials that consist of superimposed population acivity. Adapted from [26]. **b)** Local field potential (LFP) can be subdivided into different frequency bands that modulate with different activities and brain states.

sensory activity. Many signs and symptoms of movement disorders relate to abnormal oscillatory synchronization. For instance, excessive beta oscillations can lock neural networks into ineffective states, inhibiting voluntary movement [22].

2.1.2 Closed-loop neuromodulation

Neural interfaces can be bi-directional (capable of both afferent and efferent interactions), though most applications of neurostimulation are open-loop, in that stimulation parameters are pre-programmed, and stimulation pulses are triggered externally or scheduled in advance. For example, Medtronic DBS devices are often programmed to continuously deliver stimulation until toggled by a physician or patient using an external device. Although this practice is generally effective, it is far from optimal. Continuous stimulation can cause side effects such as impaired speech, dyskinesia, or cognitive impairment [28], and excess stimulation unnecessarily drains precious device battery power. Ideally, clinical neural interfaces would be able to autonomously provide personalized, optimal treatment that is halted when side effects are induced and that only acts when there is a demand for therapy. This requires "closing the loop" by continuously sensing the patient's neural state and tuning therapy parameters (Figure 2.5). Looking forward, closed-loop therapies can be utilized to treat complex conditions whose symptoms are not always present, necessitating simultaneous sensing, biomarker computation, and targeted therapeutic stimulation in the brain.

In order to deliver closed-loop neuromodulation, devices must be bi-directional while also possessing some processing capabilities to run closed-loop algorithms on-board. One

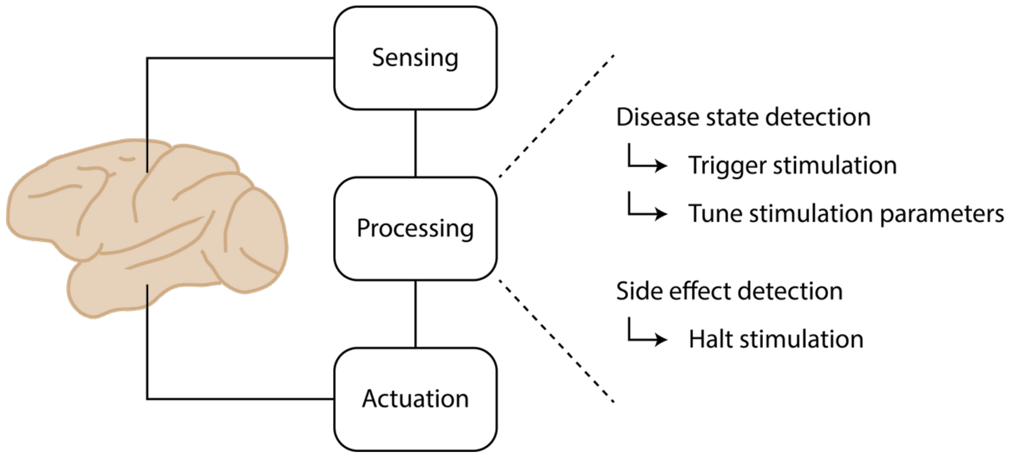


Figure 2.5: **Closing the loop on neuromodulation.** By continuously sensing the neural state through recording and local signal processing, stimulation parameters can be adaptively tuned for optimal effect.

of the more successful closed-loop neuromodulation devices is the NeuroPace RNS system for responsive neurostimulation. The device senses electrocorticographic (ECoG) activity and monitors it for abnormal, epileptiform activity. Once such activity is detected, the device is triggered to deliver cortical stimulation which disrupts that activity and treats the seizure [23]. Closed-loop DBS has also recently been proposed as an improvement over open-loop DBS for treatment of Parkinson's disease where stimulation is only delivered when the patient is symptomatic, improving battery life and reducing side effects [29]. A gamma band oscillation biomarker has been proposed for detection of dyskenisia as a result of stimulation and can be used to abort therapy when sensed [30].

2.2 The challenge of stimulation artifact

While responsive neurostimulators have shown a lot of promise and have made tremendous impact in healthcare, there are still some major limitations in using therapeutic stimulation in a closed-loop, on-demand manner. Stimulation pulses create interference with recording electronics, appearing as artifacts which mask the underlying neural signal and make simultaneous sensing and stimulation a challenge [31]. To continuously record neural activity while simultaneously delivering stimulation, neuromodulation systems must be resistant to large stimulation artifacts and be able to actively remove artifacts from the recorded signals for real-time biomarker computation. Figure 2.6 shows an example of adaptive closed-loop neuromodulation using LFP feedback. Stimulation artifacts contaminate the recorded signal and prevent robust detection of biomarkers. Note that even an out-of-band stimulation signal can corrupt the entire sensing spectrum. To enable real-time feedback, neuromodulation systems attempt to mitigate artifact through the electrode configuration, resilient recording front-ends, and back-end cancellation methods.

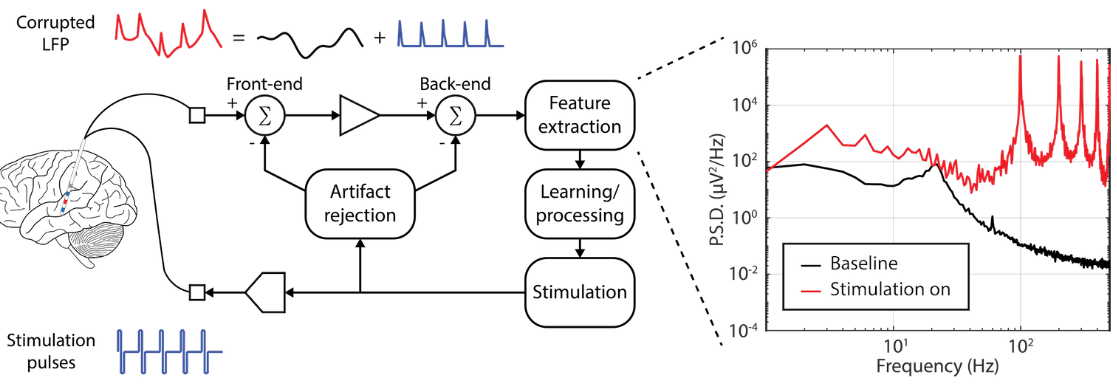


Figure 2.6: **Stimulation artifacts in closed-loop devices.** Often, stimulation artifacts can be orders of magnitude larger than the underlying neural signals, completely masking the power spectrum if not handled properly. Cancellation methods can be applied in the front-end analog domain, in the digital back-end, or both. Adapted from [32].

In order to effectively and efficiently remove or prevent stimulation artifacts, we must first gain an understanding of their origin. By modeling and understanding stimulation artifacts, the most appropriate measures for a given application can be chosen. In this section, we review several techniques that involve artifact-preventing system configurations, resilient recording front-ends, and back-end signal processing for removing recorded artifacts. Co-designing and integrating these artifact cancellation techniques will be key to designing neuromodulation systems that can stimulate and record at the same time. The information presented in this section is adapted from a review done in collaboration with Prof. Benjamin Johnson [32].

2.2.1 Origin of stimulation artifacts in neural interfaces

Stimulation artifacts typically consist of large voltage transients coinciding with the delivery of stimulation pulses. Their morphologies are dependent upon stimulator architecture and performance, stimulation waveform, and electrode configuration. Recorded artifacts consist of a short, high-amplitude peak (direct artifact) followed by a slow, exponential decay (residual artifact) superimposed on the local neural activity. Cancellation of these artifact components is crucial for analyzing spikes, as well as LFP and ECoG. Artifacts may be misclassified as spikes by some detection algorithms, and subsequent spikes cannot be recorded until the electrode voltage has returned to within the input range of the recording amplifier. The artifact peaks and ensuing decay together create strong distortions in the power spectrum at the stimulation frequency and also spreading into frequency bands of interest for LFP and ECoG (Figure 2.6).

To demonstrate the origin of stimulation artifacts, a simplified electrode model with biphasic current stimulation waveforms is shown in Figure 2.7. The electrode-tissue interface can be modeled with an equivalent circuit consisting of simple linear components: the electrode double-layer capacitance (C_{DL}) models charge stored at the interface, the charge transfer resistance (R_{CT}) models Faradaic exchange currents from oxidation or reduction, and the spread resistance (R_S) models tissue resistivity [33]. These parameters all depend on the electrode area, geometry, material, surface roughness, and surrounding medium, and example values are given for a 1 mm diameter platinum electrode in brain tissue.

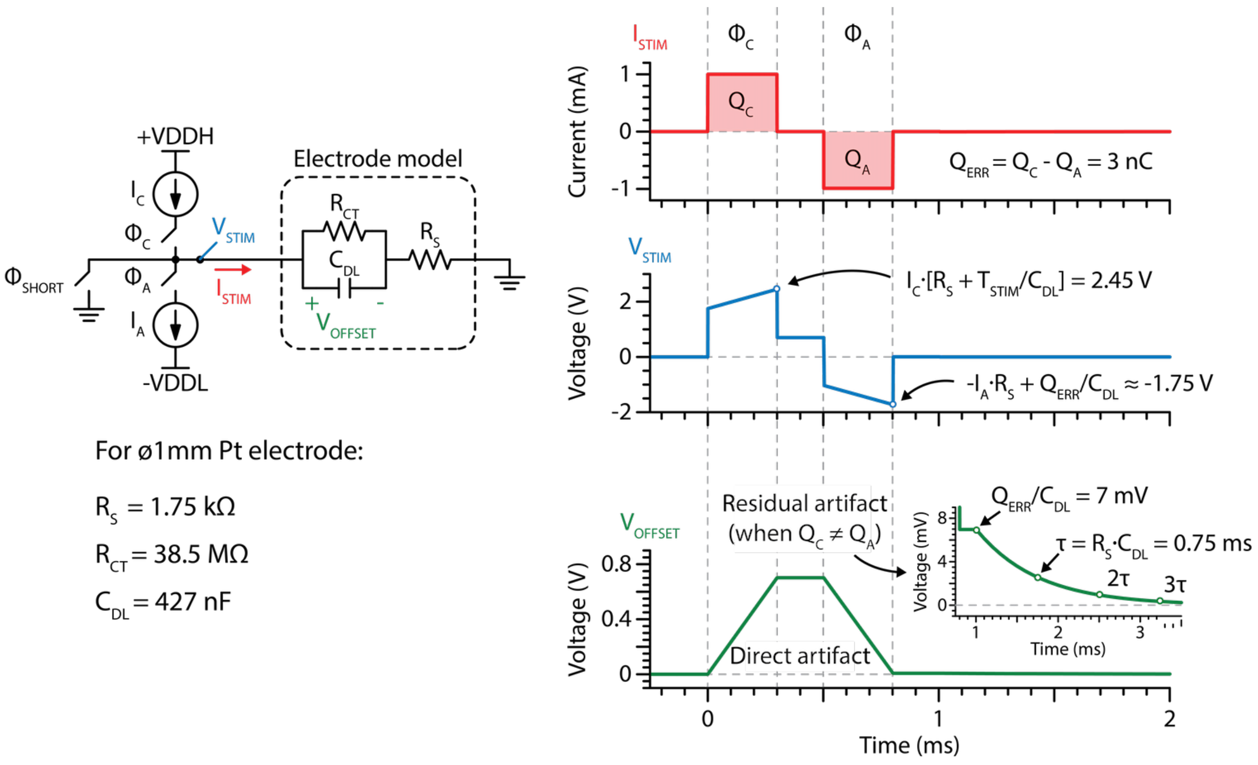


Figure 2.7: **Electrical model of stimulation electrode and waveforms.** Stimulation artifact, which can be measured as V_{OFFSET} across the double-layer capacitance, depends on the accuracy of the charge balance and the time constant associated with dissipating residual charge. Adapted from [32].

During the first phase Φ_C, a cathodic current I_{STIM} is delivered through the electrode interface, causing a voltage drop across the spread resistance and charging up the double-layer capacitance. The electrode is then disconnected from all sources for an interphase gap period, where the charge on the double-layer capacitance remains. This is followed by the anodic phase Φ_A during which current is sunk in the reverse direction, causing an opposite voltage drop across the spread resistance and discharging the capacitance. An additional shorting phase Φ_{short} (not shown in the waveforms) allows for further discharging after the anodic phase.

Ideally, the cathodic and anodic phases of biphasic stimulation are perfectly matched with the same amount of charge ($Q_C = Q_A$) and completely discharge the electrode capacitance, C_{DL}. In this case, the system would only experience direct artifact lasting the duration of the stimulation pulse. However, any mismatch in current amplitude ($I_C \neq I_A$) or pulse width ($\Phi_C \neq \Phi_A$) leaves residual charge on the capacitors, resulting in a persisting artifact voltage that discharges slowly. Residual artifact duration depends on the degree of mismatch and the time constant associated with dissipating residual charge. Aside from saturating recording front-ends, slow discharge can limit stimulation frequency since accumulated charge creates a DC current at the electrode, resulting in tissue damage and corrosion from electrolysis [33, 34].

This model only describes the generation of stimulation artifact on the electrode through which stimulation current is delivered. In addition, we can also model how that artifact spreads to neighboring electrodes. Because of the finite input resistances of neural front-end amplifiers, neighboring recording electrodes that aren't actively used for stimulation can still suffer both direct and indirect artifacts (Figure 2.8). This non-ideality has two negative effects: 1) it can further imbalance the amount of current or charge across different components during the anodic and cathodic phases of stimulation, and 2) even the small amounts of current through the recording electrode interface can generate large voltages compared to the neural signal amplitudes.

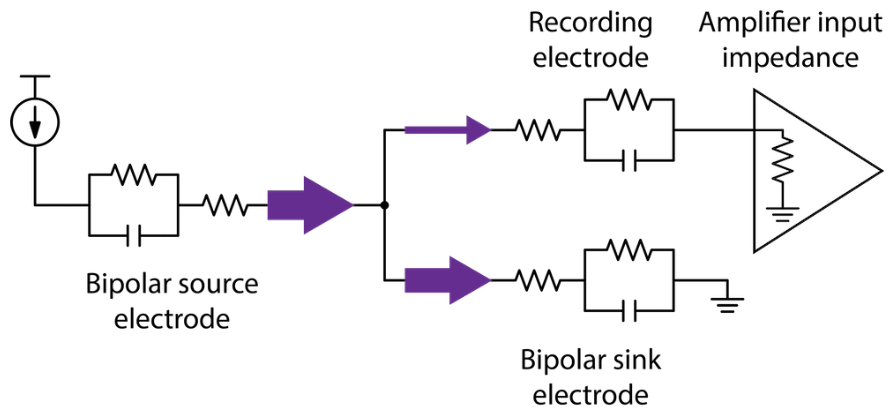


Figure 2.8: **Spread of stimulation artifact to neighboring recording electrodes.** Finite amplifier input impedances (and other return paths to ground) can hurt charge balancing and cause artifacts to appear on recording-only electrodes. As shown by the purple arrows, not all of the current sourced by the driven electrode is sunk through the return electrode.

Precise modeling of the artifact is difficult since electrode impedance is in reality nonlinear and varies with voltage. Furthermore, electrode impedance can change over time with chronic implantation [35]. The linear model, however, is effective for estimating the peak voltage due to stimulation and duration of an artifact [36–38], both of which are important in designing counter-measures against stimulation artifacts for specific system specifications and applications.

2.2.2 A taxonomy of artifact cancellation methods

Many aspects of device and experiment design all contribute to the generation and severity of stimulation artifacts. Table 2.1 describes a number of state-of-the-art solutions which we detail in the sections below. We first describe system configurations and preventative measures that attempt to reduce the amount of stimulation artifact that can be produced. Next, we discuss mitigation methods that enable cancellation or removal of recorded artifacts. These can be divided into categories of front-end circuit resiliency methods and back-end digital signal processing methods.

Preventative measures

Figure 2.9 describes methods to prevent the generation of stimulation artifacts, limiting the amplitude and duration of recorded artifacts and making them more manageable. As discussed above, a major source of artifacts is residual charge left on capacitive elements of the electrode-tissue interface resulting from mismatch between stimulation phases. Several techniques have been utilized to improve stimulator phase matching. Pulse timing is typically well-controlled, whereas different current sources commonly have an amplitude mismatch of 1%, which is enough to produce significant artifact [36]. Some systems monitor voltage across the double-layer capacitance at the electrode from accumulated charge and calibrate the second phase current to minimize this offset [39]. Other systems calibrate prior to stimulation via current-copying using the using the cathodic current source as a reference for the anodic current [40, 41]. Another alternative uses an H-bridge circuit, which is a stimulator topology that utilizes a single current source and a set of switches that allows the same current source to be used in both stimulation phases [42–44]. This method has been shown to achieve a mismatch between pulses of less than 0.02% [42].

Even stimulation pulses that are perfectly balanced with equal charge in each phase can generate large and long artifacts if recorded using inadequate circuits. Some work uses a tri-phasic stimulation waveform to minimize artifact duration [45]. Chu et al. [46] took an alternative approach by modeling the brain and interface as a communication channel and designing a waveform shape that inverts the transfer function, thereby reducing the artifact duration by 73%.

Deliberate placement of stimulation, recording, and reference electrodes has also been shown to reduce the stimulation artifact. For example, a symmetric configuration between stimulation and recording electrodes can present the artifact as a common-mode signal,

Methods	Variants	Toward closed-loop
Prevention		
Stimulation pulse charge-balance	Voltage offset correction [39] Current copying [40, 41] Current source reuse [42–44]	Improving charge balance reduces artifact size and duration, relaxing requirements on signal acquisition chain
Stimulation waveform design	Tri-phasic stimulation [45] Zero-forcing equalization of waveform [46]	Compensates for artifact-inducing properties of stimulator, neural tissue, and recording circuitry
Electrode and reference configuration	Symmetric stim and sense electrode geometry [11] Artifact-tracking voltage supply [47]	Keeps artifact common-mode, which can be tracked by the supply and also cancelled through differential amplification
Front-end techniques		
Saturation prevention	High dynamic range [42, 48] Front-end subtraction [49–52] Electrode disconnection [14, 53–56]	Keeps recorded artifacts linear, improving the performance of back-end techniques
Rapid recovery	Amplifier charge reset [42, 57] High-pass pole shifting [36, 58, 59] Active electrode discharge [36–38]	Recovers from saturation quickly, reduces data loss, and lowers requirements on front-end dynamic range
Back-end techniques		
Data reconstruction	Sample-and-hold interpolation (offline) [60, 61] Linear interpolation (offline) [62] Gaussian interpolation (offline) [63] Cubic spline interpolation (offline) [64]	Simplest to implement and is effective with relaxed SNR requirements
Artifact subtraction	Averaged template subtraction (online) [50, 59, 65] Averaged template subtraction (offline) [61, 66, 67] Averaged template resampling and subtraction (offline) [68, 69] Function fitting template subtraction (offline) [31, 66, 70] Adaptive filter (online) [51]	Can theoretically remove artifact without distortion to underlying signal if paired with high-dynamic-range front-end
Component decomposition	Ensemble empirical mode decomposition (offline) [71, 72] Independent component analysis (offline) [71, 73]	Can remove stimulation artifact while providing other information and removing other types of artifact

Adapted from [74].

Table 2.1: Comparison of techniques for stimulation artifact prevention and mitigation.

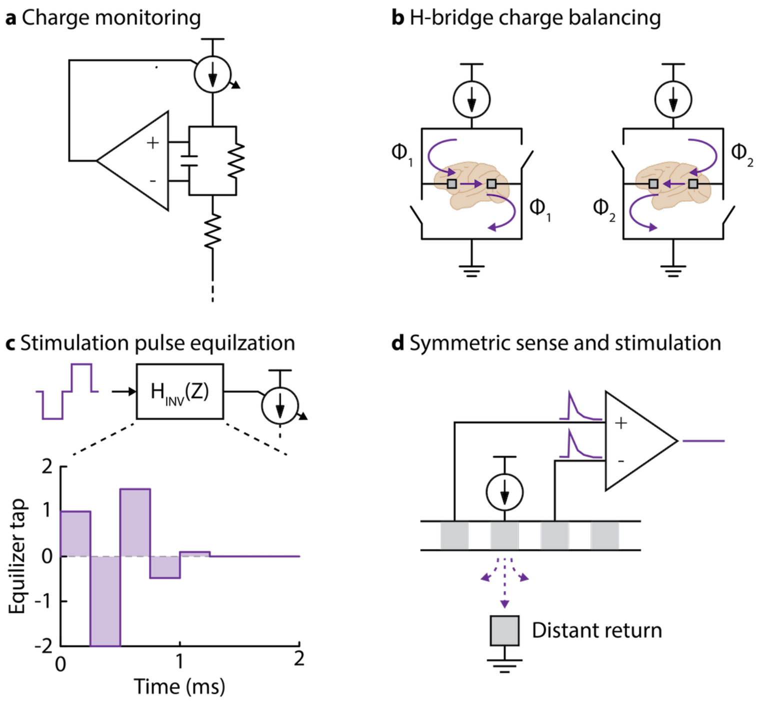

a Charge monitoring
b H-bridge charge balancing
Φ1
Φ1
Φ2
Φ2
c Stimulation pulse equilzation
H_INV(Z)
Equilizer tap
Time (ms)
d Symmetric sense and stimulation
Distant return

Figure 2.9: **Artifact prevention methods. a)** The voltage across the stimulation electrode can be monitored to detect residual charge and apply corrective current pulses [39]. **b)** H-bridge architectures re-use the same current source for both phases of biphasic stimulation, ensuring current matching. Adapted from [42]. **c)** Equalization can be applied to stimulation pulses in order to null out the long residual artifacts from the electrode/tissue "channel". Adapted from [46]. **d)** Symmetric placement of stimulation and recording electrodes creates a common-mode artifact on recording channels, which can be cancelled through differential amplification. Adapted from [11].

which can more easily be rejected by differential recording amplifiers [11]. Peterson et al. [47] eliminated a common ground between the recording and stimulation subsystems, allowing the recording reference to track common-mode artifacts. So far, the presented prevention techniques do not entirely eliminate artifacts, but do ease requirements on front-end acquisition.

Front-end artifact immunity

High fidelity recording of extracellular neural activity requires low-noise instrumentation to detect neural signals down to microvolt amplitudes. Noise performance trades off with power consumption, so recording front-ends, consisting of neural signal amplification and digitization circuitry, typically dominate the overall power of a neural recording system

[75, 76]. As a result, neural recording designs have primarily focused on optimizing power efficiency to extend battery life, reduce wireless power harvesting requirements, or minimize heat dissipation [77–80]. As recording systems scale to higher channel counts (> 1000), power-efficient design is further exacerbated by tight area constraints [49, 81, 82] and high communication data rates [83].

Due to these constraints, most existing neural recording designs have not considered concurrent stimulation and are susceptible to artifacts. For example, power-efficient front-ends typically have a large gain to maximize their sensitivity. A high gain reduces the power requirements of subsequent processing stages, such as an analog-to-digital converter (ADC), but also causes the amplifier to saturate when presented with large stimulation artifacts. Furthermore, conventional front-ends utilize a low frequency high-pass corner to block DC offsets [78–80], but this results in a slow recovery from saturation due to the large, rapid transient voltage of a stimulation artifact (Figure 2.10a). Non-linearity from clipping and long-lasting residual artifacts hide the underlying neural signal.

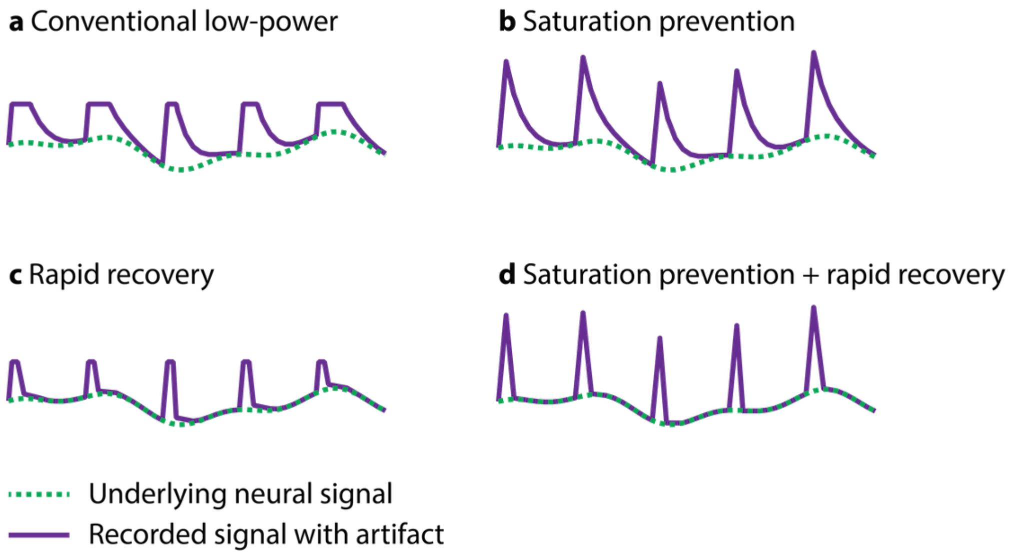


Figure 2.10: **Example artifact waveforms for different front-end artifact resiliency methods.** Recorded analog signals (solid purple) contain artifacts that distort the underlying neural signals (dotted green) to various degrees. **a)** Conventional low-power front-ends have a low dynamic range and large input capacitors which saturate easily and discharge residual artifact slowly. **b)** Saturation of the front-end can be prevented by increasing the dynamic range, retaining linearity of superposition of the artifact and underlying neural signal. **c)** Rapid recovery from saturation and residual artifact reduces the duration of the effect of the artifact, minimizing data loss. **d)** High dynamic range recording and rapid recovery minimizes the amount of recorded and digitized artifact. Adapted from [32].

Newer front-end techniques have focused on mitigating the effects of stimulation artifacts by preventing saturation. One such technique is to increase the dynamic range of the recording circuits, enabling them to withstand and record larger voltages without significantly increasing the electronic noise (Figure 2.10b). With these front-ends, large artifacts are fully captured without saturation and clipping. A direct approach is to decrease the gain in signal amplification, ensuring the amplified artifact can remain in the linear range of successive amplifier stages [48]. This way signal linearity is maintained and stimulation artifact can, in theory, be removed digitally in the back-end. However, this method sacrifices power efficiency by necessitating a higher supply voltage and a higher-resolution ADC for signal digitization. Another approach is to reduce the requirement for a large dynamic range by subtracting, using hardware methods, a model of the artifact at the input during stimulation to keep signals within a smaller range. The model of the artifact is learned through template building [49, 50], filtering the stimulation pulse [51], or creating a replica artifact signal [52]. This technique holds promise, but can degrade the signal-to-noise ratio (SNR) of the underlying neural signal [51]. Alternatively, disconnecting the front-end via a series switch at the input prevents artifacts from reaching the recording circuitry [14, 53–56]. However, this approach can suffer from slow transient settling once reconnected.

While spikes may be detected and analyzed even when superimposed on the long decays following amplifier saturation, these settling responses severely degrade LFP and ECoG signals. Thus, a resilient recording front-end must also rapidly recover from a saturating signal, as shown in Figure 2.10c. Resetting the recording circuits at every sample can clear the saturating charge and eliminate the long transient responses that result. Several designs add hard reset switches [42, 57] or a controllable weak resistance [36, 58, 84] to reset the amplifier and rapidly return to the baseline voltage. In some cases, stimulation causes charge to accumulate on the recording electrode, so some work actively discharges the electrode itself to an averaged pre-stimulus voltage [36–38].

Johnson et al. [42] combines techniques by resetting at every sample while also increasing front-end dynamic range, resulting in both saturation prevention and rapid recovery as shown in Figure 2.10d. The design uses a mixed-signal approach that combines analog and digital domains, and places the ADC inside the amplifier feedback loop. This has several advantages, as it reduces normally-saturating signal swings while providing the same overall gain, allowing for an increased input range of 100s of mV while maintaining power and noise performance. As a result, recorded stimulation artifacts may be more simple to cancel digitally.

Back-end neural signal recovery

Back-end signal processing techniques can be applied to the digitized signals at the ADC output to remove remaining stimulation artifacts. We divide these techniques into three categories: 1) data reconstruction (Figure 2.11, top), 2) artifact subtraction (Figure 2.11, middle), and 3) component decomposition (Figure 2.11, bottom). Although many of these techniques originated as offline methods, some have been implemented online for real-time artifact cancellation. Closed-loop and responsive applications require fast, low-complexity, and low-power online implementations.

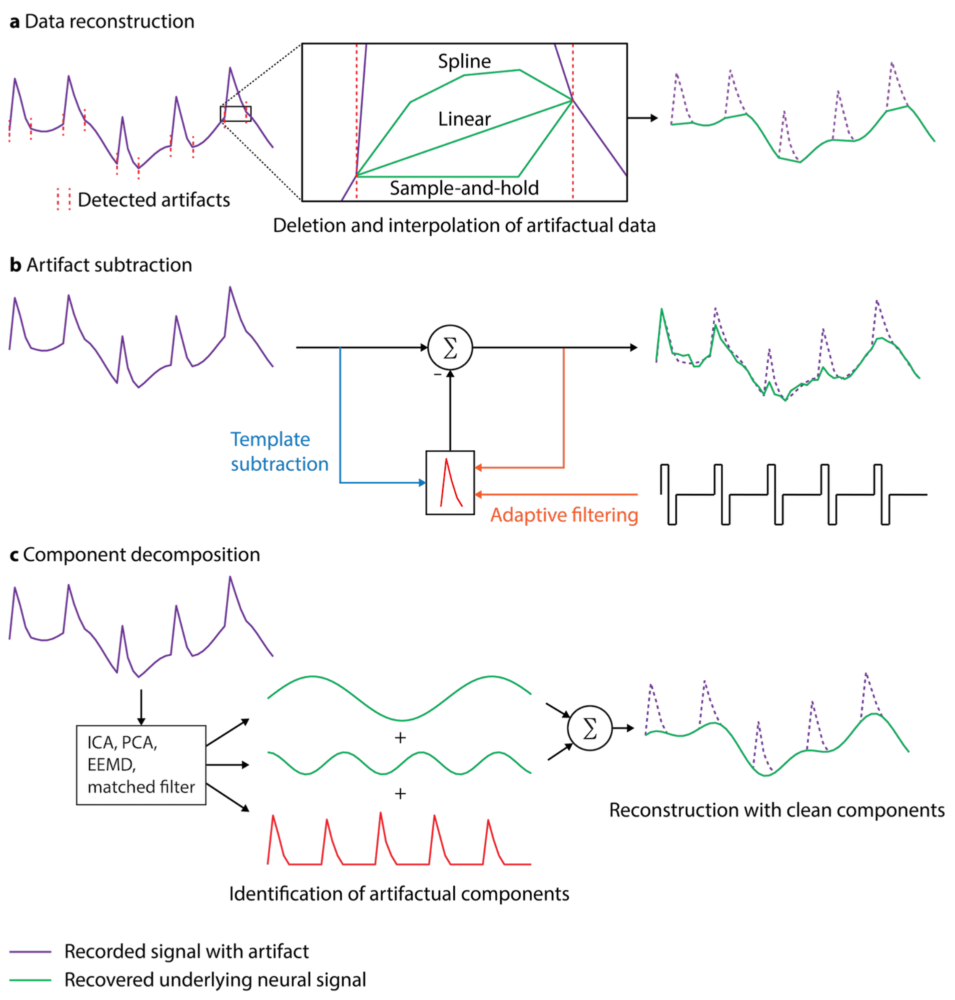

a Data reconstruction
Detected artifacts
Spline
Linear
Sample-and-hold
Deletion and interpolation of artifactual data
b Artifact subtraction
Template subtraction
Adaptive filtering
c Component decomposition
ICA, PCA, EEMD, matched filter
Identification of artifactual components
Reconstruction with clean components
Recorded signal with artifact
Recovered underlying neural signal

Figure 2.11: **Back-end artifact cancellation methods.** These methods attempt to recover the neural signal (green) from the recorded signals with artifacts (purple). **a)** Simplest back-end techniques identify the target segments and reconstruct the underlying data using interpolation. **b)** Subtractive methods remove artifact by subtracting estimated artifact waveforms. Estimation is done through template building or adaptive filtering. **c)** Recorded waveforms can be separated into artifactual and neural components. Clean neural components are used to reconstruct the underlying signal. Adapted from [32].

Reconstruction methods remove samples contaminated with artifacts and replace them with interpolated values (Figure 2.11, top). Sample-and-hold methods hold over the last known good sample for the duration of each artifact [60, 61]. This requires only a single sample of memory, but may cause significant distortion. To reduce distortion, samples may be replaced by linear interpolation between the nearest clean samples [62], an estimation from a learned Gaussian probability density for data segments [63], or a reconstruction using cubic spline interpolation [64]. Hoffman et al. [63] report an SNR of between 30 and 40 dB for linear interpolation and Gaussian estimation. Although simple to implement, data reconstruction requires artifact detection. This can be done using blind detection algorithms [61–63], or using timing indicators from the stimulator [42, 50].

Subtraction and decomposition algorithms assume that the artifact is linearly superimposed onto the neural signal and can be subtracted either sample-by-sample or as an interference source. Theoretically, these algorithms allow for the most accurate recovery of the underlying signal, crucial for spike detection and sorting.

First introduced as an offline method [61, 66, 67], template subtraction has been implemented in both online hardware [65] and software [50, 59] (Figure 2.11, middle). Templates may be formed from averaging artifacts [50, 59, 61, 65–69] or fitting artifacts to a predefined function type [31, 66, 70]. These subtraction techniques suffer from varying artifact morphology resulting from undersampling the artifact shape and misalignment of stimulation and sample timing [68]. To improve template accuracy, more complex methods resample and shift the artifact waveforms and templates [68, 69]. Similarly, adaptive filtering estimates the artifact by filtering the stimulation pulse [51] or the artifact recorded on a neighboring channel [85], then subtracts it while filter coefficients are adapted. These subtraction methods can be implemented with low latency, but require artifact detection, template building and on-board memory for template storage. Templates must be updated often to track any changes to artifact shape or stimulus waveform, otherwise signal distortion may occur. Often estimated templates take time to converge, resulting in varying levels of cancellation over time.

Component decomposition methods separate ensembles of recorded channels into artifact and non-artifact components and reconstruct a clean neural signal with only the non-artifact components (Figure 2.11, bottom). Ensemble empirical mode decomposition [71, 72] and independent component analysis [71, 73] are common approaches to blindly separating artifacts from neural source. While these methods offer great accuracy in reconstruction, they also involve the most intensive computation, requiring iterative processing steps. Therefore, to our knowledge, they have to date not been implemented for online use.

2.2.3 System-level co-design for effective cancellation

It is clear from the above discussion that individual artifact prevention and mitigation techniques often depend on each other to be the most effective. For example, preventative or front-end methods alone are not sufficient for *full* cancellation of stimulation artifact. At the same time, adequate artifact prevention and front-end mitigation methods are required to enable back-end cancellation of sampled artifacts. For example, since subtractive algorithms require linearity of superimposed neural and artifact signals, they must be paired with high dynamic range front-ends to prevent signal loss. Therefore, back-end signal processing and resilient front-ends must be co-designed in closed-loop neuromodulation systems for sufficient artifact cancellation. Culaclii et al. [50] demonstrate a system that implements template subtraction in two stages separated between the front-end and back-end. Front-end subtraction keeps the recorded signal within the linear range of the amplifiers without requiring gain reduction, and back-end subtraction cancels any remaining artifact.

The performance required out of stimulation artifact cancellation methods also depends quite a bit on the target application. For example, back-end data reconstruction methods lose information during the artifact, degrading SNR and potentially also removing action potentials which are typically shorter in duration than the artifact. Thus, interpolation is better suited to lower frequency recordings like LFP and ECoG than spike recordings. Ideally, this technique is paired with rapid recovery front-ends to keep the artifact short. High dynamic range is less critical as saturated data is discarded anyway. High dynamic range front-ends, however, are essential for subtraction and component decomposition techniques, which require the undistorted artifact waveform.

2.3 WAND: Wireless Artifact-Free Neuromodulation Device

Although there have been many successful applications, closed-loop neuromodulation is still very much in its early stages. Progress has often been challenged by the lack of available tools, with existing systems limited by low channel counts, lack of algorithmic flexibility, and stimulation artifact. To enable advanced research in closed-loop neuromodulation, there is a need for a flexible research platform for testing and implementing these various closed-loop paradigms that is also wireless, compact, robust, and safe. Here, I describe work done in collaboration with Prof. Samantha Santacruz, Prof. Benjamin Johnson, Dr. George Alexandrov, Dr. Ali Moin, Fred Burghardt, and Prof. Jose Carmena to build and test WAND [74]. WAND combines: (1) two custom, 64-channel neural interface application specific integrated circuits (ASICs) supporting simultaneous low-noise recording and high-current stimulation, specifically designed to minimize stimulation artifact [42]; (2) flexible and reprogrammable back-end processing on a SoC FPGA for cancelling residual artifacts, computing neural biomarkers, running closed-loop algorithms, and controlling stimulation; and (3) a robust, bidirectional wireless link to a graphical user interface (GUI) for device configuration and control, as well as data logging. These features are tightly integrated into a small form factor, low-power device, enabling many proposed closed-loop and responsive neuromodulation applications, as well as offering a platform for developing new ones.

2.3.1 Application-driven system design

We designed WAND to be a general-purpose tool with immediate applicability in various research environments. Inclusion of a wide feature-set is balanced by limitations in device size and power. A useful closed-loop neuromodulation research platform requires unification of multi-channel recording, biomarker detection, and microstimulation technologies into a single unit with careful consideration of their interactions. Wireless, multi-channel recording-only devices capture activity from wide neuronal populations [86, 87], but do not have the built-in ability to immediately act on that information and deliver stimulation. Several complete closed-loop devices have been proposed and demonstrated, but in addition to their susceptibility to stimulation artifacts, they are also limited by low channel-counts [6, 11, 88, 89] and low wireless streaming bandwidth [6, 11, 88–90]. Most recently, a number of fully-integrated and optimized closed-loop neuromodulation system-on-a-chips (SoCs) have been presented, but full system functionality has not yet been adequately demonstrated *in vivo* [14, 91–96]. While future versions may be paired with miniaturized external battery packs and controllers, current systems built around these SoCs require large, stationary devices to deliver power inductively from a close range [91, 93, 94]. This limits studies to using small, caged animals.

A custom neuromodulation integrated circuit (NMIC) was designed by Prof. Benjamin Johnson and Cortera Neurotechnologies, Inc. to deliver stimulation pulses ranging from subthreshold currents (down to 20 µA) to those required by DBS (5 mA), and to record local field potentials (LFPs) with a bandwidth of up to 500 Hz [42]. We chose LFP as the signal of interest for its usefulness in medical applications as an indicator of disease [97–101]. There is also evidence that LFP can be used for motor decoding in brain-machine interfaces [25, 102–105], with comparable accuracy and better longevity than spike decoders [25]. This signal is also extremely useful to understanding neural processing and incredibly relevant for a variety of basic neuroscience studies, from investigating how neural oscillations coordinate movement [106–108] to cued transitions between dynamic states in cortico-basal ganglia circuits [109] and working memory [110, 111]. Finally, the lower (1 kHz) sampling rate required for LFP recording utilizes lower wireless bandwidth for real-time streaming, allowing the use of low-power, off-the-shelf radios.

Our decision to target LFP recording and DBS-like stimulation settings informed our selection of stimulation artifact mitigation and cancellation methods. Since LFP is a lower bandwidth signal, it is especially important that the slowly-decaying residual artifacts be reduced to a minimum. The NMICs were designed with improved stimulation and recording architectures to actively clear large residual artifacts and minimize their effects on the front-end circuits [42]. This then motivated our implementation of back-end linear interpolation in an on-board SoC FPGA to completely remove stimulation artifacts in real time. In particular, the reduced artifact duration allows for a computationally inexpensive but effective back-end cancellation solution at the cost of losing only one or two samples. Below, we show that such information loss has very little effect on the LFP power spectrum, which is our target neural biomarker. These innovations allow for online, real-time biomarker computation for closed-loop stimulation.

2.3.2 Device architecture

WAND was designed in collaboration with Prof. Benjamin Johnson, Dr. George Alexandrov, Dr. Ali Moin, and Fred Burghardt. WAND components and architecture are shown in Figure

2.12. For experiments with the rhesus macaque, the form factor was designed to fit into the polyetherimide housing for a custom chronically implanted microelectrode array (Gray Matter Research, Bozeman, MT). The device has a board area of 10.13 cm^2 and weight of 17.95 g together with a rechargeable 500 mAh Li-ion battery pack, allowing 11.3 hours of continuous, wireless operation (Figure 2.12a). The main components of WAND are the pair of custom NMICs, an SoC FPGA, a radio SoC, and support circuitry for power regulation and programming (Figure 2.12b, d). Most of these components have also been carried over into an even smaller version of the device, the WANDmini (Figure 2.13), which sacrifices one of the NMICs for a smaller, more conventional footprint.

Each NMIC consists of 64 recording channels and 4 stimulators that can address any of the 64 channels, meaning that stimulation can occur simultaneously on up to 8 channels by leveraging stimulators across the two on-board NMICs. Multi-site stimulation is desirable for implementing specific spatiotemporal patterns of stimulation, with many recent studies performing stimulation on 2 – 8 channels simultaneously [112–117]. Using WAND, the stimulation channels can be dynamically assigned, thus allowing this device to be utilized for multi-site stimulation on up to 8 channels concurrently in a highly flexible manner. Ultimately, stimulation can be delivered using an open-loop paradigm or a closed-loop approach that relies on continuous sensing of on-board computed biomarkers.

Stimulation parameters are rapidly reprogrammable (Figure 2.14) by writing to registers on the NMIC through commands transmitted from a GUI or automatically based on calculations performed on-board. All stimulation settings listed in Table 2.2, as well as selection of stimulation sites and triggering of pulses, can be set through the same interface. A new setting can be preloaded while stimulating with a previous setting, potentially reducing latency between biomarker state detection and the resulting stimulation update.

Unlike conventional neural interface ICs, the NMICs enable simultaneous low-noise, low-power neural recording of LFP with high-compliance electrical stimulation (Figure 2.12c). The NMIC prevents large amplitude residual artifacts by employing stimulators with highly accurate charge balancing [42]. Accurate charge balancing is achieved by reusing the same current source for both phases of a biphasic pulse and a return-to-ground stimulator architecture. To address both direct and residual artifact, the NMIC recording front-ends are designed simultaneously for low noise (1.6 μVrms mean channel noise) recording and a large linear input range of 100mV. The input range is over 10 times larger than conventional designs [76, 78] and avoids saturation in the presence of a large stimulation artifacts (10's of mV) while still being able to resolve μV level neural signals. This large range of resolvable signals is achieved with a mixed-signal architecture that integrates the analog-to-digital converter (ADC) into the feedback loop, thereby reducing the required gain and signal swings. The architecture also resets at every sample, enabling memory-less sampling and rapid recovery from stimulation artifacts [42]. Therefore, stimulation artifacts do not persist beyond the samples when stimulation is occurring, and a minimal amount of data is lost when using reconstructive back-end cancellation methods such as interpolation.

All 128 channels of neural data from both NMICs are sampled, digitized (15 bits, 1 kS/s), and serialized on chip and transmitted to the on-board SoC FPGA with a 166 MHz Advanced

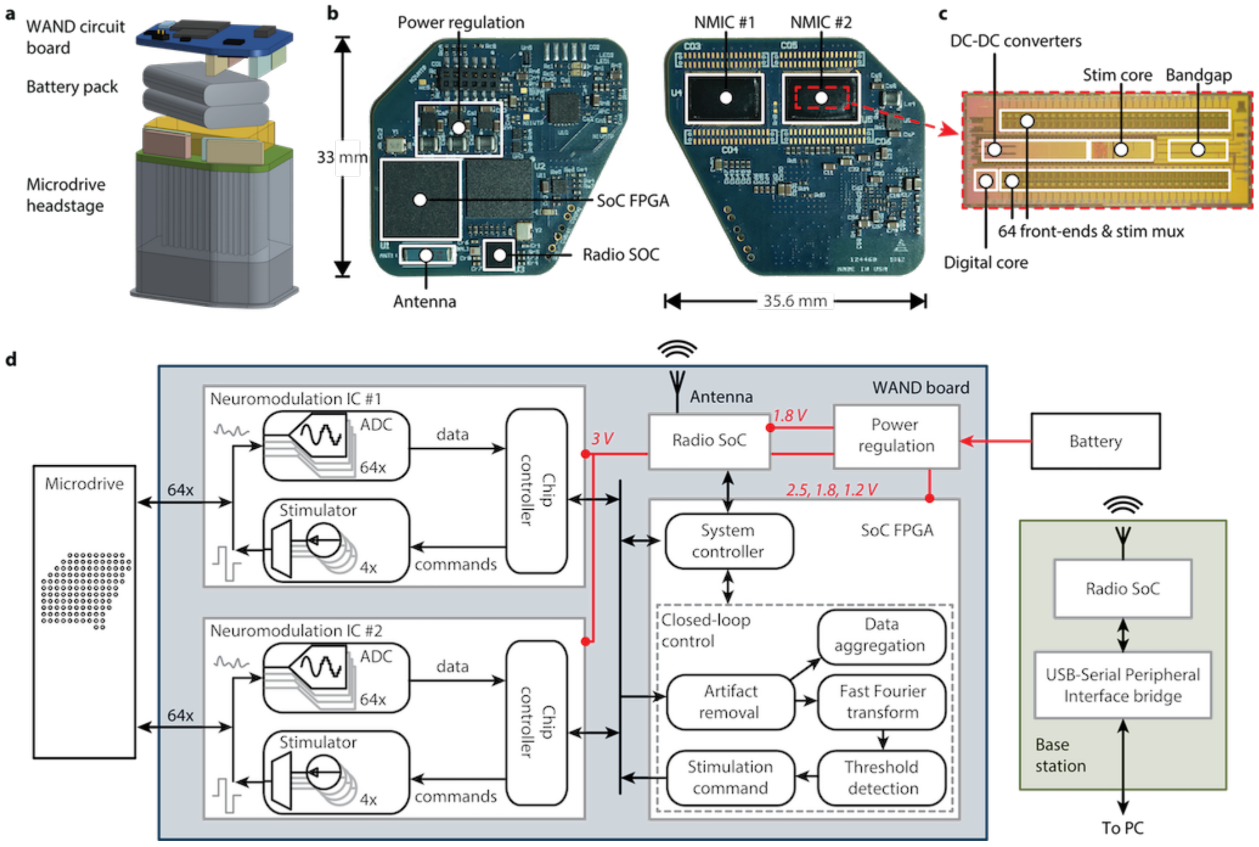


Figure 2.12: **WAND system architecture.** **a)** The 3D computer-aided design (CAD) model of WAND with the primate headstage and battery pack, shown without polyetherimide case. **b)** Top- and bottom-view photographs of the WAND circuit board showing its relevant subsystems and dimensions. **c)** Micrograph of custom neuromodulation integrated circuit (NMIC) with annotated subcircuits: DC-DC converters, stimulation core, bandgap, digital core, 64 front-ends and stimulation multiplexer (stim mux). **d)** Functional diagram of the WAND system showing data and power connections on the main device board, and connections to the microdrive electrode array, battery, and a wireless base station. Adapted from [74].

RISC Machine (ARM) Cortex-M3 processor (SmartFusion2 M2S060T, Microsemi) acting as a master module. Custom software can be developed in the C programming language for a variety of reasearch applications, enabling the use of WAND as a general purpose tool. Additionally, custom design of the FPGA fabric can allow for development of more energy-efficient implementations of algorithms in which higher performance is crucial. The FPGA forms a custom 2 Mbps digital signal and clock interface with each of the NMICs, aggregating data and commands in hardware first-in first-out memories (FIFOs) in both uplink and

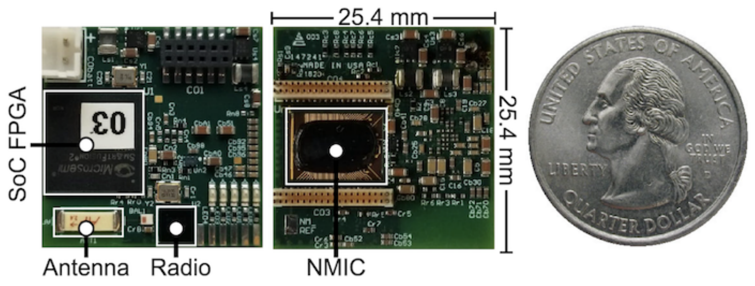


Figure 2.13: **WANDmini system.** Top- and bottom-view photographs of the WANDmini circuit board, with US quarter coin for scale.

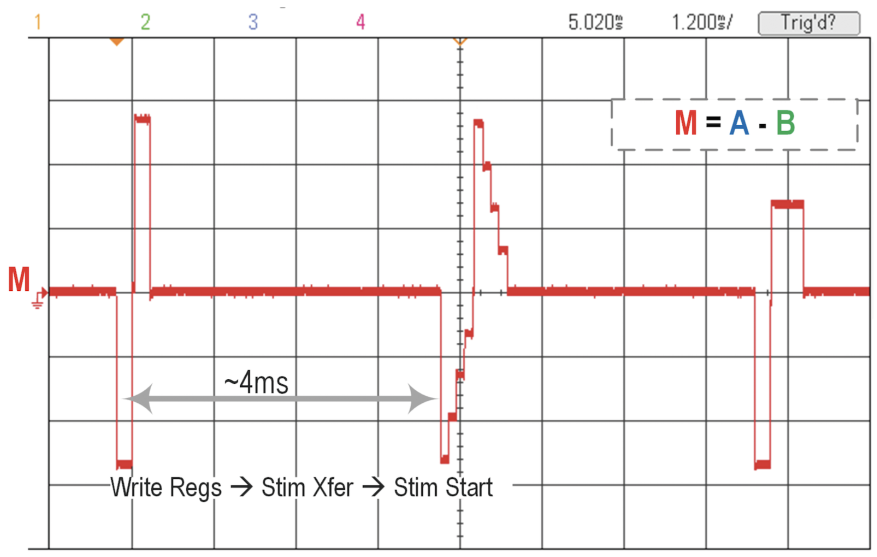


Figure 2.14: **Rapid stimulation setting updates.** Stimulation settings can be updated every pulse at maximum frequency (255 Hz). Additionally, multiple current sources can be multiplexed to a single electrode to create higher amplitude stimulation or complex waveforms, for example. Adapted from [74].

downlink directions [118]. Software running on the included Cortex M3 microprocessor then aggregates neural and other sensor data, cancels stimulation artifact, selects a subset of data to be streamed back to the base station, and runs closed-loop neuromodulation algorithms (Figure 2.12d). The FPGA fabric and Cortex M3 software are reprogrammable through a serial wire debug interface allowing customization for different applications. The SoC FPGA also interfaces with a 2 Mbps 2.4 GHz Bluetooth low-energy radio (nRF51822, Nordic Semiconductor) via serial peripheral interface (SPI) running at 3.08 MHz to form a bidirectional, half-duplex link with the base station and GUI, allowing for communication

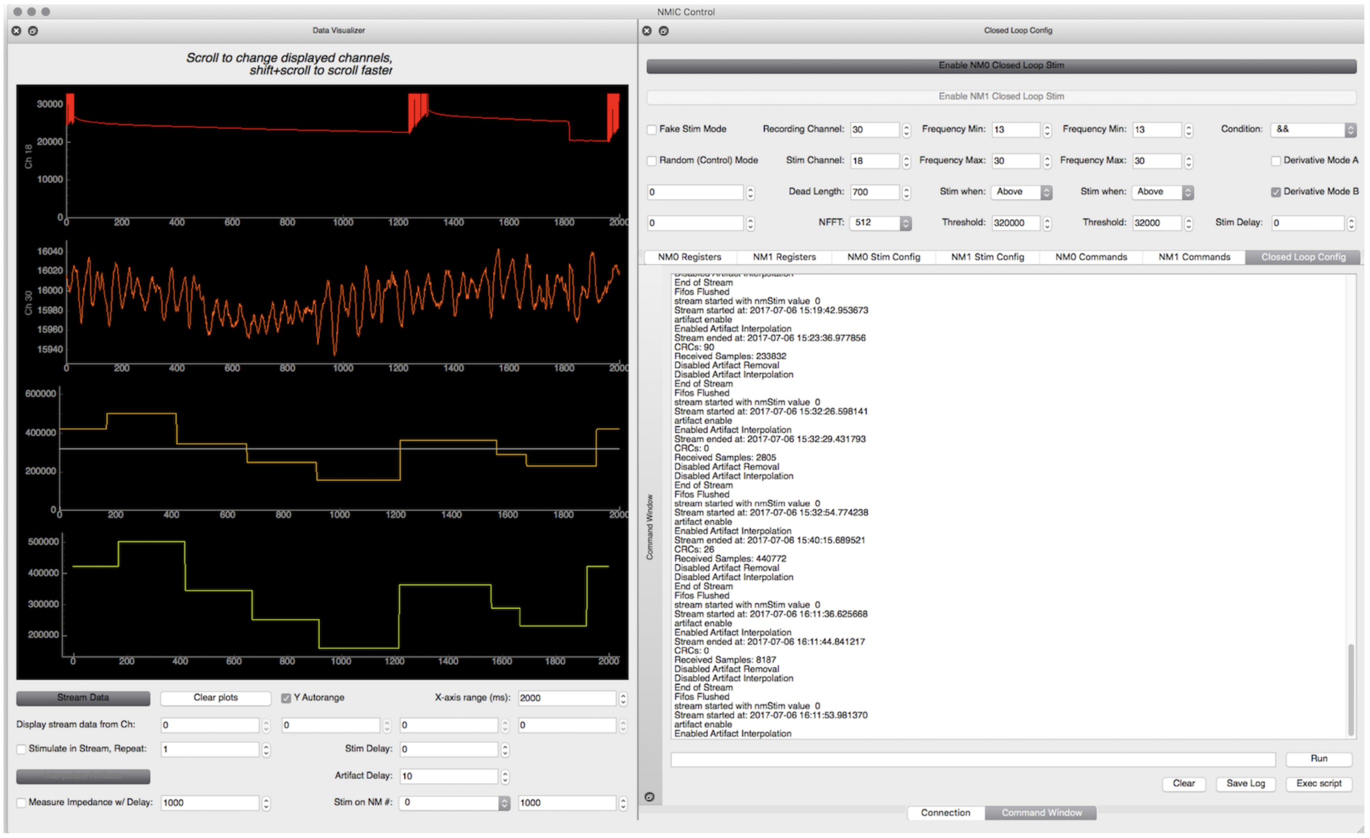
Parameter	Value	Unit
Stimulation subsystem		
Nominal supply voltage	3, 6, 9, or 12	V
Stimulation compliance	11.8	V
Number of stimulation units	4	
Number of addressable channels	66	
Stimulation settings		
Stimulation current resolution	20, 40, 60, or 80	μA
Maximum stimulation current	5.04	mA
Pulse duration resolution	15.625	μs
Maximum pulse width	500	μs
Interphase gap	31.25 - 1000	μs
Short time	31.25 - 1000	μs
Frequency	15-255	Hz

Adapted from [32].

Table 2.2: NMIC stimulator programmability.

up to 2 m from the subject. We developed a custom radio protocol using a time division duplex scheme, allowing low-bitrate commands to be sent from the base-station to the board and high-bitrate neural recordings to be continuously streamed out for logging. The exact division between uplink and downlink can be adjusted to suit the application and streaming state, with a maximum effective bitrate of $\approx$1.6 Mbps. Two streaming modes are available. In open-loop mode, 96 channels of data are streamed to the base station. In closed-loop mode, only the control channel and one of the stimulation channels are streamed, along with the calculated power spectral density.

A wireless base station consisting of a radio (nRF51822 Evaluation Kit, Nordic Semiconductor) and an SPI-to-USB bridge (CP2130EK, Silicon Labs) is used to communicate with WAND (Figure 2.12d). A custom Python GUI was developed to control and monitor data streamed from WAND on a PC (Figure 2.15). Users can setup the system for multiple use cases, visualize real-time neural recordings, configure all NMIC settings, and configure the closed-loop classification algorithm. Recorded data is saved in HDF5 data format along with relevant use case settings, NMIC configurations, and other notes for the recording.



[✓] Y Autorange X-axis range (ms): 2000
Display stream data from Ch: 0 0 0 0
Stimulate in Stream, Repeat: 1 Stim Delay: 0
Artifact Delay: 10
Measure Impedance w/ Delay: 1000 Stim on NM #: 0 1000

Fake Stim Mode Recording Channel: 30 Frequency Min: 13 Frequency Min: 13 Condition: &&
Random (Control) Mode Stim Channel: 18 Frequency Max: 30 Frequency Max: 30 Derivative Mode A
0 Dead Length: 700 Stim when: Above Stim when: Above [✓] Derivative Mode B
0 NFFT: 512 Threshold: 320000 Threshold: 32000 Stim Delay: 0

```
End of Stream
Fifos Flushed
stream started with nmStim value  0
Stream started at: 2017-07-06 15:19:42.953673
artifact enable
Enabled Artifact Interpolation
Stream ended at: 2017-07-06 15:23:36.977856
CRCs: 90
Received Samples: 233832
Disabled Artifact Removal
Disabled Artifact Interpolation
End of Stream
Fifos Flushed
stream started with nmStim value  0
Stream started at: 2017-07-06 15:32:26.598141
artifact enable
Enabled Artifact Interpolation
Stream ended at: 2017-07-06 15:32:29.431793
CRCs: 0
Received Samples: 2805
Disabled Artifact Removal
Disabled Artifact Interpolation
End of Stream
Fifos Flushed
stream started with nmStim value  0
Stream started at: 2017-07-06 15:32:54.774238
artifact enable
Enabled Artifact Interpolation
Stream ended at: 2017-07-06 15:40:15.689521
CRCs: 26
Received Samples: 440772
Disabled Artifact Removal
Disabled Artifact Interpolation
End of Stream
Fifos Flushed
stream started with nmStim value  0
Stream started at: 2017-07-06 16:11:36.625668
artifact enable
Enabled Artifact Interpolation
Stream ended at: 2017-07-06 16:11:44.841217
CRCs: 0
Received Samples: 8187
Disabled Artifact Removal
Disabled Artifact Interpolation
End of Stream
Fifos Flushed
stream started with nmStim value  0
Stream started at: 2017-07-06 16:11:53.981370
artifact enable
Enabled Artifact Interpolation
```

Figure 2.15: **WAND graphical user interface (GUI).** Custom software written for data visualization, device configuration, and running closed-loop experiments. The left side of the window is dedicated to real-time data visualization. From the top, the traces are of one of the stimulation channels, the control channel, and two calculated biomarkers with thresholds shown as a horizontal line. In this case, the two bottom traces appear to show the same information because both the raw value and the time derivative of the same biomarker were used. Full power spectra were calculated on the device and transmitted to the GUI, where local integration of band power was performed before visualization. The right side of the window contains settings for the closed-loop experiment and a command shell for displaying data stream information. Adapted from [74].

2.4 *In vivo* characterization of WAND

To demonstrate the various functions of WAND, we performed a series of *in vivo* experiments with a non-human primate (NHP) that validate long-term, high-fidelity, and wireless multi-channel recording; real-time, complete removal of stimulation artifacts for accurate recovery of neural signals; and on-board biomarker detection for closed-loop stimulation to disrupt movement preparatory activity during a delayed-reach task. These experiments were led by Prof. Samantha Santacruz, were performed in compliance with the NIH Guide for the Care and Use of Laboratory Animals, and were approved by the University of California, Berkeley Institutional Animal Care and Use Committee (protocol AUP-2014-09-6720).

2.4.1 High-fidelity, multi-channel wireless neural recording

To evaluate the quality of recordings made using WAND, we recorded 96 channels of LFP activity from a male rhesus macaque (Macaca mulatta, weight 9.1 kg, age 9 years) using a chronically implanted microdrive electrode array (Gray Matter Research, Inc.; Bozeman, MT, Figure 2.12a) with access to both cortical and subcortical nuclei (Figure 2.16a). The device was head-mounted, and electrical contact with the electrodes was achieved through a PCB, while Omnetics headers were used to connect the PCB to neural recording systems such as WAND or standard tethered electrophysiology equipment. We compared WAND recordings with sequentially recorded neural data from a wired, state-of-the-art, commercial neurophysiology system (Tucker-Davis Technologies [TDT], Alachua, FL). Respective recordings from each system have qualitatively similar signal properties, as assessed by computing the power spectral densities (PSD) of the recorded data (Figure 2.16b, c). The WAND recordings exhibit lower 60 Hz interference due to the lack of long interface cables and better isolated recording references.

To demonstrate robust detection of biomarkers in WAND recordings and establish a baseline for neuromodulation experiments, we recorded LFP activity during a standard self-paced, center-out joystick task (Figure 2.17a, b). For this task, the subject was trained to use a joystick to control a cursor on a computer screen and move to circular targets presented on the screen. The joystick was affixed to the front of the primate chair and the subject was free to use either hand at any point in the task to control the joystick. Each trial began with the subject holding the cursor at a center circular target for 500 ms. Following this hold period, a peripheral target appeared at one of eight target locations equally distributed around the center target at a distance of 10 cm and the center target was removed from the screen, acting as a “go cue”. The subject would then move the cursor (i.e. “reached”) to the peripheral target and held at this target for another 500 ms. If successful, the subject was administered a small juice reward lasting 800 – 1000 ms. A trial was considered successful if the subject completed the two hold periods within a 10 s period.

During this behavior, ongoing beta and high-gamma rhythms are inversely modulated by task-related periods of movement (Figure 2.17c, d). Beta band oscillations are found to emerge during specific motor actions and notably prior to instructed reaches or movements

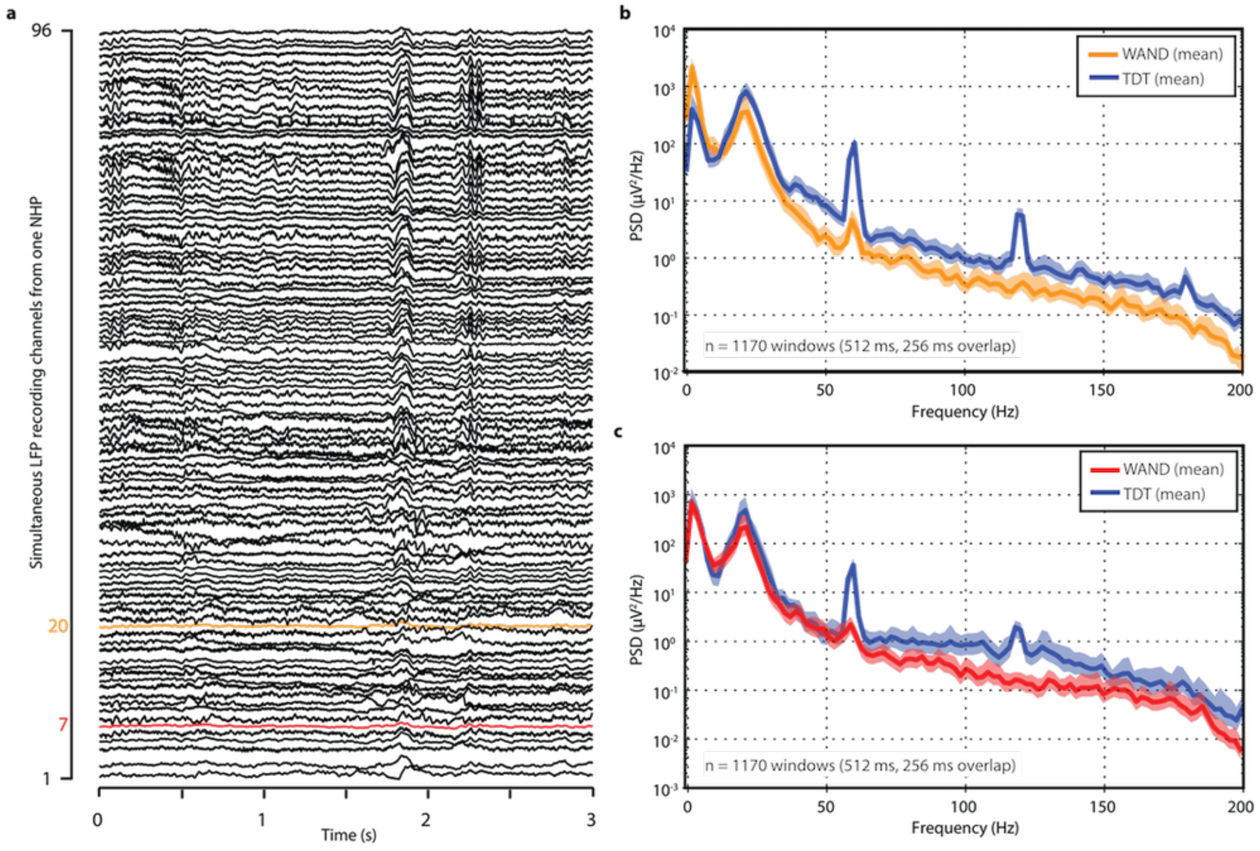


Figure 2.16: **Wireless, multi-channel recording with WAND. a)** Representative 3-second segments of 96 channels of simultaneous LFP recordings taken from one NHP during freely moving behavior. **b, c)** Comparison of mean PSD (Welch's method, one 5 min recording, 512 ms windows, 256 ms overlap) from channel 20 **(b)** and channel 7 **(c)** for recordings taken from WAND and subsequent recordings taken from a commercial wired neurophysiology system (TDT). Shaded area represents the standard deviation (s.d.). Adapted from [74].

[107, 119, 120]. In pre-motor and motor areas, this rhythm has been linked to neural activity related to motor preparation [121–124]. The subject had an average reaction time (RT) of 183.3 $\pm$ 4.8 (standard error of the mean [SEM]) ms across 400 trials. For LFP signals recorded from pre-motor and motor areas, we found that RT was significantly correlated with the average power of beta band activity around the Go Cue (Pearson's correlation: r = 0.12, p = 0.03).

To validate long-term, wireless system functionality, we performed unconstrained, overnight recordings for five nights in the subject's home cage, recharging the battery between each session (Figure 2.18a). Overnight recordings were carried out with the subject moving

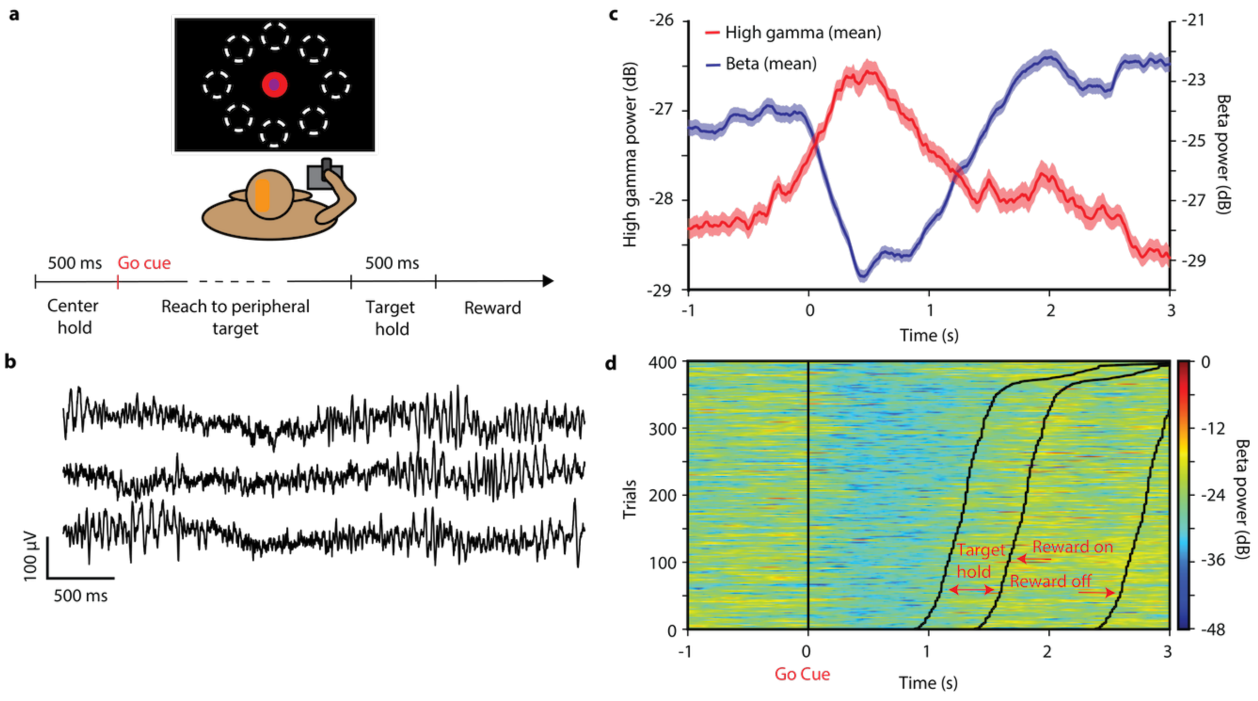


Figure 2.17: **LFP recordings during center-out joystick task.** **a)** Diagram of the center-out joystick task with timeline of task periods for movement and reward. The orange patch on the NHP's head represents the location of the WAND device and headstage implant. **b)** Representative LFP recordings from three channels during the center-out task. **c)** Trial-averaged (n = 400) beta (13 – 22 Hz) and high-gamma (70 – 200 Hz) power aligned to the Go Cue during the center-out task. Shaded area represents the standard error of the mean. **d)** Beta power aligned to the Go Cue. Each row represents activity from a single trial. Trials are organized by the time to Target Hold following the Go Cue. Adapted from [74].

freely throughout the home environment and were typically taken from approximately 8 pm to 6 am. The subject was pair-housed with an NHP cagemate and was in social contact with the cagemate throughout the recording session. It worth noting that the dimensions of the homecage environment could accommodate up to four NHPs, and thus it is feasible to utilize WAND for recording from a small population of socially housed animals without compromising the streaming wireless signal integrity. The base station receiver was mounted on the ceiling approximately 0.5 m from the top of the cage and was connected to a computer running the custom GUI application for acquiring the neural recordings. We recorded on average of 10.2 consecutive hours per night, with a mean packet error rate (PER) below 0.5% and median PER below 0.1%, where each transmitted wireless packet contained 1 ms of neural data from all streamed channels. Offline analysis of the data revealed useful sleep-related biomarkers. Delta (0 – 4 Hz; Figure 2.18b) and theta (4 – 7 Hz; Figure 2.18c) frequency bands are known to have elevated power during sleep states relative to wake states [125, 126]. K-complexes are sleep-specific phasic waveforms that occur spontaneously and are observed through the obtained neural recordings during epochs of increased delta power (Figure 2.18d-g), consistent with classification of sleep state intervals.

2.4.2 Artifact cancellation performance

True simultaneous recording and stimulation is enabled through co-design of the NMIC artifact prevention and mitigation methods with back-end cancellation algorithms running on the on-board microcontroller. To demonstrate WAND's ability to recover neural signals from stimulation artifacts in real-time, we performed experiments delivering open-loop stimulation while recording LFP. There were two types of microelectrodes used in the semichronic array. The first were tungsten electrodes with epoxylite insulation (500 – 800 kOhm; FHC), a standard electrode type for acute neural recording experiments. The second type were platinum-iridium (PtIr) electrodes with parylene-C insulation, which are standard for neuromodulation experiments (200 – 350 kOhm; Microprobes; Alpha Omega). Electrical stimulation in this study was exclusively performed using the PtIr microelectrodes. Stimulation electrodes were chosen to be the same ones for the chosen closed-loop experiment (described in the next section). During the open-loop stimulation experiments, the monkey was in-chair and did not perform any tasks. For each set of stimulation parameters, we recorded three consecutive 30-second segments of LFP: (1) without stimulation, (2) with stimulation turned on but without back-end artifact cancellation, and (3) with artifact cancellation turned on (Figure 2.19). Biphasic stimulation, with amplitudes swept in 40 µA steps between 40 µA and 160 µA, were delivered for pulse widths of 125 µs and 62.5 µs and with 100 Hz and 20 Hz stimulation frequencies.

Artifacts recorded with back-end cancellation disabled were sorted offline into 10-sample windows aligned with the sample 0 being the clean sample before artifact starts and sample 1 being the first flagged sample of the artifact (Figure 2.20b). These segments were used to analyze the size and consistency of recorded artifacts. Offsets were then subtracted from each window such that sample 0 was 0 V. Artifact amplitude was calculated as the average sum of the magnitudes of samples 1 and 2. Artifact duration was then calculated as the average number of samples for which the magnitude was greater than -60 dB of the maximum calculated artifact amplitude.

During the train of identical bipolar, biphasic stimulation pulses, the segment of LFP with uncancelled artifact (Figure 2.19, middle section) demonstrated varying artifact morphology due to the non-integer ratio between the sampling rate and the stimulation frequency (99.8482 Hz shown) (Figure 2.20a). Stimulation pulses occurring completely within a single sample integration window caused only a single sample direct artifact, while pulses occurring at the boundary between two integration windows caused the direct artifact to last two samples. We calculated averaged templates of single- and double-sample artifacts (Figure 2.20b) for varying stimulation amplitudes and pulse widths, and confirmed a linear relationship between these parameters and the artifact amplitude (Figure 2.20c). For all

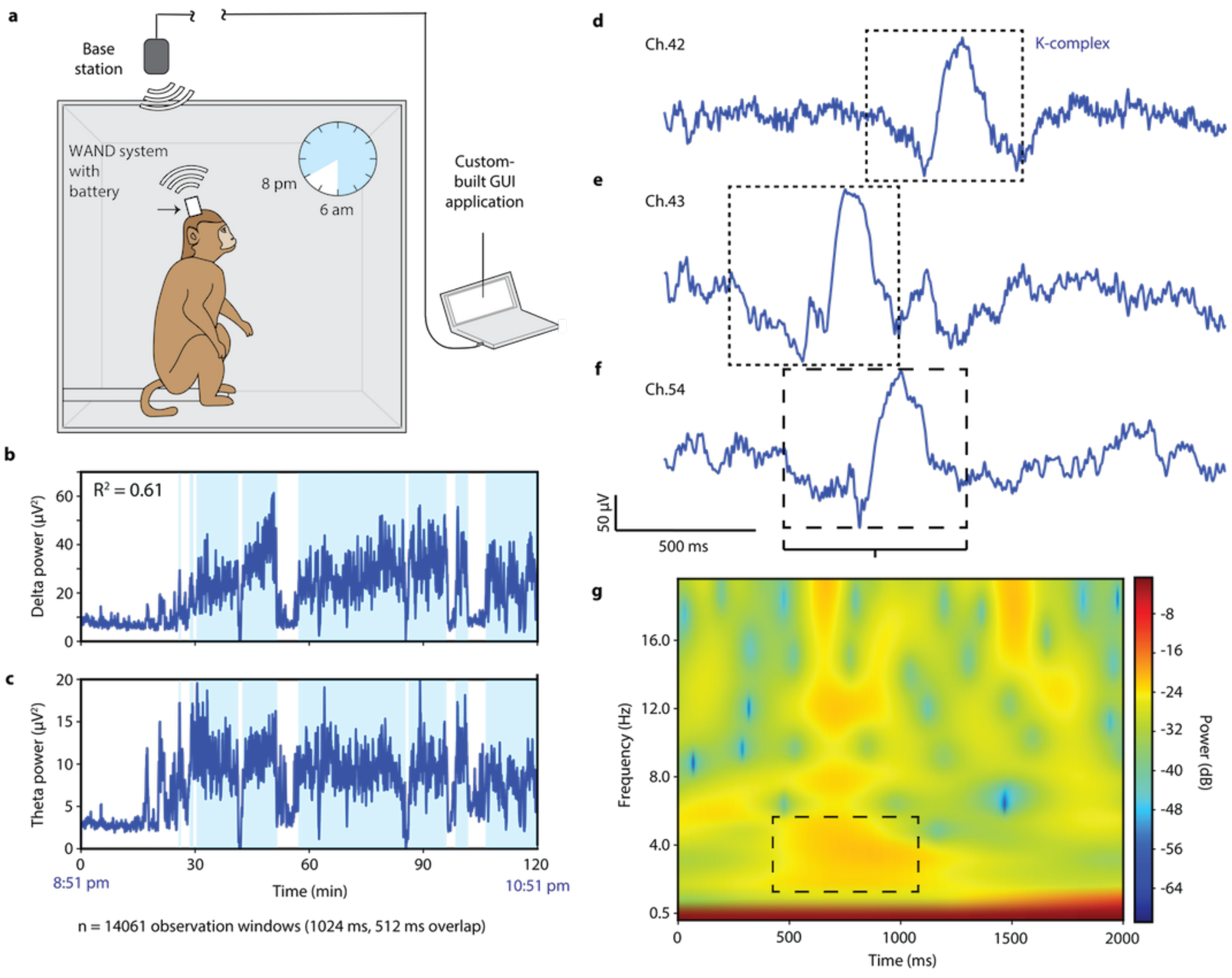


Figure 2.18: **Overnight, untethered recording of brain signals from an NHP during sleep. a)** Schematic diagram of in-cage wireless recordings. **b, c)** Delta (0.5 – 4 Hz) **(b)** and theta (4 – 7 Hz) **(c)** power from two-hour segment of overnight recording beginning at 8:51 pm. The powers are significantly correlated as shown inset in **(b)** (Pearson's correlation: R2 – 0.61, p ¡ 0.001). K-means was used to classify the activity into states of increased (light blue background) and decreased (white background) delta and theta activity (n = 14061 observations). **d, e, f)** Example K-complexes from the caudate **(d, e)** and from the anterior cingulate cortex (ACC) **(f)**. **g)** Spectrogram of activity from channel 54 during the same time window as the waveform shown in **(f)**. Increased delta power occurs coincidently with the K-complex. Adapted from [74].

stimulation parameters within our protocol, recorded direct artifacts on the non-stimulating electrodes remained well within the 100 mV linear input range of the front-end amplifiers,

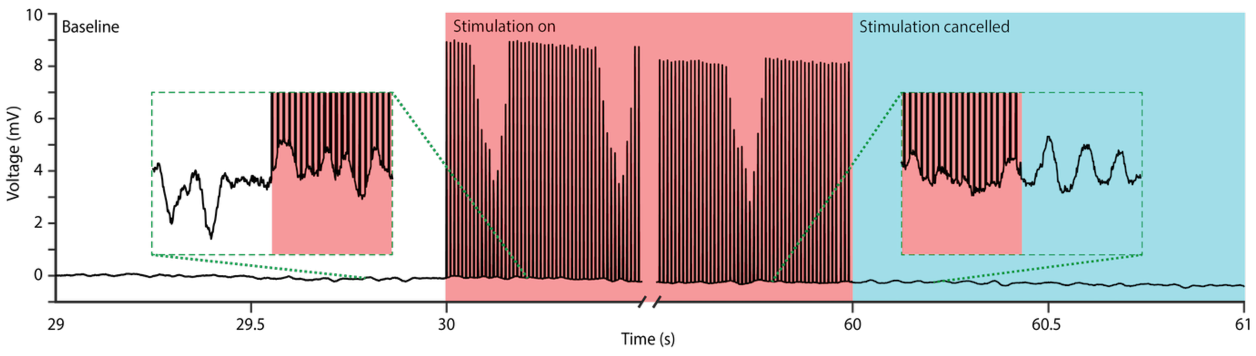


Figure 2.19: **Stimulation artifact analysis and on-board cancellation experiment.** One second segments of raw signals recorded during different epochs of the open-loop stimulation experiment: baseline LFP with no stimulation (white), stimulation with no artifact cancellation (red), and stimulation with artifact cancellation (blue). The inset plots show zoomed 0.4s segments of the LFP during transitions from no stimulation to stimulation without artifact cancellation (left inset), and from stimulation without artifact cancellation to stimulation with artifact cancellation (right inset). In this example, biphasic pulses were delivered at 100 Hz with 160 µA amplitude and 125 μs pulse width per phase. Adapted from [74].

despite the high voltages (10 V) induced on the stimulating electrodes, thus demonstrating saturation-free recording in the presence of stimulation.

Following both single- and double-sample direct artifacts, the residual artifacts were very small and brief, visible only in the averaged templates (Figure 2.20b, inset). Following a single-sample direct artifact, the residual artifact was already suppressed to within -60 dB of the peak amplitude by the following sample, and to within the electronic noise floor of 1.6 µVrms by the second sample. Residual artifacts following double-sample direct artifacts were fully suppressed below the noise floor. These results demonstrate that the recording front-ends rapidly recovered from stimulation pulses, minimizing data distortion.

The residual stimulation artifacts still caused broadband contamination of the recorded spectrum, which we quantified with the ratio, R = 32.78 dB, of signal power integrated from 1 – 200 Hz of LFP during stimulation to baseline LFP (Figure 2.21a, b). Since the recorded artifacts were short in duration (1-2 samples), we chose to implement a method of linear interpolation for artifact cancellation in the back-end [62]. Samples coinciding with stimulation pulses were flagged by the NMIC, ensuring accurate detection of artifacts. Samples were then buffered in the microcontroller, and artifacts were removed by linearly interpolating between the pre-artifact sample and the sample following the maximum possible direct artifact duration (Figure 2.20a).

While more sophisticated techniques may be employed, we found that this simple linear interpolation was sufficient to suppress the artifact power below the neural signal spectrum.

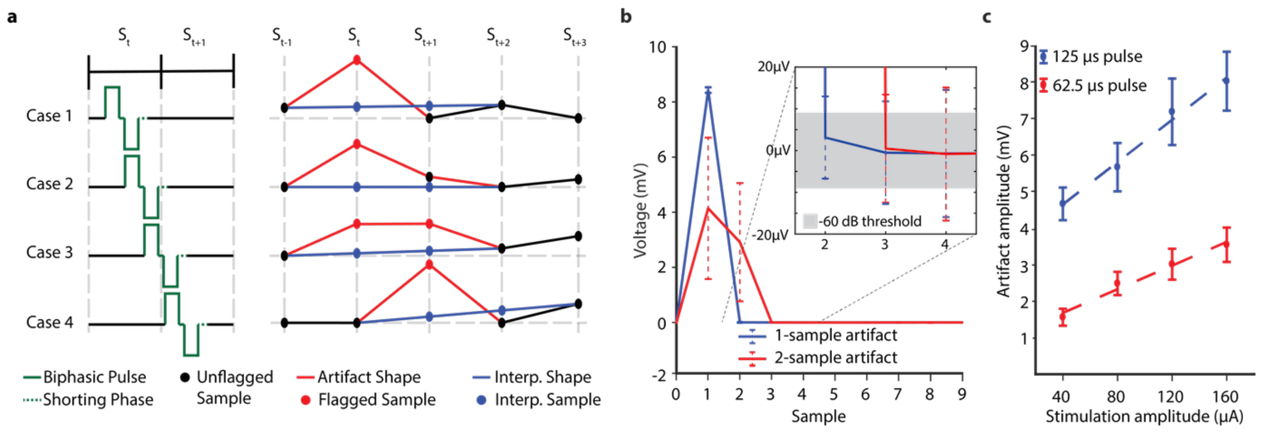


Figure 2.20: **Stimulation artifact morphology analysis. a)** Different cases of relative phase between stimulation pulse and sampling periods for samples S_t at time t (left) with example resulting samples S_t, artifact flags, and cancelled samples using linear interpolation (right). **b)** Averaged templates of single- (blue) and double-sample (red) flagged artifacts. The inset is a zoomed portion showing decay of artifact to within -60 dB of the artifact peak (shaded gray). Error bars are the SD (n = 2106 for single-flag and n = 897 for double-flag). **c)** Average amplitude of artifact from stimulation amplitudes between 40 μA and 160 μA and pulse widths of 125 μs (blue) and 62.5 μs (red). Error bars are the SD (n = 3003 artifacts). Adapted from [74].

Interpolation over two samples at 100 Hz in baseline LFP data without stimulation caused no significant degradation of the spectrum (R = 0.0091 dB). Furthermore, this method did not depend on the actual values of the artifacts and was not affected by the varying artifact morphology, which would have complicated and increased convergence times of template subtraction and adaptive filtering techniques. With on-board artifact cancellation enabled, we were able to recover the baseline LFP spectrum for signals recorded during the simulation pulse train with R = -0.60 dB (Figure 2.21a, b).

2.4.3 Real-time *in vivo* biomarker extraction and closed-loop experiment

To further demonstrate WAND's ability to mitigate stimulation artifacts in real-time, and to perform responsive stimulation using on-board computations, we designed a closed-loop stimulation experiment to disrupt movement preparatory activity during a delayed-reach task (Figure 2.22a). Previous work in macaque monkeys has shown that microstimulation delivered to dorsal premotor (PMd) and primary motor (M1) cortical sites during the delay hold period of a delayed-reach task disrupts preparatory activity and causes an increase in RT [123]. In the study, stimulation was timed synchronous to the task and was not triggered

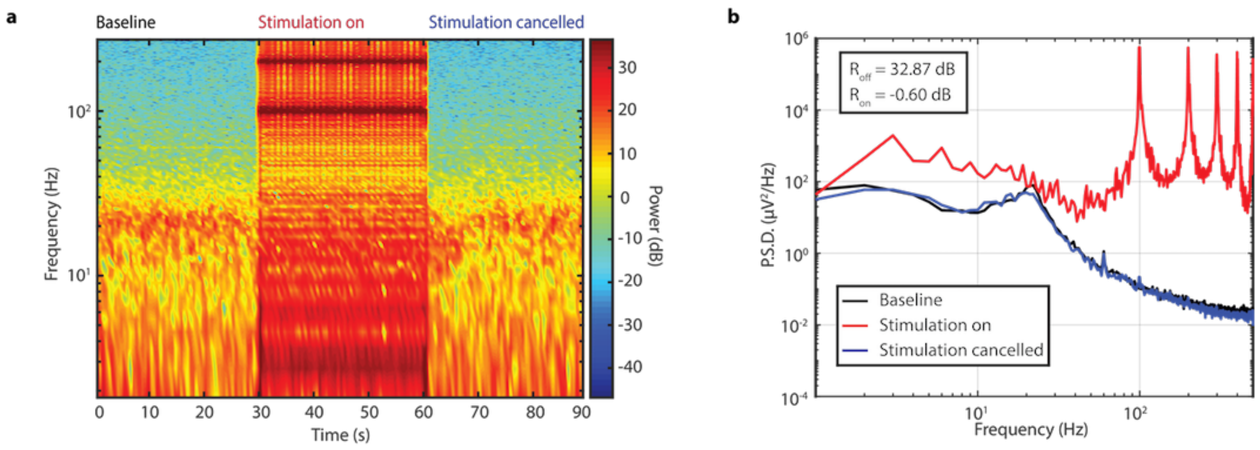


Figure 2.21: **Stimulation artifact cancellation in frequency domain. a)** Spectrogram of full 90-second recording during the different stimulation epochs for the example in Figure 2.19. **b)** Welch power spectral density estimate for each 30-second epoch for the example in Figure 2.19. Adapted from [74].

on recorded neural activity. We reproduced this result by detecting periods of preparation (holding) prior to movement using recorded neural activity in M1 and delivering stimulation to electrodes in PMd in response. In this way, stimulation was automatically controlled by WAND, running in a closed-loop manner relying solely on neural activity and completely separate from the task.

Beta band activity (13 – 30 Hz) is known to reflect movement states, with lower beta band power associated with periods of movement and higher beta band power associated with the absence of movement. Thus, we chose beta band power as the WAND control signal for closed-loop classification of hold periods prior to movement. In this way, closed-loop operation of WAND was completely agnostic to the behavior task states, and stimulation delivery relied solely on the control signal. We heuristically selected a policy of delivering a preconfigured stimulation pulse train when both the beta power and it derivative (calculated over 512-sample windows with 256-sample overlap) exceeded programmed thresholds during the delayed-reach task (Figure 2.22b). Stimulation was enabled when both the beta power exceeded 33 μV-rms and the delta of the beta power exceeded 10.45 μV-rms.

The pulse train parameters were selected to closely match values used in previous work (333 Hz for 57 ms) [123] within WAND specifications. Biphasic, bipolar stimulation was delivered to a pair of PtIr electrodes in PMd. Stimulation pulses were 160 μA in amplitude with 125 μs pulse widths and 31.25 μs shorting phase. Pulse trains were 18 pulses long and delivered at 256 Hz. To avoid stimulation multiple times within the same delay hold period, our policy also incorporated a "dead time" of three calculation periods, or 768 ms. Neural activity was recorded throughout the task, and while stimulation turn-off could be based on neural signature, we chose to adhere to durations used in previous work to demonstrate

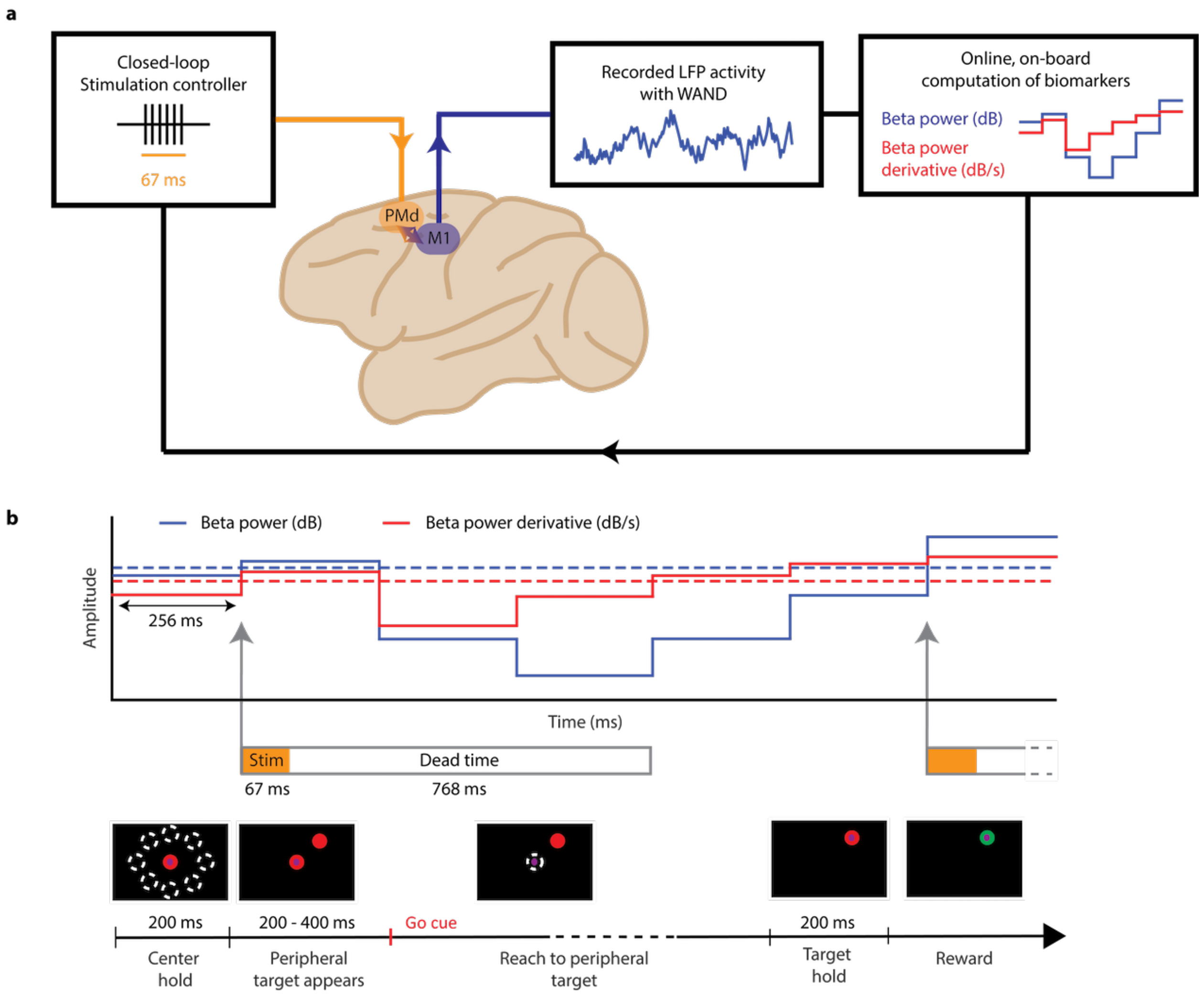

a
Closed-loop Stimulation controller
67 ms
Recorded LFP activity with WAND
Online, on-board computation of biomarkers
Beta power (dB)
Beta power derivative (dB/s)
PMd
M1
b
Beta power (dB)
Beta power derivative (dB/s)
Amplitude
256 ms
Time (ms)
Stim
Dead time
67 ms
768 ms
200 ms
200 - 400 ms
Go cue
200 ms
Center hold
Peripheral target appears
Reach to peripheral target
Target hold
Reward

Figure 2.22: ***In vivo* closed-loop experiment.** The goal was to disrupt movement preparatory activity during a delayed-reach task. **a)** Description of closed-loop paradigm, where recorded activity in M1 is used to control stimulation in dorsal premotor cortex (PMd). **b)** Diagram of the delayed-reach task and the closed-loop algorithm implemented during this task. Stimulation is delivered when beta power and its derivative exceed their thresholds. Adapted from [74].

reproducibility of an established result.

Post-hoc analysis showed that RT increased significantly in trials when stimulation was delivered during the hold period prior to the Go Cue, relative to trials when it was not (Figure 2.23a). The increase of 22.0 ms in average RT, consistent with previous results for microstimulation delivered in PMd, and the change in the distribution of RT (Figure 2.23b)

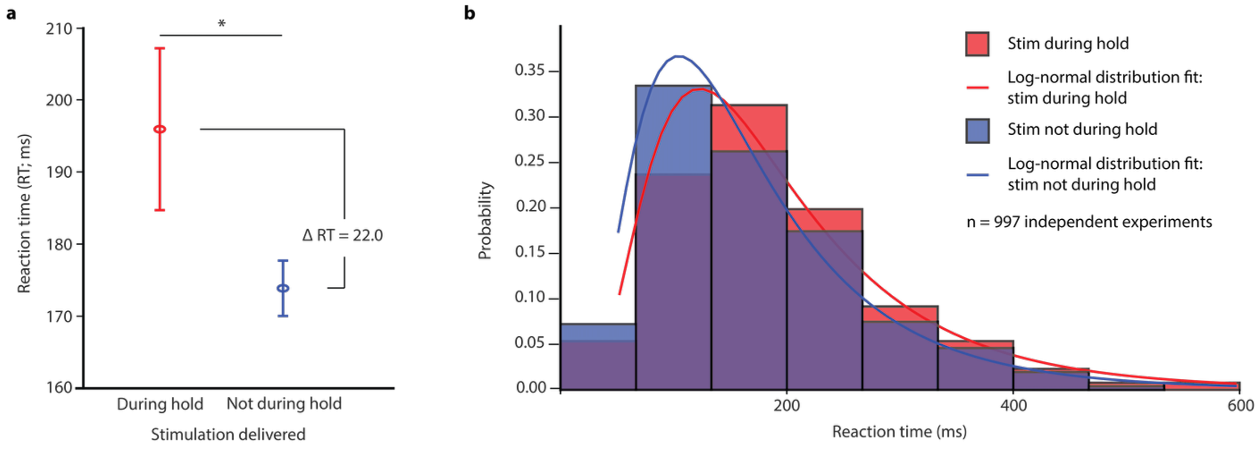


Figure 2.23: **Effect of closed-loop stimulation on NHP reaction time. a)** The mean RTs for trials (n = 997) in which stimulation was delivered successfully during the hold period compared to when it was not. Error bars are the SEM. Significance was determined using a two-sided Mann-Whitney U-test (U = 44193.5, n1 = 131, n2 = 763, *p = 0.003). **b)** Normalized RT histograms and log-normal fit to approximate the respective probability density functions. Adapted from [74].

indicate that neural preparatory activity was successfully disrupted using our closed-loop neuromodulation approach. This functional change in behavior serves as a representative demonstration of how WAND may be used to compute biomarkers in real-time as part of a closed-loop stimulation paradigm and perform online stimulation artifact mitigation.

2.5 Discussion

Closed-loop neuromodulation holds the promise of transforming disease treatment to be performed in an intelligent, patient-specific, and on-demand manner using devices that learn neurological biomarkers and automatically adapt stimulation for optimized therapies. To enable this vision, future devices will have hundreds to thousands (or more) of recording and stimulation channels with embedded signal processing and intelligence. In this chapter, we first reviewed the state-of-the-art for addressing one of the numerous challenges in realizing such devices: the ability to perform recording, stimulation, and computation simultaneously and unhindered by stimulation artifact. Through this review, we conclude that an optimized system must combine preventative, front-end, and back-end techniques, which, in combination, we believe are key to obtaining the best results in recording the full underlying neural signal during stimulation and thereby enabling true closed-loop neuromodulation.

We then shared our approach to this problem in demonstrating WAND, a small form factor, wireless neuromodulation device enabling simultaneous recording and stimulation for a variety of research purposes. Artifact cancellation is made possible due to the full-system co-design. Using custom NMIC recording front-ends [42] with wide dynamic ranges and fast recovery from artifacts, as well as accurate timing information indicating which samples are affected by both direct and residual artifacts, we can design simple, yet effective cancellation algorithms in the digital back-end using linear interpolation. In all, integration of a custom ASIC with an on-board FGPA and radio enables high-quality, long-term multi-channel recording and stimulation during free behavior, full cancellation of stimulation artifacts, flexible programmability, and low-latency processing for delivery of closed-loop microstimulation based on detected biomarkers.

Table 1 summarizes WAND system specifications and compares them to other state-of-the-art closed-loop neural interfaces that have been fully packaged and validated in vivo at the time of original publication. We view WAND as a deployable research tool ready for use with large animals and potentially humans, and we therefore limit our comparisons to similar devices that can operate fully wirelessly and autonomously to make immediate impact on scientific and clinical discovery. Our criteria include fully on-board recording, computation, and stimulation ability; wireless data streaming or on-board memory for data storage; and an implantable or wearable power source. The Neurochip-2 [88], PennBMBI [89], NeuroPace RNS [6], and Activa PC + S [11, 127, 128] devices allow for flexible biomarker detection and triggered stimulation, but with a low number of channels. A device developed at the University of Toronto [44, 90, 129] enables recording and stimulation on more channels, but all of these devices still stream data at a low rate, preventing large-scale multi-site analysis. WAND improves upon these limitations as the only device to our knowledge that incorporates a large number of recording and stimulation channels, have a wireless data rate to support a large number of streaming channels, and also provide closed-loop neuromodulation capabilities. It is also, to our knowledge, the only neural interface system that actively cancels stimulation artifacts through both hardware and software techniques for completely artifact-free recording during stimulation. Although the Neurochip-2 and Activa PC+S devices utilize some hardware and experimental setup techniques, large residual artifacts still appear to effect performance during stimulation [11, 88].

In addition to the systems described in Table 2.3, a number of new, comparable systems have emerged since our initial demonstration of WAND that have made further progress the in field of closed-loop neuromodulation. A wide range of system-on-chip integrated circuits have been demonstrated with concurrent, full-fledged recording, stimulation, and biosignal processing and classification capabilities [132, 133]. Medtronic has also since received FDA approval for its new Percept PC DBS device with "BrainSense" technology, allowing it to simultaneously record and deliver DBS using specially designed leads [7]. With the Percept PC, clinicians will be able to monitor and make better judgements on the quality of individualized DBS treatment.

The experiments shown here are intended to outline and demonstrate the capabilities enabled by WAND, paving a path toward use of this technology as a tool in clinical and neuroscientific research. Even in it's current configuration running the presented closed-loop algorithm, it has the potential to already be applicable in human research. For example, in

	Neurochip-2 [88]	**PennBMBI** [89]	**U. of Toronto** [44, 90, 129]	**NeuroPace RNS** [6, 130]	**Activa PC + S** [11, 127, 128, 131]	**WAND**
Dimensions	$63 \times 63 \times 30$ mm^3	$56 \times 36 \times 13$ mm^3 recording and processing, $43 \times 27 \times 8$ mm^3 stimulator	$22 \times 30 \times 15$ mm^3	$28 \times 60 \times 7.7$ mm^3	39 cm^3	$36 \times 33 \times 15$ mm^3
Weight	36 g boards, 145 g total	–	12 g total	16 g	67 g total	7.4 g board, 17.95 g total
Power	284-420 mW	290 mW	45 mW	–	–	172 mW
Wireless link	IR	Nordic Enhanced Shockburst	ZigBee	20-50 kHz short range inductive	175 kHz ISM	Nordic BLE
Data rate	–	2 Mbps	250 kbps	–	11.7 kbps	1.96 Mbps
Real-time streaming	No	4 channels	1 channel	1 channel	2 channels raw, 4 channels compressed	96 channels + 3 accelerometer channels
# Recording channels	3	4	256	4	4	128
Recording power/channel	–	1.25 mW/channel	52 μW/channel	–	5 μW/channel	8 μW/channel
Sampling rate	2/24 kS/s (24 kS/s only on 1 channel)	21 kS/s	15 kS/s	250 S/s	422 S/s	1 kS/s
ADC resolution	8 bits	12 bits	8 bits	10 bits	10 bits	15 bits
Artifact mitigation	Transient gain reduction	None	None	Low pass filter	Front-end filtering, heterodyning, symmetric sensing	Dynamic range increase, memoryless sampling
# Stimulation channels	3	2	64	8	8	128
Maximum current	5 mA	1 mA	250 μA	11.5 mA	25.5 mA	5 mA
Compliance voltage	$\pm$15 (50) V	$\pm$12 V	2.6 V	12 V	$\pm$10 V	12 V
Charge balancing	Matching resistors (0.1 %)	0.75% mismatch	–	$< 10\mu$C/sec charge imbalnce	Passive discharge	Biphasic current source reuse, 0.016% mismatch
Artifact cancellation	No	No	No	No	Stimulation as feature for SVM	Linear interpolation
Biomarker detection	Spectral power, action potential detection	Spectral power, time domain features, action potential detection	Phase locking value (PLV) seizure precursor	ECoG signal intensity, line length, half-wave	Spectral power	Spectral power
Closed-loop control	Detection/threshold triggered	Detection/threshold/sensor triggered	Thresholding	Detection/thresholding	2D SVM	Thresholding
Animal model	Primate	Rodent	Rodent	Human	Ovine	Primate
***In vivo* closed-loop paradigm**	Spike triggered	Sensor-node event (button press) triggered	PLV threshold triggered	Seizure detection and responsive treatment	SVM-classification (spectral power, stimulation) triggered	Spectral power threshold triggered

Adapted from [32].

Table 2.3: Comparison of closed-loop neuromodulation systems with full *in vivo* validation.

Parkinson's Disease patients treated with DBS, a spectral peak biomarker linked to dyskinesia, an adverse effect of DBS therapy, has been discovered [97]. The closed-loop paradigm we demonstrate in this work can be easily modified for this application, as our current biomarker detection algorithm can already be used to sense this spectral peak. Modifying the control policy to reduce stimulation amplitude rather than trigger stimulation when the peak is detected can thereby reduce the incidence of this undesirable side effect.

Additionally, the WAND platform, in its various form factors, has already been applied to other biosignal recording and classification applications. WANDmini has been used in a non-invasive neural interface for recording in-ear electroencephalography (EEG) [1]. The system has also been adapted for recording surface electromyography (EMG) for classification of muscle activity in hand gesture recognition [134, 135]. In the next chapter, I will describe how we have leveraged the in-sensor compute capabilities to implement a learning algorithm complete with on-board model training and updates for gesture recognition in a wearable form factor.

Chapter 3

Real-Time, In-Sensor Gesture Recognition

Hand gesture recognition based on recorded muscle activity has many applications in human-computer interaction and can potentially also be of use in controlling active prosthetics for rehabilitation. A practical, wearable device for performing gesture recognition should be energy efficient, light-weight, and unobtrusive, all while providing good classification accuracy using adaptable online machine learning models. In this chapter, I first cover the basics of hand gesture recognition using surface electromyography, as well as some challenges affecting the robustness of classification algorithms. I then introduce the FlexEMG system, which combines: (1) high channel count, high density electromyography recording using a flexible, screen-printed electrode array with (2) in-sensor signal processing and machine learning for real-time gesture recognition and incremental updates to the classification model. My collaborators and I utilize a brain-inspired hyperdimensional (HD) computing classification model, which allows us to implement the entire flow of model training, inference, and updates all in the on-board hardware. We demonstrate real-time classification performance in a series of gesture recognition tasks within various situational contexts.

3.1 Introduction

Hand gestures are an integral part of human-to-human communication and, as such, have the potential to be leveraged for more natural human-computer interaction (HCI), providing a fast and effective physical medium for communicating with and controlling intelligent devices. Accurate and efficient gesture recognition enables intuitive control for applications ranging from controlling poly-articulated robotic hands [136] to interacting with mobile devices and game interfaces [137].

Various methods exist for detecting or inferring hand gestures, as shown in Figure 3.1. A common method employed in many commercial devices (such as the Microsoft Kinect or the Leap Motion Controller [138]) is based on vision and depth sensing using external

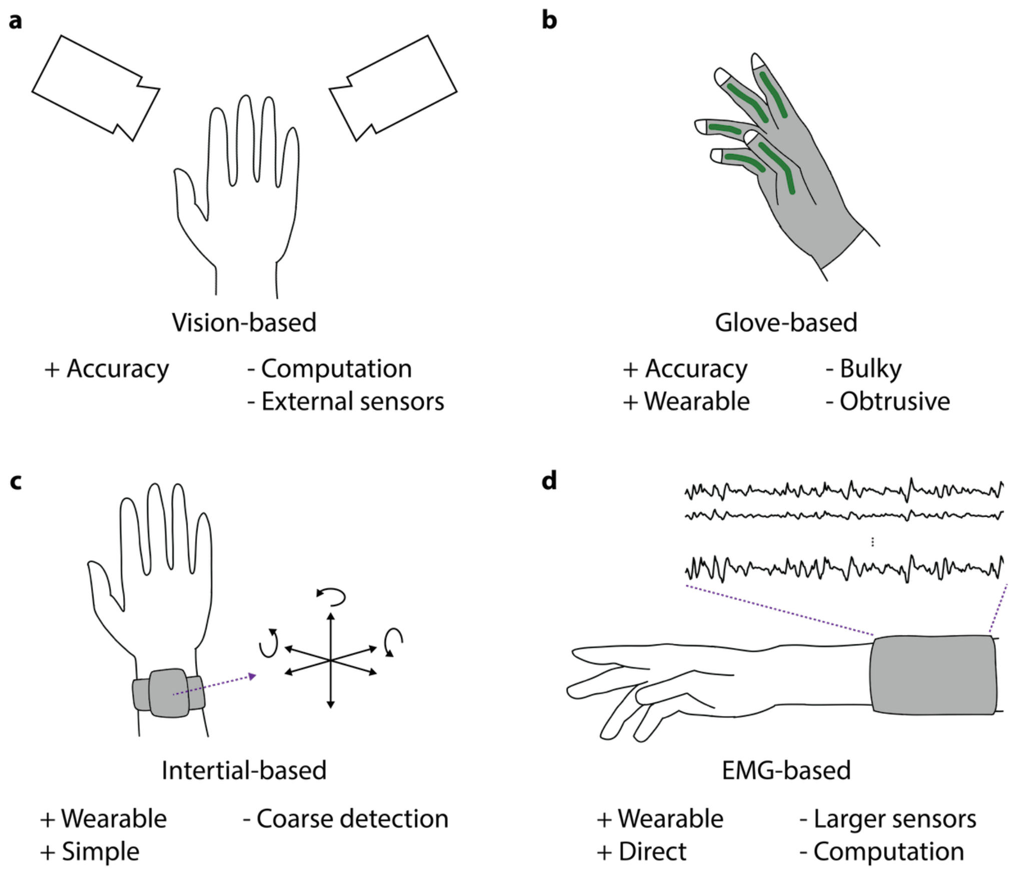


Figure 3.1: **Various modalities for hand gesture recognition.** **a)** Vision-based methods rely on external sensors to reconstruct a user's position and movements. **b)** Glove-based methods require a user to wear sensors directly on their digits to sense motion in the joints. **c)** Inertial-based methods sense coarse movements through small, wearable sensors. **d)** Electromyography (EMG)-based methods sense muscle activity and the users intentions.

sensors (Figure 3.1a). Using LIDAR, high quality reconstructions of a user's hand, arm, or additional body parts can be generated to then infer the gesture being performed. While highly accurate, these vision-based methods rely on heavy computation and external sensors which make them inappropriate for wearable applications. "Cyber-glove" systems can be worn on the hand and fingers to directly detect their motions [139, 140], typically using resistive force or strain sensors to measure joint angle (Figure 3.1b). These systems offer accurate gesture recognition in a wearable form-factor, but can be bulky and obtrusive from covering the entire hand. Inertial sensors can also be used to classify gestures based on motion sensing using a combination of accelerometers and gyroscopes (Figure 3.1c) [141]. Although the sensors are very small, they are typically worn on the wrist, and thus cannot sense finer digit motions. Additionally, inertial sensors are typically better suited for classifying dynamic gestures and movements rather than static postures. This makes inertial sensing a good *auxiliary* input for hand gesture recognition, and it is often incorporated in addition to other gesture sensing modalities [3, 141].

In comparison to the other listed methodologies for gesture recognition, methods based on electromyography (EMG) (Figure 3.1d) may offer a good balance of accuracy, wearability, and unobtrusiveness. Hand gestures and movements are produced through contraction and relaxation of various groups of muscles in the hand and forearm. These muscles are composed of groups of muscle fibers which are innervated and activated by motor neurons, which in turn are innervated by other neurons in a person's nervous system. The basic unit of this signal chain is called the motor unit, which consists of a single motor neuron and the specific group of muscle fibers it excites (Figure 3.2) [142]. The target signal of EMG is the motor unit action potential (MUAP), which is a summation of the motor neuron action potential and the responses of the innervated muscle fibers. It is possible to use penetrating electrodes implanted intramuscularly to record individual MUAPs, much like single-unit spike recording from the central nervous system as described in Chapter 2.

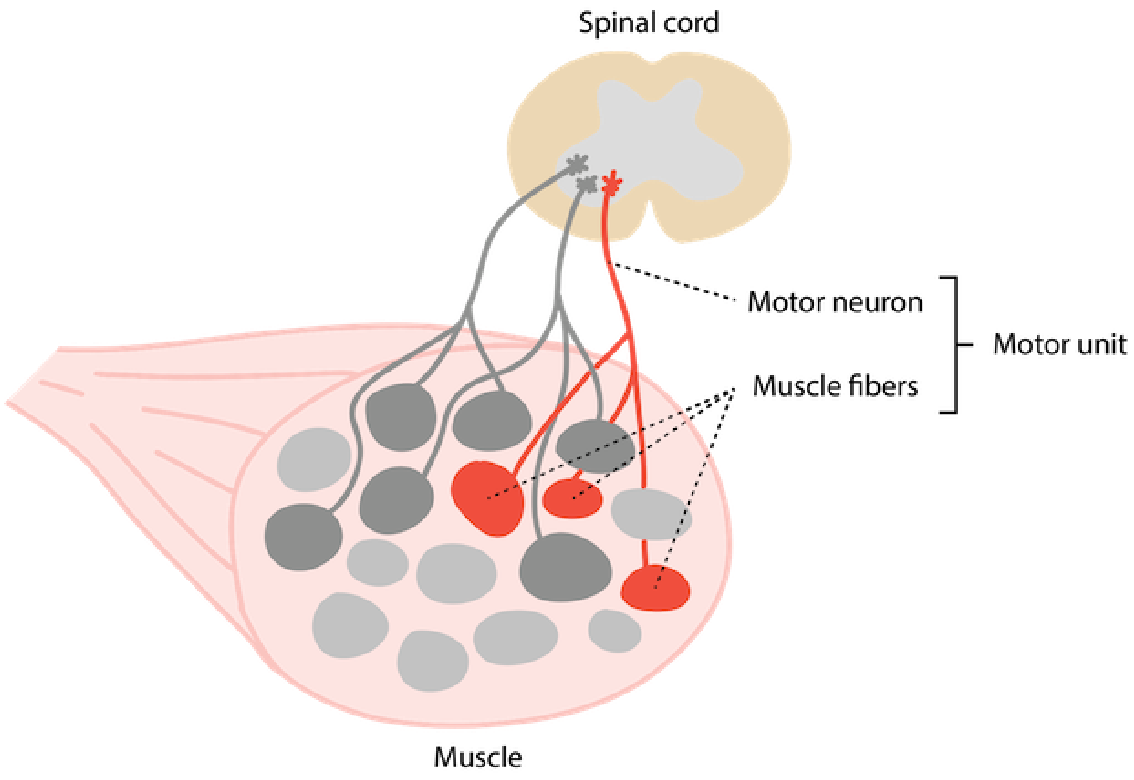


Figure 3.2: **Basic motor unit of neuromuscular system.** A lower motor neuron originating in the spinal cord innervates various muscle fibers within a muscle and electrochemically excites them.

Hand gestures may be recognized based on the measured activity of large populations of these motor units. In fact, unlike the methods described above which measure the gestures and the resulting motion, EMG-based gesture recognition more directly measures the *cause* of gestures and potentially even the intent behind them. The ability to accurately classify hand gestures and movements using electromyography depends heavily on the instrumentation used to record EMG signals, as well as the algorithms for decoding or recognizing patterns within the activity.

3.1.1 Electromyography systems and devices

The most basic, but also the most accurate, way to measure electromyography is to use intramuscular electrodes connected to an amplifier for recording of individual MUAPs (Figure 3.3). There are many drawbacks to this, including a high level of invasiveness and requiring many transcutaneous wires to route from the electrodes to external acquisition equipment. One situation in which this is not an issue is when EMG recording is added as a subsystem part of an already highly invasive procedure: intramuscular EMG can be used to allow patient control of bone-implanted osseointegrated prosthetics [143]. In these systems, leads to intramuscular electrodes can be routed through channels already part of a larger implanted system.

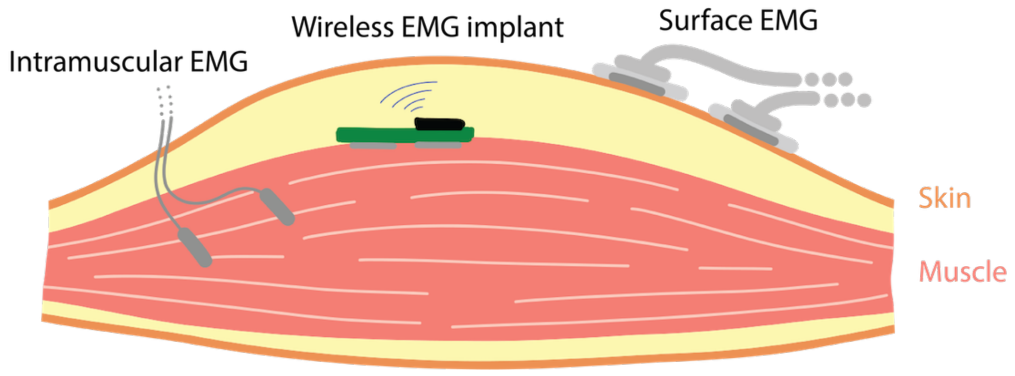


Figure 3.3: **EMG electrode recording sites with various levels of invasiveness.**

Alternatively, intramuscular EMG recording can be improved through the use of wireless interfaces, precluding the need to route signals through tissue and skin [144–146]. Wireless power and communication in these subdural implants improve stability for chronic use with lower risk of infection, but the invasiveness of implantation is still a barrier to widespread adoption.

Surface EMG is a non-invasive method of acquiring motor unit signals by placing recording electrodes on the surface of the skin rather than inside or on the surface of the muscles (Figure 3.3). This enables EMG recording to be used in *wearable* devices rather than implants, opening it up to many new applications. Traditionally, EMG is recorded using wet electrodes with an electrolytic gel for improved impedances. In laboratory settings, these electrodes are connected to bench-top biosignal acquisition units which enable multi-channel recordings from multiple electrode sites.

When limited to just a few electrodes, a lot of care must be taken for optimal electrode placement [147]. A few shortcomings of surface EMG, as compared to implanted or intramuscular EMG, arise because measurements of the electrical activity are made through skin and tissue. This leads to a great deal of attenuation of the signal, reducing MUAP amplitudes from millivolts down to 10's of microvolts or less. Electrodes must be placed directly over muscles to maximum amplitude signals, and recordings are limited to those superficial muscle fibers. Signals from deeper targets are blocked and attenuated far more. Furthermore, the distance makes it impossible to single out individual motor units for recording. A single electrode will instead measure the superimposed activity of a large number of motor units in its vicinity, including those from multiple different muscles. This crosstalk makes it difficult to directly correlate the signal from a single electrode with the activity of a single muscle or group of motor units.

Thus, slight changes in electrode positioning can greatly affect both the signal quality and the targeted structures of surface EMG recordings. Recently, high-density surface EMG has been proposed to alleviate the problem (Figure 3.4a). Instead of deliberately and precisely placing a few electrodes, a large grid of densely-packed electrodes (> 50) can be positioned over a general area on the skin. While typical individually placed electrodes can measure over 25 mm in diameter, dense electrode arrays can have recording sites every few millimeters. These sensors require custom-made electrodes acquisition setups to provide tight enough grouping and handle recording the high bandwidth signal from all of the channels [148, 149]. Since all of the signals are simultaneously captured, it is now up to the researcher (or back-end algorithm) to decide which channels contain important information for the task. Channels can be dropped using feature selection techniques, but now the burden of precise placement is no longer on the wearer properly donning the device. Furthermore, useful information can be extracted from the higher channel count, enabling detection of new muscle characteristics [150, 151], more accurate or robust algorithms [149], as well as the possibility of deconvolving and separating out activity from individual motor units [152–155].

While these high-density EMG recording systems may enable easier and more plentiful access to desired signals, they are far from wearable and cannot be used in most everyday activities and applications. A few examples of more wearable EMG recording devices have emerged in the past few years (Figure 3.4). These include the Myo armband [3], which has been very popular in the research community, as well as the CTRL-kit [4] from CTRL-Labs (now part of Facebook Reality Labs). These devices feature a fewer number of recording channels spaced evenly around a wrist-/forearm-worn armband, as well as an accelerometer for measuring movement and posture. These signals are captured and digitized by on-board electronics which transmit them over Bluetooth in real-time for logging or control of applications. Ideally, a future wearable EMG device for everyday use merges the best aspects of all of these devices: high-quality recording on a large number of channels; wearable and wireless form factor; and, as we will discuss later, on-sensor processing and learning

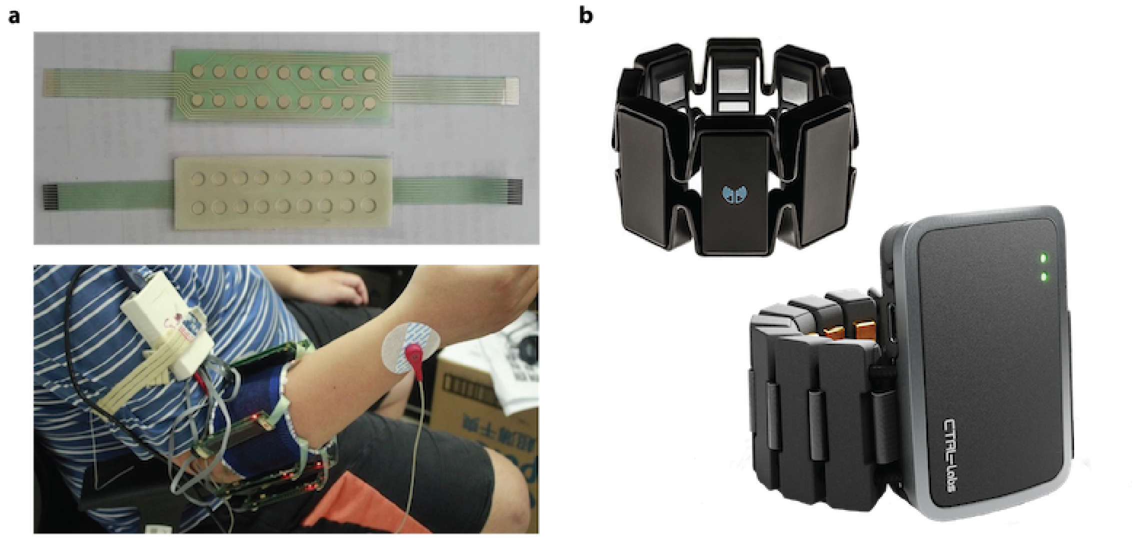


Figure 3.4: **Example EMG devices with desirable characteristics. a)** High-density surface EMG recording for large channel count acquisition [148]. **b)** Fully wearable, commercial devices with wireless connectivity [3, 4].

capabilities.

3.1.2 Machine learning approaches to gesture recognition

Inferring the gestures and movements performed by a user based on their EMG signals is a well-studied problem, and a wide variety of signal processing and machine learning techniques have been leveraged with good accuracy in a lab setting. A wide range of EMG-based control strategies for human-computer interaction with various levels of complexity have been proposed. The simplest and earliest is direct EMG control, in which the power or amplitude of a recorded EMG signal can be directly used to modulate some single degree-of-freedom (DOF) actuator, be it a digital cursor or a joint in a prosthetic [156]. This, however, is a very unnatural mode of user interaction, requiring users to activate specific muscles rather than perform normal movements. At the other extreme, EMG decoders take a similar approach to traditional brain-machine interfaces [157]. Much like true neural interfaces, individual MUAPs may be decoded from decomposed surface EMG signals to derive motor unit recruitment for a very fine-grained representation of muscle activity [158]. At a slightly higher level, EMG can also be decomposed into muscle "synergies" which correlate to individual degrees of freedom in the arm and hand [159]. While these methods would clearly give the best performance, they depend on computationally expensive decomposition algorithms and very high-quality recordings.

EMG-based pattern recognition lies between these extremes on the spectrum of control strategies. Rather than directly deriving control signals from EMG, features like signal amplitude or frequency components are extracted from the EMG signal given to a machine learning algorithm. Labeled training data are used to build a classification model for selecting an inferred gesture from among a discrete set of gesture classes 3.5. Various machine learning algorithms have been applied to this problem, including: k-nearest neighbors [160], support vector machines (SVM) [161–163], linear discriminant analysis (LDA) [164], and artificial neural networks [165, 166]. Most recently, hyperdimensional (HD) computing has also applied for the gesture recognition task [134, 135, 167, 168].

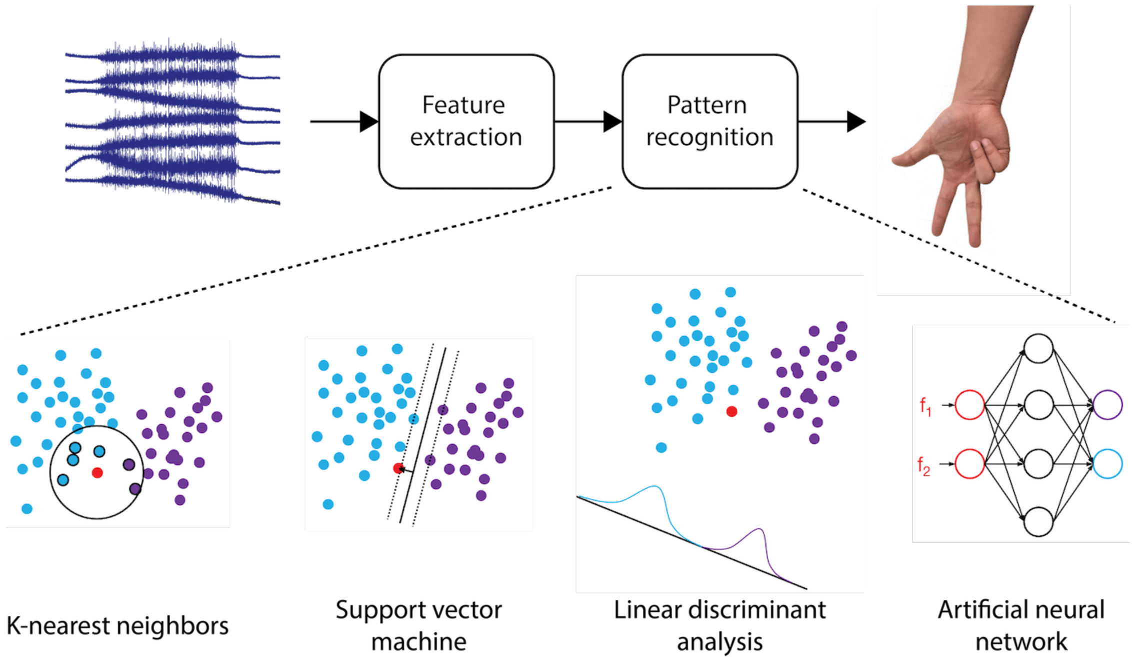


Figure 3.5: **Gesture classification using EMG pattern recognition.** Features are first extracted from the EMG signals and fed into a pattern recognition algorithm (top). Typical algorithms for EMG pattern recognition include k-nearest neighbors, support vector machines, linear discriminant analysis, and artificial neural networks (bottom).

Many of these methods can achieve comparable classification accuracy, so the choice of which model to implement depends heavily on other application constraints. For instance, the classification model may be deployed on a wearable device, as local (in-sensor) processing has advantages over wirelessly streaming raw data to an external computational device. These include reduced communication link bandwidth, lower radio power requirements, as well as improved latency and security. In these cases, the computational complexity in inferring the gesture becomes a limiting factor, with more complex algorithms requiring higher energy and introducing longer latency. Modern, light-weight machine learning algorithms perform well when the training data align with the conditions that are expected during deployment, but when the initial training of a classification model fails to capture a wide set of conditions, the classification accuracy of the model degrades, resulting in sub-optimal performance or poor user experience. For instance, recording EMG from a wearable device is afflicted by many signal variations from sweating, fatigue, varying muscle contraction effort, and electrode displacement due to changing situational contexts such as limb and body position or device doffing and donning [160, 169–173] (Figure 3.6). Therefore, the ability to train and update an in-sensor classification model on the fly during practical application is desirable.

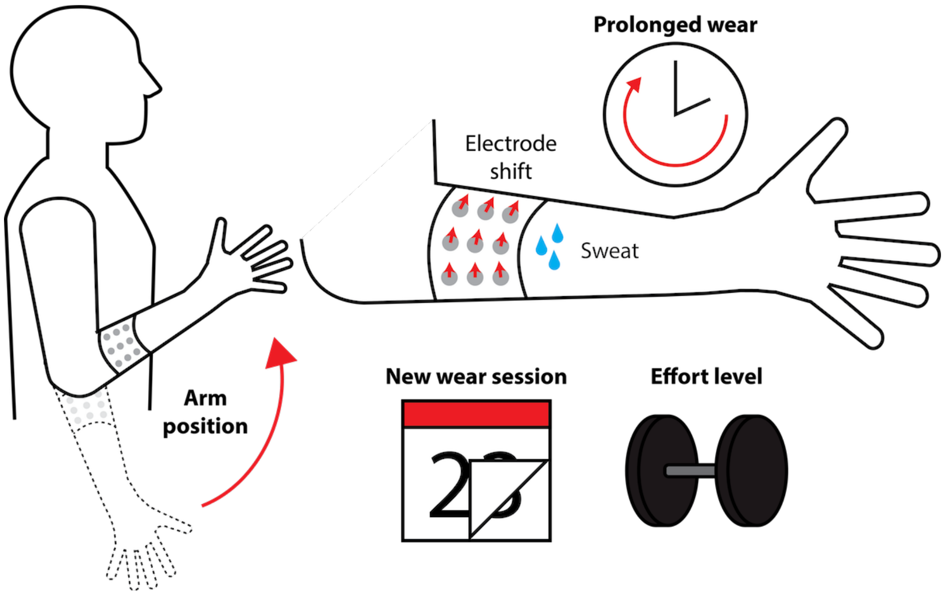


Figure 3.6: **Situational contexts encountered during EMG gesture recognition.** Context changes include different arm positions, prolonged wear of the device in a single wear session, differences over multiple wear sessions, and the effort with which gestures are performed. These changes can introduce EMG signal variations which confound the classification of gestures.

In summary, a device for performing EMG gesture recognition should have: a small, standalone form factor for autonomy; a high channel count and high-density electrode placement for more comprehensive spatial coverage and improved classification accuracy; and in-sensor intelligence that is generalizable or adaptable to various wear conditions and situational contexts. In the rest of this chapter, I present the FlexEMG device, which can satisfy these requirements using a screen-printed, conformal electrode array and in-sensor adaptive learning capabilities. Together with my collaborator Dr. Ali Moin, we implement a neuro-inspired hyperdimensional computing algorithm locally for real-time gesture recognition, as well as model training and updating under varying situational contexts. The system can classify 13 hand gestures with 97.12% accuracy for two subjects when training with a single trial per gesture. A high accuracy (92.87%) is preserved when expanding to 21 gestures, and accuracy is recovered by 9.5% by implementing model updates in response to varying conditions, without additional computation on an external device.

3.2 FlexEMG: Wearable EMG device with in-sensor learning capabilities

For the application of wearable, in-sensor hand gesture recognition using EMG, we started off with a device platform very similar to the WAND system from Chapter 2. Dubbed the FlexEMG device (Figure 3.7), the wearable EMG sensor has four main improvements over the WAND platform: 1) a more standard rectangular shape and general purpose connectors for various electrodes (as opposed to custom cortical microelectrodes); 2) the addition of inertial measurement unit functionality to simultaneously measure biosignals as well as 6 axes of the accelerometer and gyroscope; 3) much more extensive utilization of the on-board FPGA computation capability for gesture classification; and 4) interface with a custom, screen-printed flexible electrode arrays for 64-channel surface EMG acquisition [135].

The system is an example of a hybrid method for interfacing soft, conformal sensors and hard silicon-based integrated circuits. Working with Dr. Jonathan Ting, Dr. Natasha Yamamoto, and Dr. Yasser Khan, we created both a prototype flexible electrode array, using a standard, commercial flexible PCB process (not shown), as well as custom electrode array made in a custom screen-printing process using conductive dielectric inks on a thin polyethylene terephthalate (PET) sheet. The PET sheet is highly flexible and conforms well to the complex three-dimensional form of forearm muscles during contractions and relaxations. The electrodes were patterned as a uniform 16×4 array of circular electodes (4.3 mm dia.) and connective trances sing conductive silver ink (NovaCentrix CG57b) sintered by photonic curing (Figure 3.7b, c) [174]. The silver ink formulation included binders to help with cohesion of the silver microflakes as well as adhesion to the substrate. Compared to high-density electrode array fabrication methods reported in literature [175, 176], our screen-printing solution with photonic sintering allows the use of temperature-sensitive substrate materials and requires only sub-second curing time for cheaper and faster large-scale production. A dielectric encapsulation layer (NovaCentrix DE SP1) was subsequently printed with via holes for exposing the electrode pads while insulating the conductive traces from the skin. Each printed layer was 15 µm thick. Overall dimensions (29.3 cm x 8.2 cm) were chosen to wrap around the entire circumference of an above-average sized forearm, capturing activity of the extrinsic flexor and extensor muscles involved in finger movements with low inter-electrode pitch in both the proximal-distal and medial-lateral directions. Four electrodes per column were spaced 14.3mm apart, and the columns were spaced 17.8mm apart with vertical offsets of 7.15mm.

The flexible electrode array was interfaced to the main FlexEMG board (Figure 3.7d,

e) using a Flat Flexible Connector (FFC) on a two-layer adapter board (Figure 3.7c). The board was powered using a single 3.7 V, 240 mAh lithium-ion battery weighing 4 g. The total weight of the wearable system was 26 g, with the computational unit PCB weighing 6 g and the electrode array and adapter weighing 16 g. In this configuration, battery life during continuous gesture recognition was approximately 6 hours, with power dissipation being dominated by raw data packetization and wireless transmission at greater than 1 Mbps.

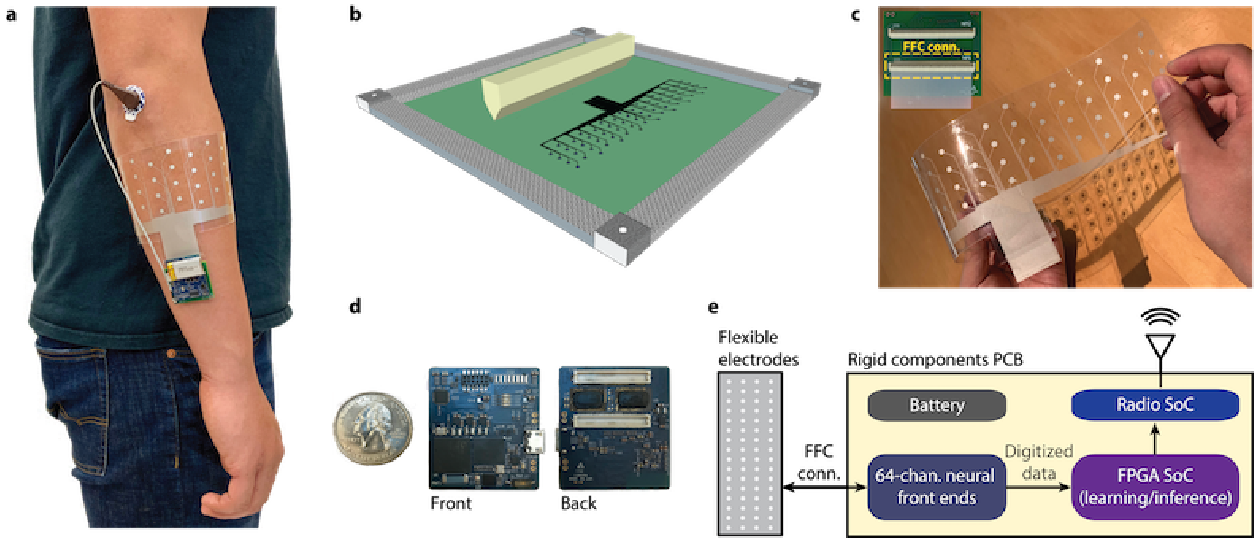


Figure 3.7: **FlexEMG system for EMG-based gesture recognition. a)** The device donned on the forearm. **b)** Illustration of the screen-printing process. A squeegee and a screen are used to print sEMG electrode arrays onto flexible substrates. The electrode pattern is defined by features on the screen. **c)** The custom-designed, flexible 16×4 array of electrodes that conforms to the forearm to provide high-density, large-area sEMG recordings without individual wires. An adapter PCB with a Flat Flexible Connector (FFC conn.) is used to interface the flexible electrode array with the rigid components. **d)** The miniaturized, 8-layer printed circuit board (PCB) that accommodates the complex, rigid components responsible for sensing, processing, and telemetry. **e)** Block diagram of the main components constituting the wearable system. Adapted from [135].

3.2.1 Offline dataset collection

In order to assess signal quality, develop our gesture classification algorithm, and tune model parameters, we used the system to record an offline dataset of sEMG signals from five healthy, able-bodied, adult, male subjects who were asked to perform gestures in multiple contexts that simulated everyday use of an sEMG acquisition system. All experiments were performed in strict compliance with the guidelines of IRB and were approved by the Committee for

Protection of Human Subjects at University of California, Berkeley (Protocol title: Flex EMG Study. Protocol number: 2017-10-10425). Informed consent was obtained from all participants. Two subjects were experienced with myocontrol, and the remaining three subjects were new to myocontrol. The dataset included flexion and extension of different finger degrees of freedom (DOF) (Figure 3.8), with single degree-of-freedom (single-DOF) gestures being performed in four different situational contexts and multiple degree-of-freedom (multi-DOF) gestures being performed in only the baseline situational context. The device was worn by each subject as shown in Figure 3.7a, with the center of the array roughly aligned to the ulna of their dominant arm. Before wrapping the array around the subject's forearm, a small drop of conductive hydrogel (SignaGel Electrode Gel, Parker Laboratories, Inc.) was applied to each electrode surface individually to improve the skin-electrode interface impedance. The array was laid on a flat surface with electrodes facing up, and the subject was instructed to roughly align the center of the array to the ulna of their dominant hand. The ends of the array were then wrapped around each side of the forearm while the subject remained at rest, and the ends were joined together with strips of adhesive tape. A single commercial Ag/AgCl electrode (H124SG, Covidien Kendall) was attached to the subject's elbow on the same arm as a reference potential for the voltage measurements.

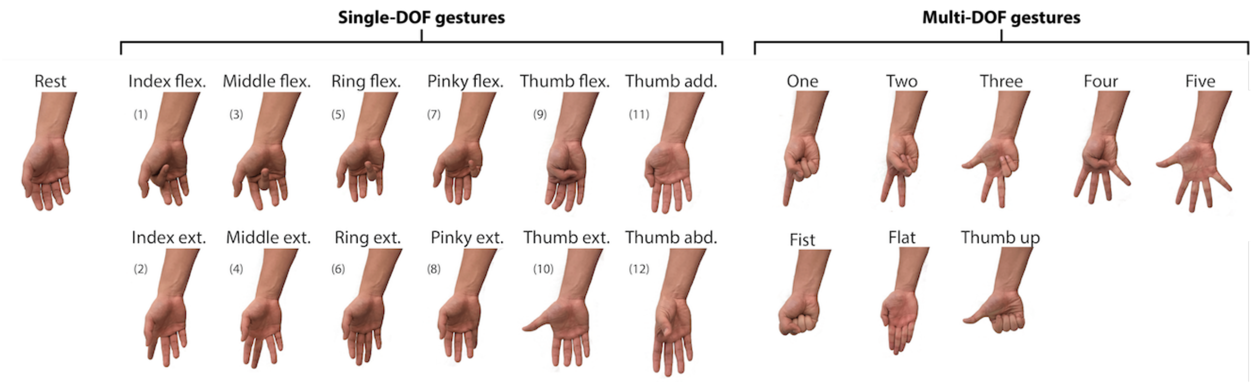


Figure 3.8: **Hand gesture classes.** The single degree-of-freedom (DOF) gesture subset includes individual finger flexions (flex.) and extensions (ext.) along with the "rest" gesture. The multi-DOF gesture subset includes common, isometric hand postures involving multiple fingers. Adapted from [135].

The changing situational contexts that were included in the offline dataset included changing arm position, changing effort levels (not used for this study), new device wear session, and prolonged wear of the device (Figure 3.6). Recordings were divided into sessions, each representing a different use case or wear condition for the device, and each consisting of a different subset of gestures (Table 3.1). During each recording session, the subject performed 5 trials of each of the gestures. Each trial lasted 11 seconds, with the subject being instructed to begin the gesture during the first two seconds and relax during the last two seconds. The middle 4 second period contained the steady-state portion of the gesture performance, which was the only portion used in this analysis. Trials were each separated by 3 second relax periods.

Session	Gestures	Context
1	Single-DOF	Baseline, relaxed arm position
2	Multi-DOF	Baseline, relaxed arm position
3	Single-DOF	New arm position (arm wrestle)
4	Effort level	Low effort, relaxed arm position
5	Effort level	Medium effort, relaxed arm position
6	Effort level	High effort, relaxed arm position
7	Single-DOF	New wear session, relaxed arm position
8	Single-DOF	Prolonged wear, relaxed arm position

Adapted from [135].

Table 3.1: Offline EMG recording sessions and situational contexts

3.2.2 EMG recording characterization

Figure 3.9a shows example waveforms recorded from all 64-channels during the flexion and extension of the middle finger DOF. Each 50 ms segment of each waveform is colored based on the mean absolute value (MAV) feature derived from that segment, indicating the local amplitude of the sEMG. Activity in contrasting subsets of channels (overlaying the anterior flexor muscles or posterior extensor muscles) can be seen in the antagonistic movements. Signal quality as acquired using our wearable biosensing system was compared to available recordings of the same gestures using a traditional commercial sEMG interface (Cometa used in Ninapro DB4 [177–179]) as well as a benchtop high-density sEMG acquisition setup (CapgMyo [148]). To compare the power spectrum of the recordings, we computed the Welch's power spectral density estimate for the channel with the highest SNR while performing the middle extension gesture (gesture 4), the gesture that produced the highest SNR across all three devices (Figure 3.9b). The frequency contents of the different recordings were qualitatively similar. Spot signal-to-noise ratio (SNR) was calculated by comparing the power spectrum during a gesture performance to the power spectrum during rest (Figure 3.9c). Overall SNR (calculated as the integral of spot SNR) varied for different channels and different gestures (Figure 3.9d). The Cometa and CapyMyo systems exhibited better peak SNR for the best channels and associated best gestures, likely because those systems consist of differential, bipolar recording configurations with improved common-mode noise rejection. Despite this, median SNRs for the three systems were approximately the same.

Furthermore, SNR values for our system were similar before and after long, 2-hour wear sessions during which subjects could sweat and the conductive gel could smear. On average,

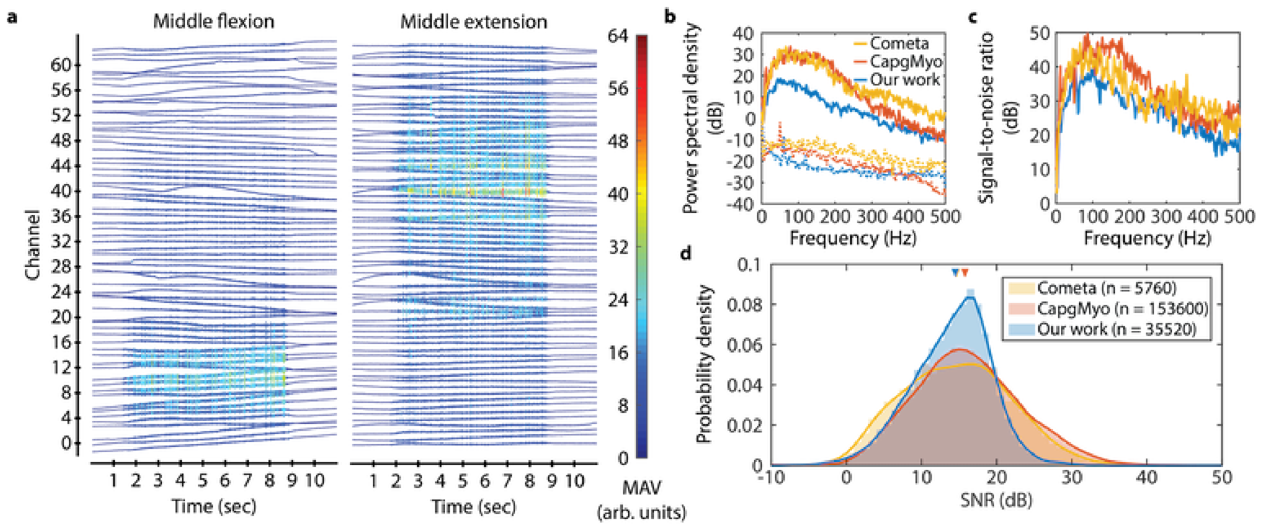


Figure 3.9: **FlexEMG recorded signal quality. a)** Example raw waveforms recorded from all 64-channels during middle finger flexion and extension. Channels are organized starting with channel 0 on the surface nearest the radius, with increasing channels wrapping around the anterior side of the forearm, to the surface nearest the ulna (channel 32), and then around the posterior side back to the radius. Each 11 s gesture trial is divided into 1.5 s rest, 2 s transition period to the gesture, 4 s hold period, 2 s transition period back to rest, and 1.5 s rest based on the instructions given to the subject. The color of the waveform indicates the local amplitude of the sEMG, as measured by mean absolute value (MAV) calculated over 50 ms segments. **b)** Example Welch's power spectral density estimates of sEMG recordings from single channels of three systems, including a commercial sEMG interface with individually placed bipolar electrodes (Cometa + Dormo, Ninapro DB4 [177–179]), a benchtop high-density sEMG acquisition setup (CapgMyo [148]), and our system. Solid lines represent the signal spectrum during performance of the middle extension gesture, while dotted lines represent the signal spectrum during rest. Spectral densities are normalized to the 0 Hz component of the rest spectrum for each recording system for ease of comparison. **c)** Spot signal-to-noise ratio for the example performances in **(b)**, measured as the ratio between the middle flexion spectrum and the rest spectrum. **d)** Distribution of overall signal-to-noise ratio for the three compared systems for all channels during the performance of all single-DOF gestures. Shaded boxes are the probability density histogram of SNR values, and lines are a Gaussian kernel fit to the distribution. Downward pointing triangles represent the medians. Adapted from [135].

there was a slight increase of about 1.4 dB in SNR after the 2 hour wear session (Figure 3.10), indicating that signal quality remained after prolonged use.

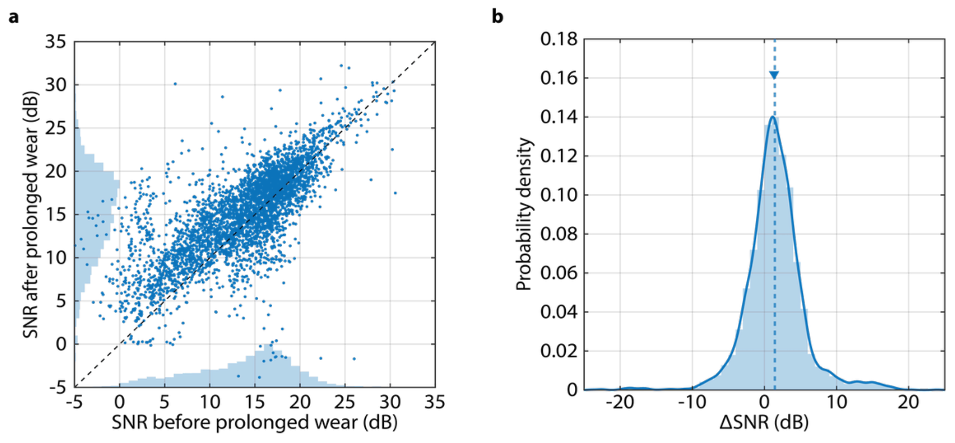


Figure 3.10: **Effect of prolonged wear on signal quality. a)** Distribution of SNR values upon first donning the device (before prolonged wear) and after 2 hours of continuous wear of the device (after prolonged wear). Univariate histograms of SNR before and after prolonged wear are shown on the axes. Scattered points each represent different combination of subject, channel, and gesture performed. Dashed line shows where SNR before and after prolonged wear would be equivalent. **b)** Distribution of the change in SNR from before to after a 2 hour period of prolonged wear. Shaded boxes are the probability density histogram of SNR shift values (ΔSNR), and lines are a Gaussian kernel fit to the distribution. Downward pointing triangle is the median value, and vertical dashed line is the mean value. Adapted from [135].

3.3 Hyperdimensional computing for biosignal classification

To perform the actual hand gesture recognition, we have implemented a model based on hyperdimensional (HD) computing, an emerging computing paradigm that supports fast, simple learning and is inherently robust against noise and errors [180]. HD computing takes advantage of information being represented by very high dimensional vectors (hypervectors) to perform otherwise complex tasks, such as classification or reasoning using simple computational operations [180]. This approach has already shown promising results in classification tasks for physiological signals such as sEMG [134], electroencephalography (EEG) [181], and electrocorticography (ECoG) [182], discriminating up to five classes on offline datasets.

A major advantage of HD computing over traditional methods described above for hand gesture recognition is the simplicity with which a classification model can be trained and updated. Classification is implemented as a nearest-prototype search, in which prototypes are created by projecting features to the HD space and then superimposing training examples together. This computation is fast, and classification is robust due to the distributed representation of information across a large number of elements.

3.3.1 Fundamentals of HD computing

The human brain contains billions of neurons and synapses, suggesting that large circuits are fundamental to its computational power. HD computing [180] explores this idea by performing computation using vectors of very high dimensionality (e.g., 10,000). Hypervectors can be combined into new vectors using well-defined vector space operations while maintaining the original information with high probability, and the composite hypervectors can be compared for similarity using a distance metric over the HD vector space.

In hyperdimensional spaces, the likelihood that any two randomly selected hypervectors are nearly orthogonal is extremely high. For instance, in a bipolar HD space, initial "seed" hypervectors can be generated at random using an equal number of randomly placed $+1$s and -1s. Such HD vectors are used to represent the basic elements in the system (e.g., the electrodes [167]), and are stored in an item memory (IM) that functions as a fixed symbol table. The following vector space operations can be used on the elements of the IM to encode information: element-wise multiplication ($\times$), or binding, takes two vectors and yields a third vector that is dissimilar (orthogonal) to the two. Element-wise addition ($+$), or bundling, takes several vectors and yields their mean vector that is maximally similar to all of them. These operations, along with scalar multiplication ($\cdot$) and permutation (ρ) of vector components for sequences, form an algebraic field beyond arithmetic and linear algebra.

Once composite hypervectors are formed that represent different features or patterns, classification can be performed using a nearest-prototype approach (Figure 3.11. For instance, in the gesture recognition case, we would like to generate a single class prototype for each gesture class. Class prototypes are calculated by superimposing all training examples from a particular class through the bundling operation, and the collection of prototypes is stored in an associative memory. Similarity between a query hypervector and the class prototypes is measured using Hamming distance, and the label of the most similar prototype is chosen as the inferred gesture class. Because of the way information is distributed across thousands of bits, this learning approach is very robust to noise. Furthermore, only small amounts of data are required to build reasonably useful prototypes, enabling few-shot learning [134]. Finally, a single iteration over the training dataset is required to completely train the model, which is in contrast with other neuro-inspired approaches in which training often employs sophisticated, iterative frameworks and is much more computationally demanding than classification (e.g., gradient descent with backpropagation).

3.3.2 EMG feature extraction and projection to hypervectors

For EMG-based gesture recognition using HD computing, we have developed a projection algorithm that encodes the level of activity recorded by each electrode, the spatial distribution of that electrode at each moment of time, and the progression of that spatial distribution over several consecutive time windows. Figure 3.12 describes the algorithm for projecting 64-channel EMG data into 1,000-dimensional hypervectors. Importantly, the same projec-

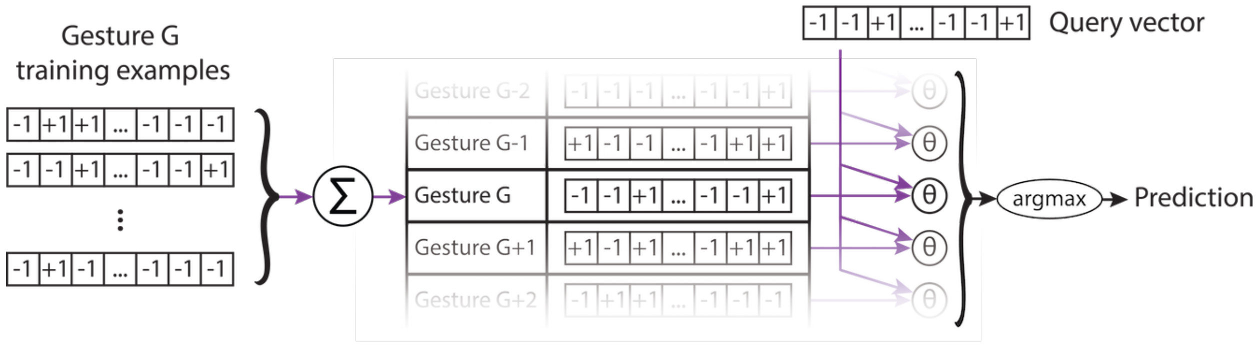


Figure 3.11: **Classification of encoded hypervectors.** Class prototypes are learned through superposition of training examples, and inference is performed through nearest neighbor search among the prototypes based on a similarity function θ. Adapted from [135].

tion process is used for both learning and inference, allowing reuse of hardware modules for both in-sensor training and classification [183].

For each segment index t, the feature vector f^t is projected into a hypervector S^t representing spatial information, i.e. which electrode channels have higher feature values (Figure 3.12b,c). Electrode channels are represented by unique, pseudo-random bipolar hypervectors stored in an item memory (IM) which remains unchanged throughout the training and deployment of the algorithm. Each electrode channel vector $IM(ch)$ is scaled by its corresponding feature value f^t_{ch}, and all scaled vectors are summed element wise. The resultant spatial hypervector is then formed as $S^t = \sigma(\Sigma_{ch} IM(ch) \cdot f^t_{ch})$, where σ is a bipolar threshold function that turns positive elements to $+1$ and negative elements to -1 (Figure 3.12c). The group of N spatial hypervectors is then encoded into a single spatiotemporal hypervector G^t (Figure 3.12b,d). The order in which they occur is encoded by performing a bitwise rotation by k bits (ρ^k), with increasing k for older hypervectors (Figure 3.12d). The rotated hypervectors are multiplied together element-wise, resulting in a single hypervector representing the entire window of data.

As described in Figure 3.11, the encoded spatiotemporal hypervectors can be used either as training examples for creating or updating a model, or as queries for inference using a trained model. We implemented offline learning and classification experiments in MATLAB in order to select features and tune our model hyperparameters. Cross validation was performed for each experiment by dividing the relevant dataset into a training set, consisting of one randomly selected trial of each gesture, and a testing set, consisting of the remaining four trials of each gesture. This division was repeated 100 times per experiment. Mean absolute value (MAV) was determined as the optimal feature function, along with the 250 ms classification window length, and 5 feature segments per classification window, by heuristically comparing different hyperparameters in the HD encoding architecture described in

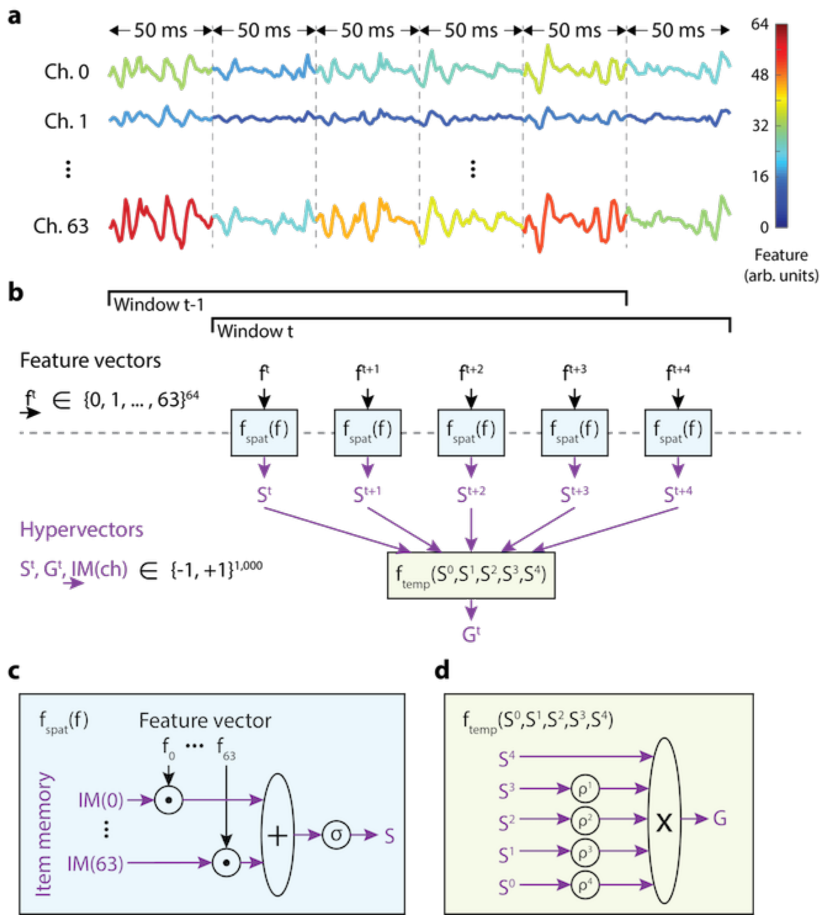


Figure 3.12: **HD projection from EMG data to hypervectors. a)** Example EMG data and feature values (in this case mean absolute value). Three example traces are shown divided into 50 ms feature segments, with segments color coded based on feature values. **b)** A single classification window at time t consists of $N = 5$ feature segments, and consecutive classification windows overlap by $N - 1$ windows. Each feature segment f^t is encoded into a spatial hypervector $S^t = f_{spat}(f^t)$. The five consecutive spatial hypervectors are then bound temporally to form a spatiotemporal hypervector representing the entire window $G^t = f_{temp}(S^t, S^{t+1}, S^{t+2}, ..., S^(t + N - 1))$. The dotted gray line shows the division between the low-dimensional feature domain and the high-dimensional hypervector domain. **c)** Spatial encoding function f_{spat} consists of computing a weighted sum of immutable item memory (IM) hypervectors $IM(ch)$ weighted $(\cdot)$ by their corresponding features f_{ch}. The sum is bipolarized (σ) back to ±1. **d)** Temporal encoding function binds consecutive spatial hypervectors S^t together through permutation by k bits (ρ^k) and elementwise multiply $(\times)$ operations. Adapted from [135].

Figure 3.12a and b. We calculated 8 different features with feature segment lengths of 10, 15, 20, 25, 35, 50, 70, and 100 ms. The tested feature functions (listed in Table 3.2) were chosen based on literature recommendations [184] as well as estimates of implementation complexity. We then created overlapping classification windows which consisted of N consecutive feature segments (e.g., 250 ms classification windows consisting of 5 50 ms feature segments are shown in Figure 3.12a and b). We tested all possible values of N for creating classification windows less than or equal to 250 ms in length, based on latency requirements recommended by the literature [185, 186].

Feature	**Description**	**Calculation**
DAMV	Difference absolute mean value	$\frac{1}{N}\sum_{k=1}^{N}\lvert\Delta x_k\rvert$
DASDV	Difference absolute standard deviation value	$\sqrt{\frac{1}{N}\sum_{k=1}^{N}\Delta x_k^2}$
MAV	Mean absolute value	$\frac{1}{N}\sum_{k=1}^{N}\lvert x_k\rvert$
MFL	Maximum fractal length	$\log_{10}\left(\sum_{k=1}^{N}\Delta x_k^2\right)$
RMS	Root mean square	$\sqrt{\frac{1}{N}\sum_{k=1}^{N}x_k^2}$
WL	Waveform length	$\sum_{k=1}^{N}\lvert\Delta x_k\rvert$
WAMPm	Willison amplitude	$\sum_{k=1}^{N}\begin{cases}1, \Delta x_k > m\\ 0, \Delta x_k \leq m\end{cases}$
ZCm	Zero crossing	$\sum_{k=1}^{N}\begin{cases}1, \Delta x_k > m, \mathrm{sgn}(x_{k-1}) \neq \mathrm{sgn}(x_k)\\ 0, \text{otherwise}\end{cases}$

Adapted from [135].

Table 3.2: EMG features tested for feature selection

To choose the best feature function, we picked the best performing classification window/feature segment size combination for each subject and feature. Figure 3.13a shows the classification accuracy results for each feature. The mean absolute value (MAV) achieved the best classification accuracy across the five subjects. Figure 3.13b shows classification results for different feature segment and classification window length combinations using the MAV feature. For shorter feature segment lengths, there exists an optimal classification window length that is less than 250 ms. However, longer feature segment lengths still achieve better accuracy, with the best performing combination being 50 ms feature segments with a maximal, 250 ms classification window. This combination was used for all other analyses.

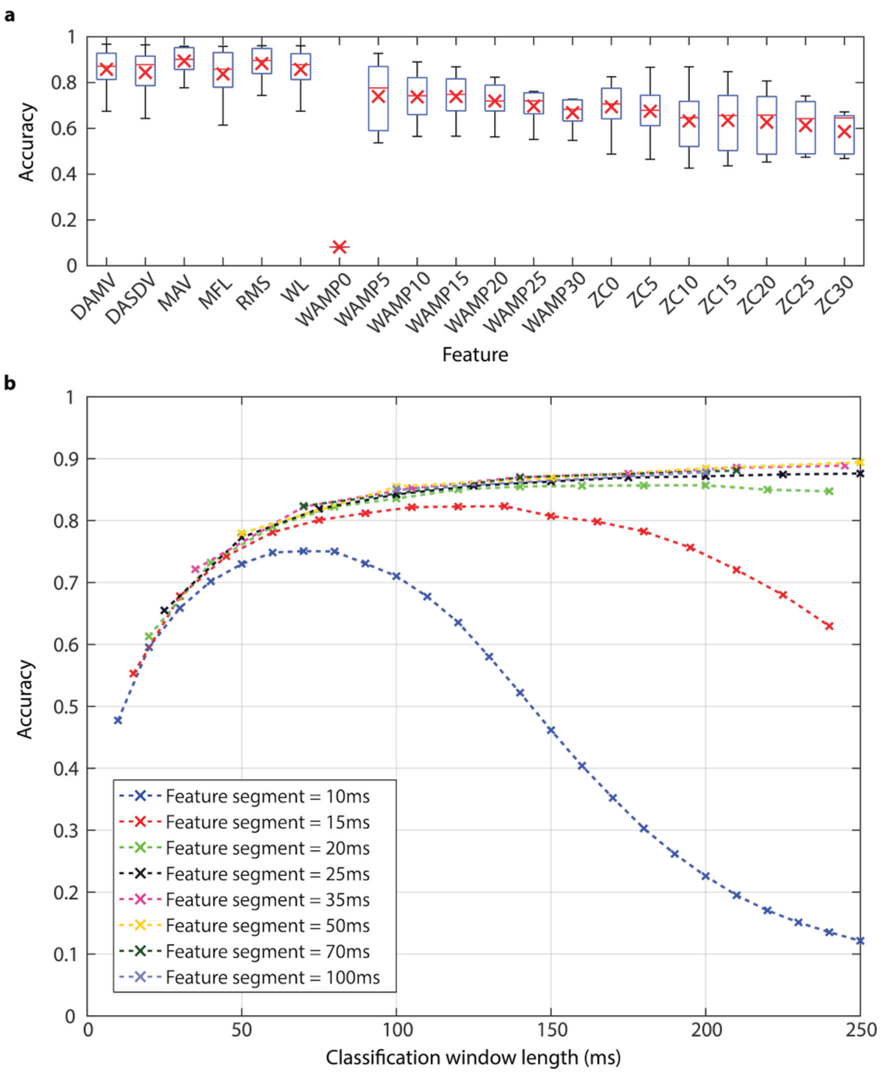


Figure 3.13: **Feature and classification window selection. a)** Comparison of classification accuracy per subject for the different features listed in Table 3.2. For each subject, the best performing feature segment and classification window length were selected for each feature. Boxes cover the middle 50% of values, horizontal red lines denote medians, and whiskers span the full range of data. Red x's denote means. **b)** Classification accuracy for different feature segment and classification window lengths for the MAV feature. For each feature segment length, N such feature segments can be combined to form classification windows of up to 250 ms. Adapted from [135].

To assess whether all 64 electrode channels were necessary for optimal classification performance, we ranked channels based on their importance according to two metrics. The first metric was a univariate Fisher score, which normalizes between-class feature variance by within-class variance [187]. The second metric was a feature gain parameter measured after training a gradient boosted tree-based algorithm for discriminating gestures (500 trees with tree depth of 2). This feature gain represented the relative weight of a feature in making a prediction with the forest. Channel Fisher scores followed similar anatomical patterns, indicating that channels placed over the flexor and extensor muscles produced features that varied more with the different gesture classes (Figure 3.14a). Boosted trees feature importances were sparser, with a few channels much more important in making a prediction than the rest.

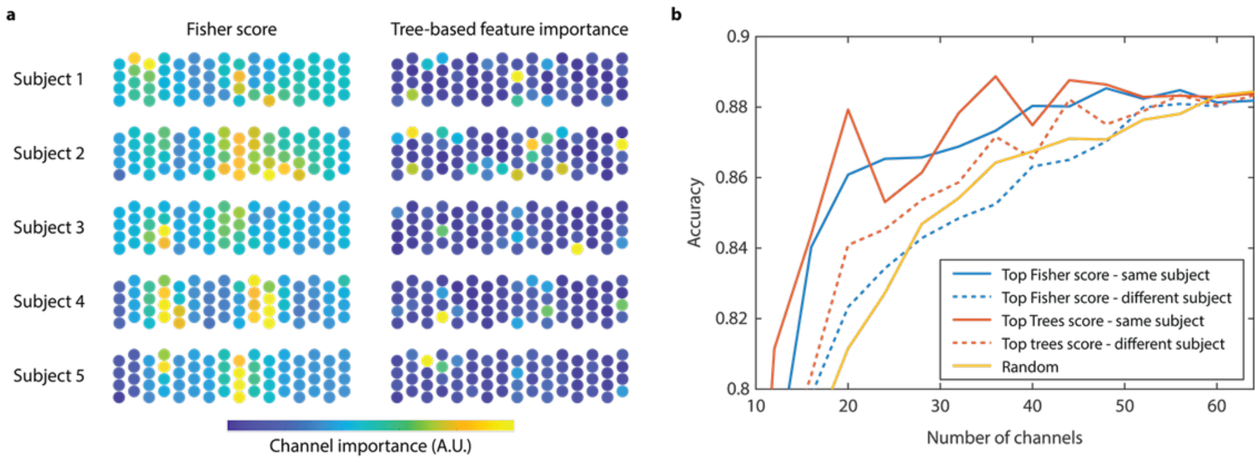


Figure 3.14: **Channel selection and effect on classification accuracy. a)** Channel importance as assessed using univariate Fisher score (left) and gain-based feature importance from a gradient boosted tree-based classifier (right) for five subjects. Each circle represents an electrode positioned as such in the array, and the color of the circle represents the normalized feature score for that channel. **b)** Classification accuracy degrades when reducing the number of channels for hypervector encoding. Five different methods were used to select the top subset of channels: Fisher scores from the same subject (solid blue line, average over 5 subjects); Fisher scores from a different subject (dotted blue line, average over 5 subjects, each tested with feature scores from all 4 other subjects); tree-based feature importance from the same subject (solid red line); tree-based feature importance from a different subject (dotted red line); and randomly (solid yellow line, averaged over 10 repetitions per subject). Adapted from [135].

We ranked the electrodes based on their feature importances for each subject and each metric and used these rankings to gradually drop out channels while applying our hyperdimensional computing algorithm. Classification accuracy gradually decreased as fewer channels were used and steeply degraded when fewer than 16 channels were kept (Figure 3.14b). Notably, classification accuracy was better preserved when dropping channels based on the subject's own feature importance ranking as opposed to dropping channels based on a different subject's rankings. In the latter case, performance was marginally, if at all, better than randomly dropping channels. Figures 3.15c and 4d show the top 32 and top 16 channel subsets for each subject based on the different rankings. Although there is some overlap, the differences in these subsets are enough to cause the accuracy degradation of a few percent when using channel rankings from a different subject.

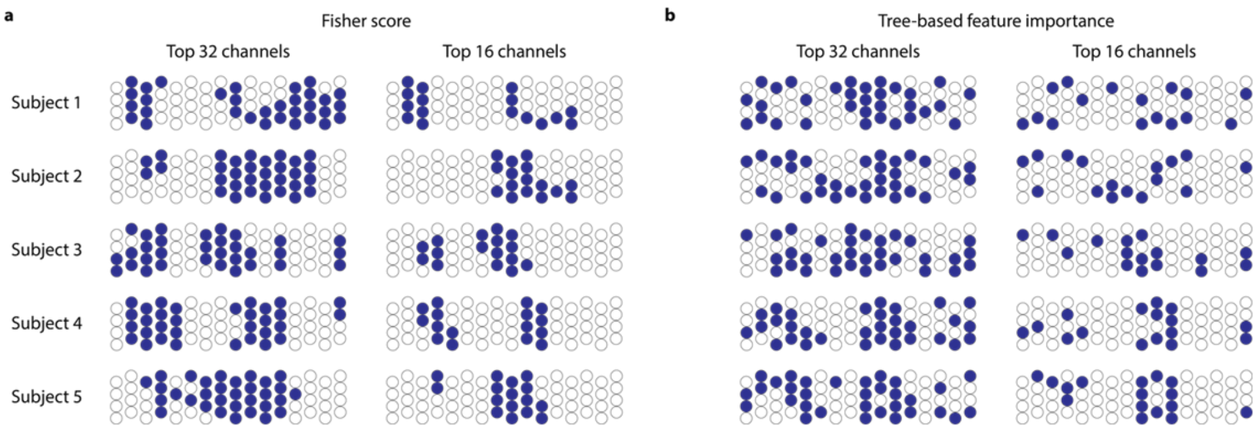


Figure 3.15: **Most useful channels for classification accuracy per subject.** Top 32 and 16 channel subsets for each subject based on **(c)** Fisher score and **(d)** tree-based feature importance. Adapted from [135].

3.3.3 Offline classification analysis

Figure 3.16 shows the classification accuracy results using the offline dataset. The left portion of Figure 3.16a shows classification accuracy using the hyperdimensional computing model in the baseline situational context for the single- and multi-DOF gesture subsets separately, as well as for all 21 gestures together. The updated model for classifying all 21 gestures was created by concatenating the associative memories previously trained separately for the single- and multi-DOF gesture subsets, requiring no new training computations.

To model fast, online learning, all classification accuracies were measured using cross-validation with one trial of each gesture for training and the remaining trials for testing. We also implemented the standard leave-one-out cross validation (LOOCV) using 4 trials for training and 1 trial for inference. Using LOOCV, the HD classifier achieves 92.47% accuracy for single-DOF, 91.71% for multi-DOF, and 89.85% for all gestures, as compared to the 88.87%, 85.27%, and 84.53% respectively when training with only 1 trial. Thus, although

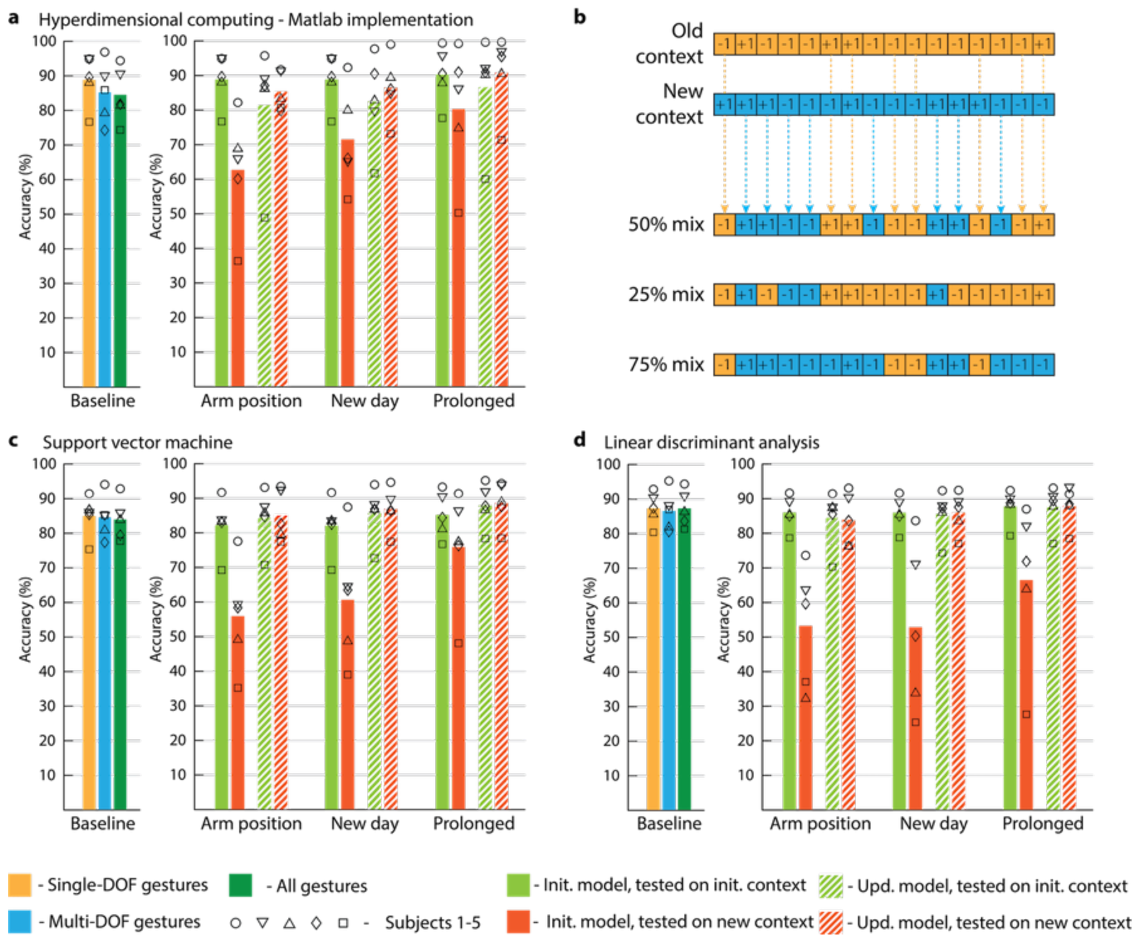


Figure 3.16: **Offline dataset classification accuracy with updates.** **a)** Results with 10,000-bit hypervector dimension and floating-point features. **b)** Bitwise merge of two prototype hypervectors approximates finding weighted centroids of clusters from different contexts of the same gesture class. The updated hypervector randomly takes elements from the stored prototype hypervector and the new context prototype hypervector. The proportion of bits taken from each hypervector determines the relative weight of each context in the updated prototype. **c)** Results using a support vector machine model with a linear kernel. **d)** Results using linear discriminant analysis. Adapted from [135].

slightly better performance could be achieved with more training data, one-shot learning still produced viable results.

To measure classification accuracy of the HD model across different situational contexts, we performed three experiments in which a model was trained using data from the initial, baseline context and then used to classify gestures in a new dataset from a different context. Figure 3.16a (right) shows the classification accuracy for the three tested contextual variations: arm position, new wear session, and prolonged wear. Although training and inference were performed on separate datasets, we still used a single trial of each gesture to train the model, and reported accuracies are averages over each of the five training trials. For each of the contextual variations, we saw accuracy degradations of 26.17% (arm position), 17.34% (new day/wear session), and 9.95% (prolonged wear), suggesting that updates were necessary.

In order to update a stored prototype to incorporate information about a new situational context for the same gesture, an approximated majority vote can be performed by merging elements from both the stored and the newly computed prototype hypervectors (Figure 3.16b). Different proportions of new and old elements can be taken to weigh the contribution of each context. As more contexts are encountered, this proportion can be changed to tune the decay rate in order to avoid catastrophic forgetting of initial contexts. Very little additional overhead is required to perform model updates, compared to standard gesture recognition algorithms like support vector machines, in which new support vectors must be selected amongst new and old training examples [188, 189], or linear discriminant analysis, in which training set statistics must be recalculated to a relatively high degree of precision [164, 190].

To test model updates, prototype hypervectors from the new contexts were trained using only a single trial of each gesture and merged with the initial model. We wanted both the initial and new contexts to have equal contributions to the updated model, so the merged hypervector randomly took 50% of its elements from each prototype hypervector. Reported classification accuracies are the average of all 25 combinations of initial context prototype and new context prototype. After updating initial models for the new arm position, wear session, and prolonged wear contexts, the classification accuracy on gestures in the new context was improved significantly (22.66%, 14.96%, and 10.51%, respectively) at the expense of small degradation (7.46%, 6.37%, and 3.65%, respectively) in classifying the initial context gestures. This accuracy degradation is due to the bitwise merge operation being only an approximation of the accumulation required to fully retrain a model on data from both contexts. This approximation introduces more error with increasingly distant context prototypes.

We also performed the experiments with two traditional machine learning algorithms commonly used for sEMG-based gesture recognition: support vector machine (Figure 3.16c) and linear discriminant analysis (Figure 3.16d). Both models were implemented in MATLAB. For the support vector machine model, we used a linear kernel with cost parameter C = 500. For linear discriminant analysis, we implemented the standard MATLAB function included in the Statistics and Machine Learning toolbox, with maximum regularization (gamma = 1) for the covariance matrix estimation. Both of these traditional methods performed similarly well to our model, but with the distinction that the "updated" models in both cases were actually retrained from scratch in order to incorporate training examples from both contexts. This makes them less suitable for online implementation.

3.4 In-sensor, real-time hand gesture learning and classification

In order to implement the HD classification model efficiently in hardware description language (HDL) for the FPGA on-board the FlexEMG device, certain optimizations needed to be made. Chief among them were calculation of quantized features, efficient generation of electrode IM hypervectors, and development of the associative memory for prototype storage. Furthermore, we had to reduce hypervector dimensionality in order to fit our design in within the constrained on-board resources. These modifications to the classification algorithm were expected to decrease classification performance, but we also anticipated that performing real-time classification could in fact improve accuracy due to the availability of live visual feedback.

3.4.1 HDL implementation of training, classification, and model updates

Raw 15-bit ADC codes were used as the input for feature extraction, with incremental calculations of the 50-sample MAV performed with each new sample. The implemented arithmetic operations consisted only of addition, two's complement inversion, and arithmetic right shift for division. Features were quantized and saturated to 6-bit integers based on analysis of the offline dataset, optimizing for the dynamic range (range divided by step size) given the arithmetic requirements. We also reduced the dimensionality of the hypervectors from 10,000 bits in the offline analysis to 1,000 bits for the FPGA implementation in order to reduce memory footprint. Figure 3.17 shows offline dataset accuracy results using these lower-dimensional vectors and 6-bit quantized features, demonstrating only a few percent accuracy degradation compared to the larger, original model.

Although described so far using bipolar ($\{-1, +1\}^{1,000}$) hypervectors, the algorithm can be equivalently implemented with binary ($\{0, 1\}^{1,000}$) hypervectors by mapping from arithmetic to Boolean operations [191]. Thousand-dimensional IM elements were generated sequentially using a cellular automaton with a hard-coded seed [192] for optimal memory usage. The spatial hypervector was computed by iterating over the 64 channels, either adding or subtracting the channel's feature value from an accumulator value for each hypervector element, depending on the bit value of that element in the associated cellular automaton output. The resulting spatial hypervector was taken as the most significant bits (the sign bits) of the accumulators. Temporal encoding was then performed using a 5-position shift register with hard-wired rotation connections between each element, and the spatiotemporal hypervector was generated using bit-wise XOR between the stored shift register elements. Prototypes were generated by accumulating training examples using saturating up/down counters.

The associative memory was implemented as a shift register containing 21 prototype hypervectors (Figure 3.18). Training simply involved shifting in new bundled vectors. During inference, prototypes would be cycled through and individually compared with an incoming

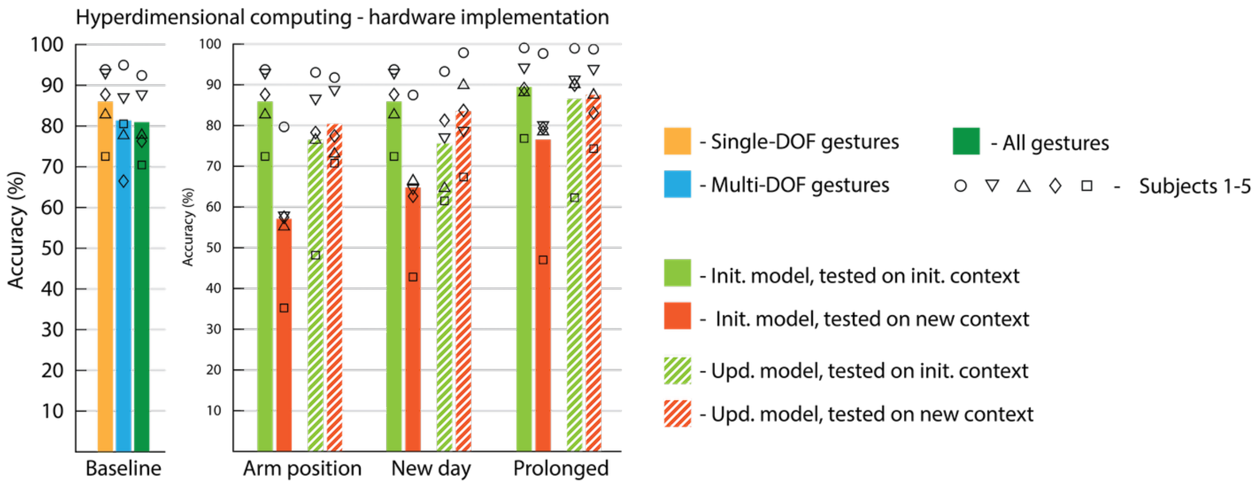


Figure 3.17: **Offline dataset accuracy using hardware-optimized implementation.** Simulated hardware design in MATLAB with 6-bit integer feature values and 1,000-dimensional hypervectors. Adapted from [135].

query hypervector. For model updates, we implemented a hard-coded merge operation that allowed half the bits from an incoming prototype to be merged with the currently stored prototype.

The implementation utilized 84% of resources on our FPGA, with detailed utilization generated using Synopsis Synplify Pro for Microsemi and illustrated in Figure 3.19. Simulations verified that the algorithm (after acquiring the last sample of the last 50 ms feature window) had a latency of 539 cycles, or 26 μs when running at 20.48 MHz (Table 3.3). The FPGA modules for classification were thus clock gated to only be active during the first 26 μs after receiving a new sample, consuming no dynamic energy during the remaining time.

Simulations and source meter measurements showed that the in-sensor HD algorithm operates with 2.437 μJ per sEMG sample and 4.39 μJ per classification every 50 samples (Figure 3.20). The power consumption of the system was measured by an Agilent B2902A precision source/measure unit in 20 μs intervals. Model training and updating consumed similar energy to classification steps. In all, the on-board algorithm accounted for less than 6% of the overall device power.

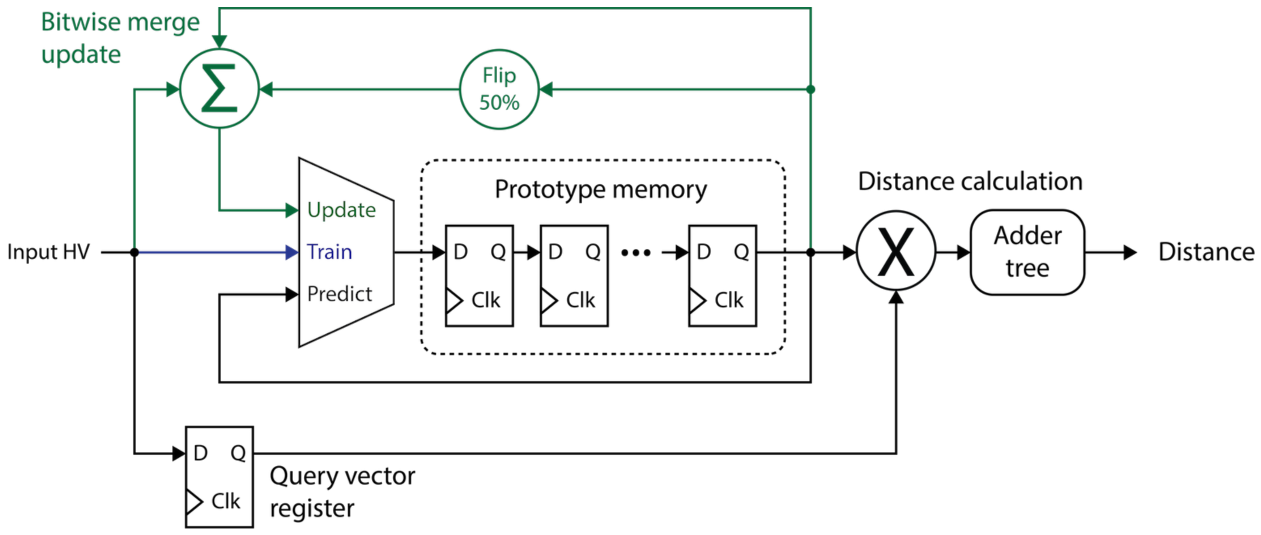


Figure 3.18: **Hardware associative memory implementation with training and update.** Prototypes are stored in a shift register for sequential access. During prediction (bottom path, black), a query hypervector is stored in a separate register and compared with each prototype by first performing a bitwise XOR ($\bigwedge$) and then summing across the resulting bits with an adder tree. During training (middle path, blue), input hypervectors are sequentially written into the shift register. During model updates (top path, green), the input hypervecgtor is merged into the currently accessed prototype using 50% of the bits.

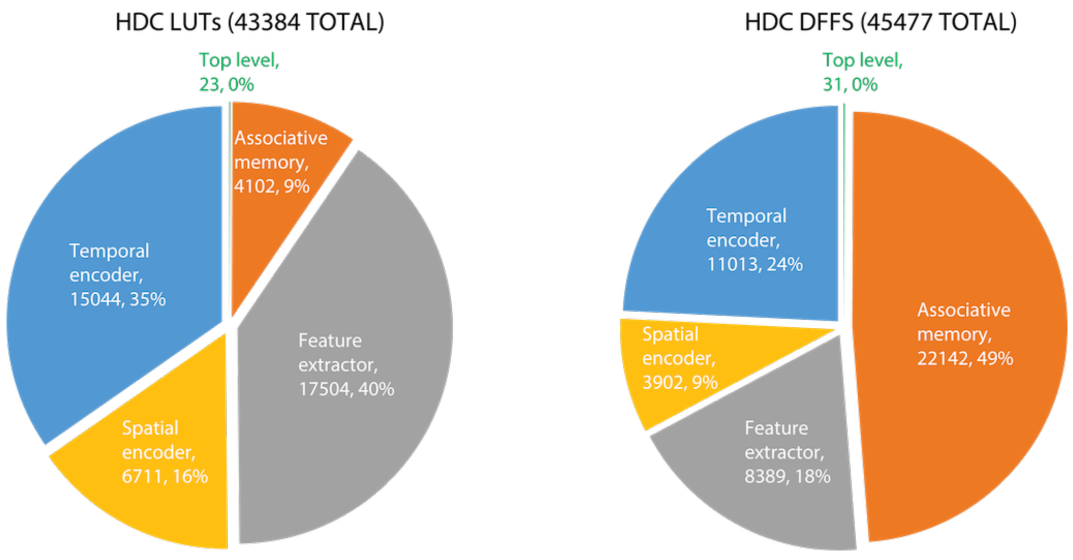


Figure 3.19: **FPGA resource utilization for in-sensor HD classifier.** Utilization is separated for each submodule of the algorithm based on the number of look-up tables (LUTs, left) and D flip flops (DFFs, right). Adapted from [135].

Submodule	**Clock cycles**	**Time (w/ 20.48 MHz clock, μs)**
Feature extractor	2	0.10
Spatial encoder	513	25.05
Temporal encoder	1	0.05
Associative memory	23	1.12
Total (inference)	**539**	**26.32**
Feature extractor	2	0.10
Spatial encoder	513	25.05
Temporal encoder	1	0.05
Total (bundling)	**516**	**25.20**
Feature extractor	2	0.10
Spatial encoder	513	25.05
Temporal encoder	1	0.05
Associative memory write	1	0.05
Associative memory predict	23	1.12
Total (train/update)	**540**	**26.37**

Adapted from [135].

Table 3.3: Algorithm latency.

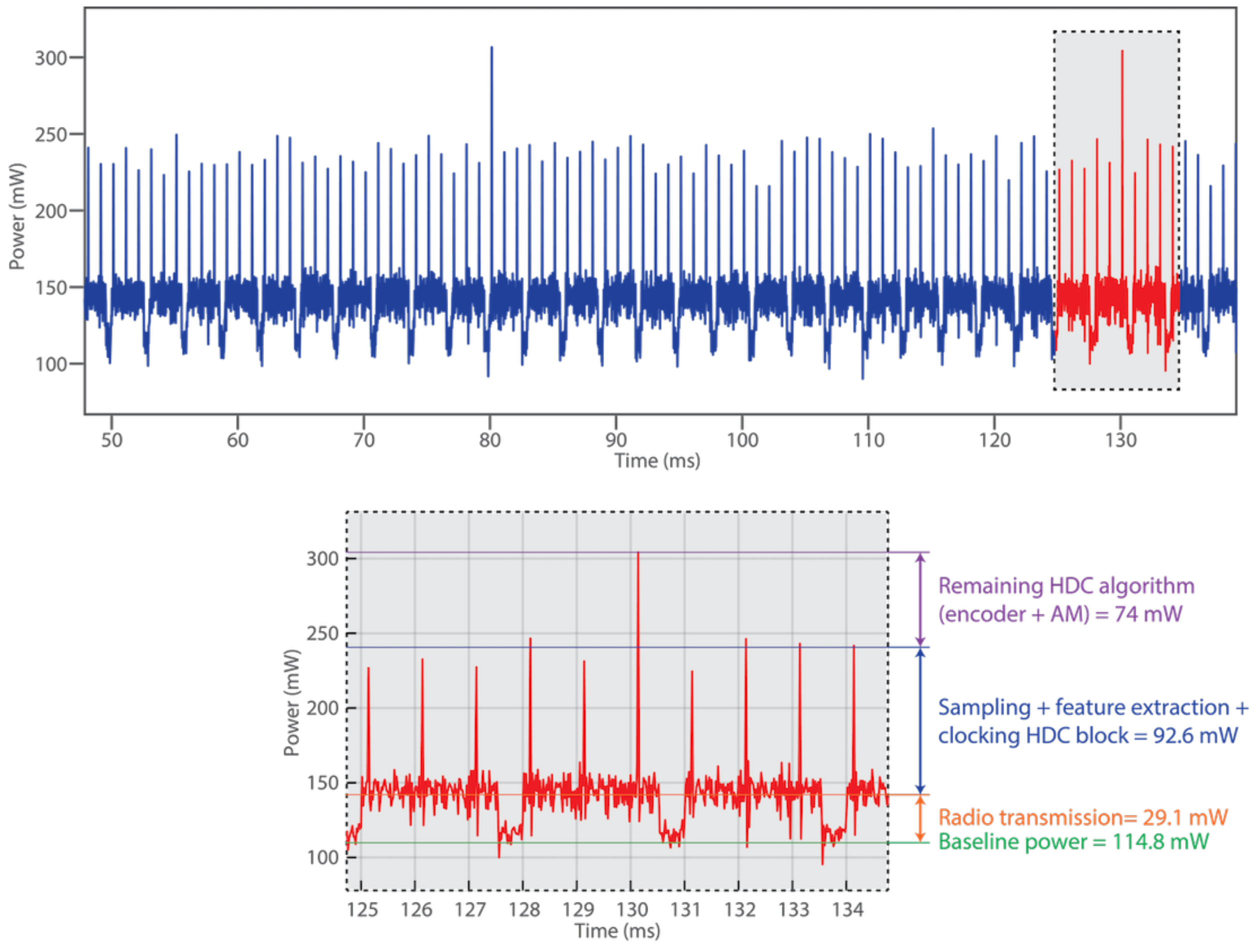


Figure 3.20: **Algorithm power consumption during classification.** The shorter peaks occurring every 1 ms correspond to the accumulation of each new sample for feature extraction. The taller peaks occurring every 50 ms include the additional power for the HD encoder and accessing the AM. The radio transmits data containing raw EMG values as well as classification labels in bursts of 3 samples. Adapted from [135].

3.4.2 Situational context experiments

We verified the online, standalone operation of our device with two subjects who were familiar with myocontrol. Four types of experiments were each conducted three times to test varying device usage conditions with all training, inference, and updates performed completely on board and in real time. A custom-made graphical user interface (GUI) was used to instruct the user on what gesture to perform, transmit the correct gesture label and the operation mode (train/infer/update) to the device, and log data. Both raw sEMG data and the classified gesture class were streamed back to the GUI and displayed in real-time, giving the subject visual feedback during inference modes.

Figure 3.21 shows example real-time prediction time series from the device during the performance of single-DOF gestures in the baseline context immediately after device donning and with a relaxed arm. Each gesture was performed once during training, including a trial at the beginning for the rest gesture. Subjects were instructed to begin each gesture within a 2 second transition period, hold the gesture for a 4 second steady-state period, and relax back to the rest gesture within a second transition period, with 3 seconds of relaxation between each gesture. The GUI was configured to only enable training mode on the device during the 4 second steady-state period. The exact same procedure was then repeated with the device in inference mode. Figure 3.22 shows the overall classification accuracy and confusion matrix for the 4 second steady-state periods of each gesture.

Actual gesture onset and offset times were determined offline with a combination of visual and change-point analysis methods in order to estimate a ground truth label (Figure 3.21). Although the classification model was not trained on data that included gesture transitions or relaxation between trials, the device still performed inference during these periods. Gesture onset- and offset-aligned predictions across baseline context tests in all 24 experiments are shown in Figure 3.23. A higher error rate during relax periods (during which the prediction should be the rest gesture) is seen both before and after gesture performance, suggesting that the short periods between gesture movements may be included as training data to improve performance. Classification of the rest state is inherently more difficult, given that our hypervector projection algorithm encodes relative feature values rather than absolute feature values. Additionally, gesture transitions produced errors, requiring about 500 ms from gesture onset (including algorithm latency) to achieve 80% classification accuracy. The larger amount of misclassification following gesture offset is in part due to unintended movements in the opposite direction or of other finger degrees of freedom during transitions back to rest.

Figures 3.24 and 3.25 show steady-state gesture classification accuracy for the different experiments requiring online, in-sensor training and model updates. The first experiment (Figure 3.24) demonstrated the ability to update the HD classification model with new gestures. We trained and tested an initial model with only the single-DOF gesture subset (Figure 3.24, yellow bar) and achieved an average classification accuracy of 98.34%. We then updated the model with the multi-DOF gesture subset and tested on all 21 gestures (Figure 3.24, green bar) with 92.87% accuracy, a 5.47% degradation. Training and testing on the

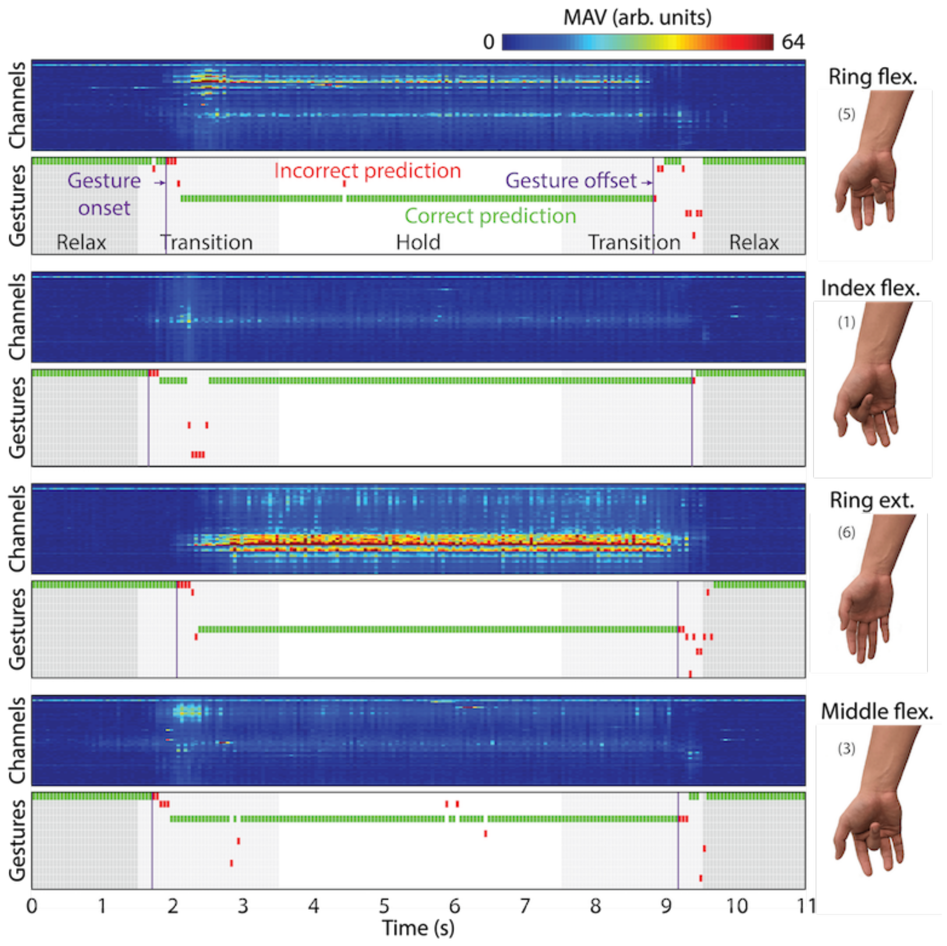


Figure 3.21: **Real-time, in-sensor classification outputs.** Examples of real-time prediction outputs for four gesture trials. For each gesture trial, the top plot displays the features calculated over 50 ms segments for all channels. The bottom plot shows the 20 Hz classification results relative to trial timing. Purple vertical lines denote offline-calculated gesture onset in the first transition period and gesture offset in the second transition period. Vertical position of predictions represents gesture class, and predictions are color-coded based on accuracy with respect to offline-estimated ground truth labels. Adapted from [135].

multi-DOF gesture only resulted in a similar accuracy of 95.04% (Figure 3.24, blue bar).

In the next three experiments, we demonstrated the ability to update the HD classification model with new contexts using only the single-DOF gestures. The context changes we explored were new arm position (going from relaxed position with the arm at the side to elbow rested on an armrest in an arm-wrestling position), new day with the device doffed and re-donned in between (>16 hr apart, mean 19 hr), and prolonged wear with the device worn while going about daily activities for two hours. An initial model was trained and

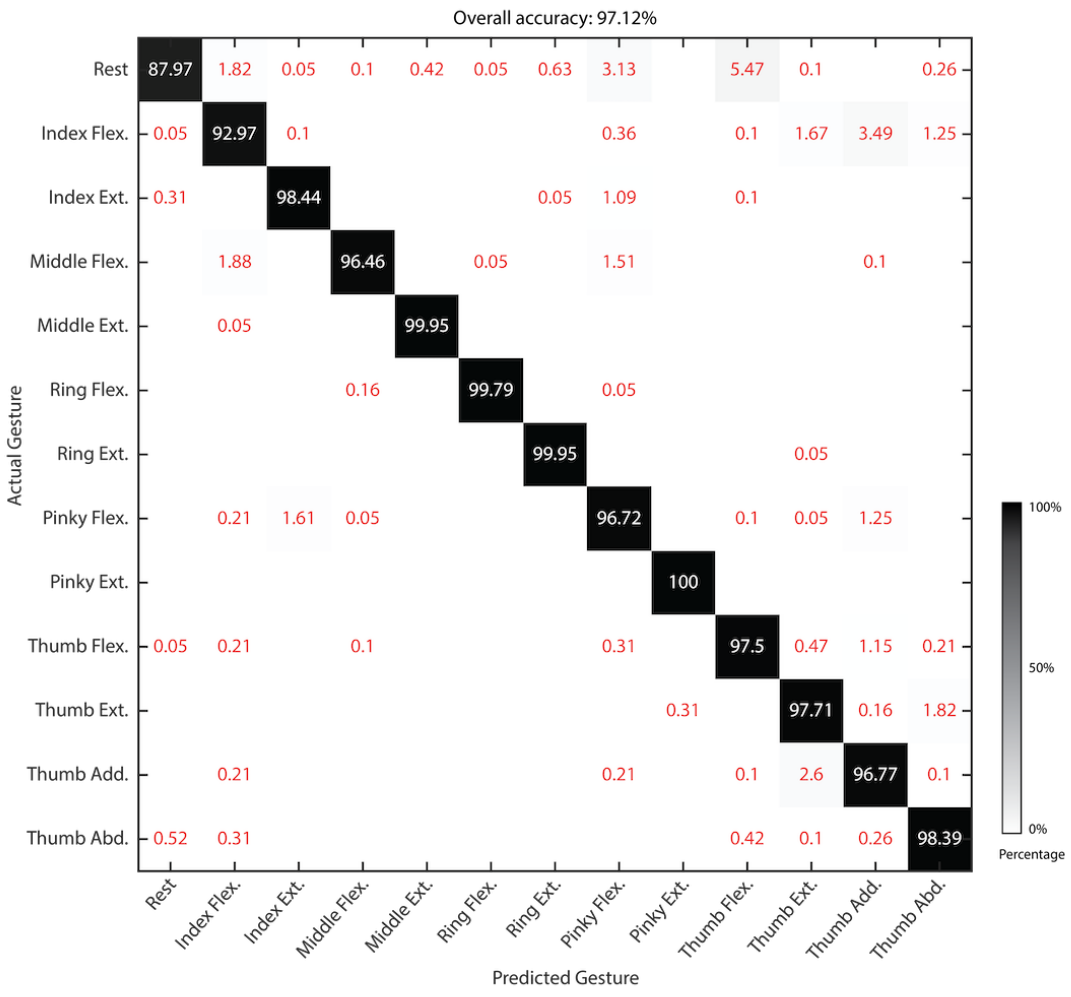


Figure 3.22: **Confusion matrix of real-time classification in baseline context.** Data consists of only the 4-second hold periods of baseline context testing for real-time experiments. White text values are percentages of correct predictions, red text value are percentages of incorrect predictions. Greyscale colored background represents proportion of predicted classes. Adapted from [135].

tested in the baseline context (Figure 3.25, solid green bar). The model was then tested in the new context, prior to updating (Figure 3.25, solid red bar). This resulted in on average a 11.89% accuracy degradation. The model was then updated in the new context and tested again (Figure 3.25, striped red bar). In the case of arm position, the updated model was also tested in the initial context (Figure 3.25, striped green bar). In each case, accuracy was recovered after updating the model for the new context with an average improvement of 9.50% compared to the initial model.

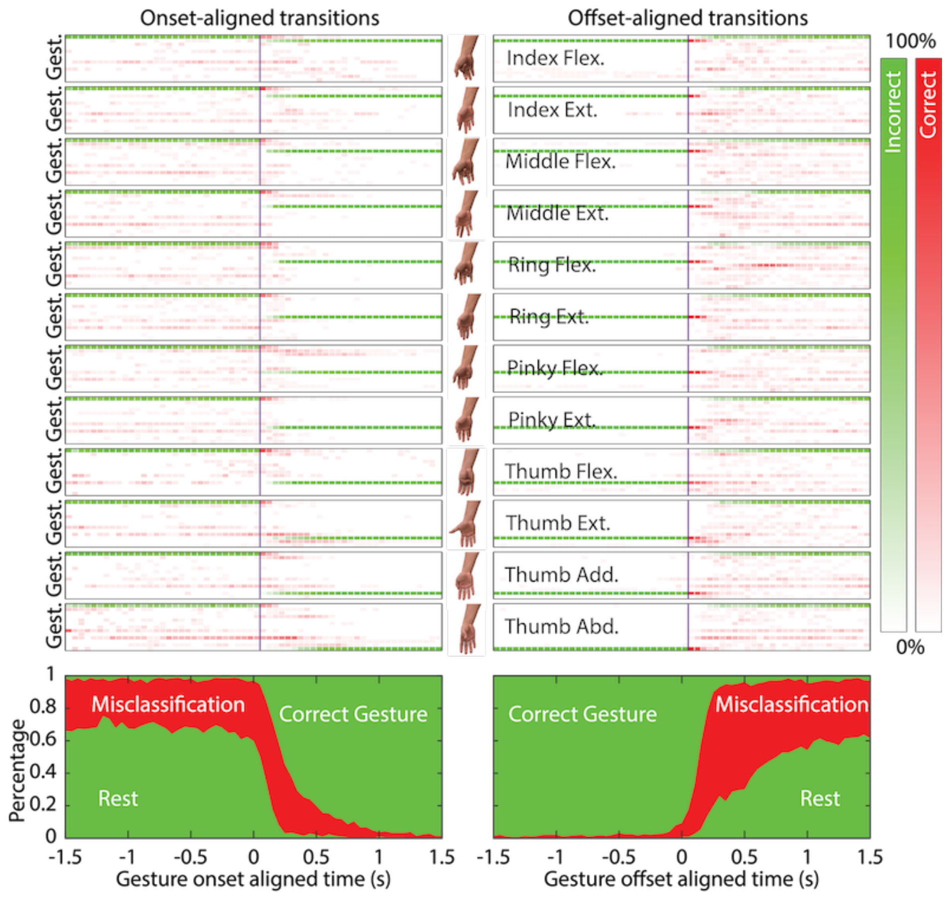


Figure 3.23: **Gesture transitions during real-time classification.** Gesture onset- (left) and offset-aligned (right) prediction statistics over all baseline context tests. Each of the first 13 rows shows the averaged 20 Hz classification results for a given gesture, with more darkly colored blocks indicating a higher proportion. Green blocks indicate correct predictions, and red blocks indicate incorrect predictions. The bottom plots show the percentages of correct (green) and incorrect (red) classifications over time, relative to the onset and offset times. Adapted from [135].

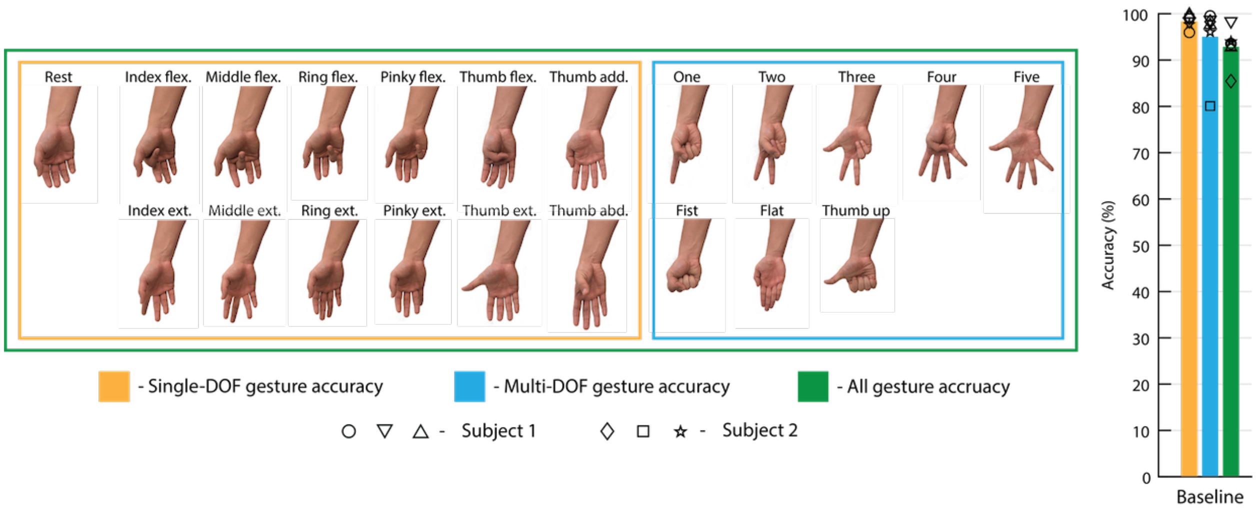


Figure 3.24: **In-sensor training, update, and classification when adding new gestures.** Two subjects performed each experiment three times. Bars represent the mean accuracy across all six trials, with overlaid data points for each individual trial. An initial model was trained and tested on single-DOF gestures only (yellow bar). The model was then updated with multi-DOF gestures to cover all 21 gestures (green bar). Results for a separate model trained and tested on multi-DOF gestures only are also illustrated (blue bar). Adapted from [135].

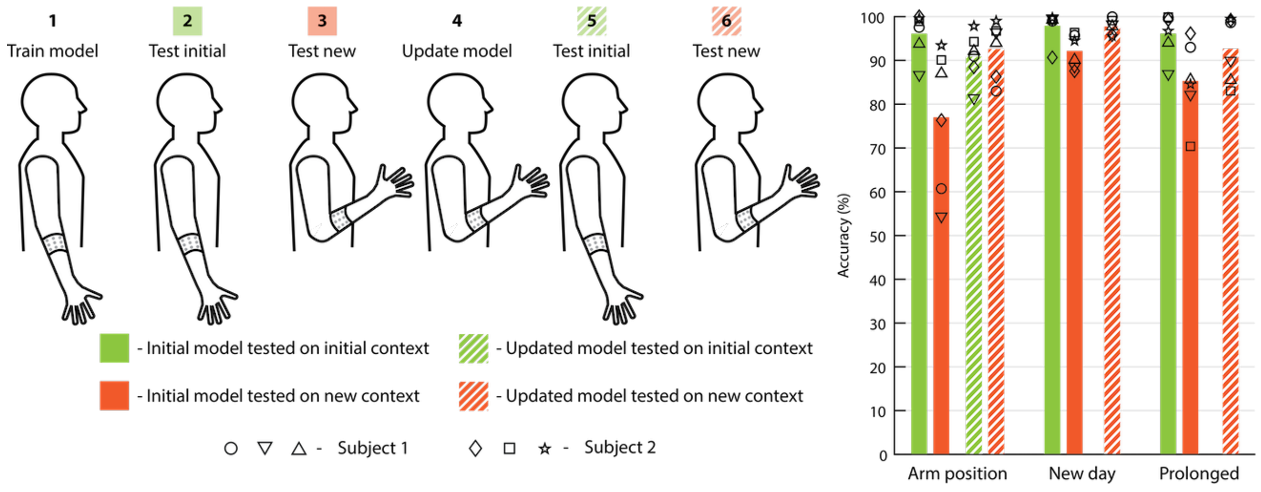


Figure 3.25: **In-sensor training, update, and classification across situational contexts.** Two subjects performed each experiment three times. Bars represent the mean accuracy across all six trials, with overlaid data points for each individual trial. For three different context changes (arm position, new wear session, and prolonged wear), an initial model was trained on the initial context (step 1) and tested on both the initial (step 2) and the new (step 3) contexts (solid bars). The model was then updated using a single trial of each gesture in the new context (step 4), and again tested (steps 5 and 6) on both contexts (striped bars). Updated models were tested only in the new context for the new wear session and prolonged wear experiments since the old context was no longer available. Adapted from [135].

3.5 Discussion

Here, I've reported on collaborative work towards a self-contained, wearable EMG biosensing system called FlexEMG that uses HD computing to process and classify hand gestures. All the functionalities of our system, including data acquisition using a printed flexible electrode interface, as well as training and classification with a machine learning model, are incorporated into a compact device that requires no external computation. The FlexEMG system is comfortable to wear and offers fast initial training, as well as on-the-fly adaptation, which are crucial for wearable human-computer interaction applications in which physiological signals vary from user to user and are not stationary. In contrast to other state-of-the-art gesture recognition systems, our device can perform training, inference, and model updates locally and in real time to adapt to changing situational contexts.

A comparison of our device to existing state-of-the-art EMG systems is given in Table 3.4. Our system is the only one tested in an incremental learning framework with in-sensor training, inference in multiple new situational contexts, and incremental updates to adapt to new contexts and recover accuracy. Compared to the systems that are capable of in-sensor classification, ours has the most electrode channels by at least a factor of 8. The custom-designed flexible electrode array and low-power integrated circuit front ends allow us to maintain a small form factor and low power consumption, comparable to systems with a smaller number of channels, which are important for device wearability. A more detailed comparison of embedded device implementations is given in Table 3.5. Our custom hardware implementation of the algorithm on an FPGA achieved the best classification energy efficiency, even with more gesture classes. Note that the total energy required for the processing (feature extraction and classification) was slightly higher compared to other reports because of the larger number of channels.

Our results are promising for the development of standalone wearable devices with full, in-sensor machine learning capabilities. Two natural extensions of this work are the incorporation of additional situational contexts (such as a larger variety of arm positions) and the incorporation of additional sensor modalities (such as the on-board IMU), both of which will be discussed in Chapter 4. As more information is encoded into gesture prototype hypervectors, it may eventually become beneficial to trade off memory footprint for improved classification performance and to represent gesture classes with more than one prototype hypervector.

	Amma et al. [149]	**Liu et al. [166]**	**Moin et al. [134]**	**Pancholi and Joshi [193]**	**Benatti et al. [194]**	**Cerone et al. [175]**	**FlexEMG**
Wearable form factor	✗	✓	✓	✓	✗	✓	✓
Electrode type	Flexible PCB	Individual surface patches	Flexible PCB	Ag-AgCl electrode band	Individual Ag-AgCl	Flexible PCB	Flexible screen-printed
Electrodes per array	192	4	64	8	8	32	64
Wireless streaming	✗	✓	✓	✗	✓	✓	✓
In-sensor classification	✗	✓	✗	✓	✓	✗	✓
In-sensor model training	✗	✗	✗	✓	✓	✗	✓
In-sensor adaptive update	✗	✗	✗	✗	✗	✗	✓
Number of classes	27	10	5	6	11	–	21
Accuracy	90.4%	94%	96.64%	94.14%	85%	–	92.87%

Adapted from [135].

Table 3.4: High level comparison of FlexEMG with other EMG classification systems.

	Amma et al. [149]	**Benatti et al. [162]**	**Liu et al. [166]**	**Pancholi and Joshi [193]**	**Benatti et al. [194]**	**Cerone et al. [175]**	**FlexEMG**
System							
Dimensions	–	85×50mm^2	34×25mm^2	–	–	30×25mm^2 PCB, $3.4 \times 3 \times 1.5$cm^3 full unit	40×33mm^2 device, 293×82mm^2 electrode array
Weight	–	–	15.2g, 4.6g battery	17.5g	–	16.7g	6g board, 4g battery, 16g electrode array
System power	–	36.4mW streaming, 86mW online recognition	100.6mW	122.5mW	10.4mW	392.7mW	141.2mW
EMG acquisition							
Electrode type	Flexible PCB	Individual Ag-AgCl	Individual surface patches	Ag-AgCl electrode band	Individual Ag-AgCl	Flexible PCB	Flexible screen-printed
Number of channels	96 differential	8 differential	4 differential	8 differential	8 differential	32 single-ended	64 single-ended
ADC resolution	–	24 bits	12 bits	24 bits	24 bits	16 bits	15 bits
Input range	–	±300 mV	–	–	–	10 mVpp	100 mVpp
Sampling rate	2.048 kHz	1 kHz	1.6 kHz	1 kHz	1 kHz	2.048 kHz	1 kHz
Processing and ML algorithm							
Feature set	Root mean square	Envelope	Mean absolute value	Root mean square, waveform length, zero crossing, slope sign change	Root mean square	–	Mean absolute value
Algorithm	Naive Bayes	Support vector machine	Artifical neural network	Linear discriminant analysis	HD computing	–	HD computing
Latency (data window /algorithm)	–	3ms/1ms	200-250ms /<0.2ms	200ms with 75ms increments/–	60ms/36μs	–	250ms/26μs
Energy per feature extraction	–	–	–	–	0.86μJ	–	121.85μJ
Onboard inference	No	Yes	Yes	Yes	Yes	–	Yes
Onboard training	No	No	No	Yes	Yes	–	Yes
Onboard updating	No	No	No	No	No	–	Yes
Dataset	5 healthy	4 healthy	4 healthy	3 healthy, 1 amputee	10 healthy	–	5 healthy (offline), 2 healthy (online)
Number of gestures	26 + rest	6 + rest	10	5 + rest	10 + rest	–	20 + rest
Classification accuracy	90.4%	89.20%	94%	94.14%	85%	–	84.53% offline, 92.87% online
Wireless							
Protocol	Wired	Bluetooth	Custom 2.5GHz	Wired	Bluetooth	WiFi	Custom 2.4GHz
EMG streaming	No	Yes	No	No	Yes	Yes	Yes
Classification streaming	No	Yes	Yes	No	Yes	no	Yes

Adapted from [135].

Table 3.5: FlexEMG system specifications compared with other EMG classification systems.

Chapter 4

Expanding the Learning Capacity of HD Classifiers

In Chapter 3, I presented a classification algorithm for EMG-based hand gesture recognition using hyperdimensional (HD) computing. In the work covered there, we demonstrated the ability to update an HD classification model on-the-fly to incorporate information about a single new situational context such as a different limb position. While this was a big step towards continuous learning, real-world usage of a gesture recognition algorithm requires performance in a larger number of contexts while still being limited to computationally efficient operations. In this chapter, I first describe how the classification accuracy using an HD model can be improved when performing gesture recognition in 8 different limb position contexts. The methods include optimizing the representation of prototypes and encoding additional information through sensor fusion. I then detail how an HD classification model can be *incrementally* trained to learn more and more limb position contexts in which gestures are performed. A range of different classifier configurations are presented that trade off classification performance with the amount of memory required for storing HD model parameters.

4.1 Introduction

Hand gesture recognition using electromyographic (EMG) signals produced by muscle activity has many applications in human-machine interfaces and upper-limb prosthetic control [157]. Various machine learning algorithms have been applied to the problem, including support vector machines (SVM [161, 195]), linear discriminant analysis (LDA [196]), artificial neural networks [165], and hyperdimensional (HD) computing [134, 135, 167, 168]. Ideally, gesture recognition capabilities can be embedded on-board wearable or implantable devices and prosthetics, eliminating the need for separate computation devices and inefficient wireless streaming of raw EMG data. [135, 197]

For users of upper-limb prostheses, these technologies can enable higher levels of dexterity over existing control strategies while remaining unobtrusive. However, lack of robustness due to the variable nature of EMG signals is one of the largest issues leading to abandonment of smart prosthetics [198]. As discussed in Chapter 3, we have fully implemented an HD classification model for gesture recognition on-board the FlexEMG device that is capable of in-sensor training, update, and inference [135]. In that work, we showed the ability to update the HD model for a single new situational context that could be a change in limb position, prolonged wear of the device over several hours, or doffing and donning the device on a new day. We saw this as a step toward addressing the time-varying nature of EMG signals, which remains a major barrier to practical use. Over time, the conditions in which a gesture classifier is deployed become different than the ones in which it is trained, leading to accuracy degradation [199]. Furthermore, datasets collected for training often are not fully representative of the entire distribution of data to be encountered during real-world deployment.

The particular problem I will be addressing in this chapter is classification error due to a larger number of changing limb positions. Users must be able to expect accurate classification with their limb in various postures, but this causes recording electrodes to shift and affects gesture performance as well as underlying muscle activity [200]. These changes can cause accuracy degradation if overlooked during initial training, and they often necessitate more complex classification models. This effect has been observed in a number of works, and Figure 4.1 shows an example of how HD classifiers trained in specific limb positions fail to perform well in others. A new dataset consisting of 8 different limb positions (see Section 4.2.1 for more details) was collected to evaluate performance in a larger number of limb positions. Using the same methodology as described in Chapter 3, we trained separate HD classification models for gesture recognition in each limb position and tested them both within the same position and in all other positions. While within-position classification accuracy (on the diagonal of the position confusion matrix) remained high, most of the time the classifiers saw large performance degradations when testing with data from a different position (off diagonals).

Many methods to address this issue have been proposed in the literature. The most commonly explored way is to augment training routines to include a larger number of contexts outside of ideal initial conditions (Figure 4.2a). Works have used training sets consisting of multiple limb positions [200–202], as well as other types of variations including temporal variation [201] and contraction effort level [203]. As we demonstrate below, updating HD classification models for multiple limb positions through prototype superposition in the manner described in Chapter 3 [135] enables some generalization, but that method alone still performs significantly worse than context-specific models. Because features based solely on EMG can overlap and take on more complex distributions when considering multiple limb positions, other methods have leveraged new input features or signal modalities that vary with and provide information about the limb position. Some works append accelerometer features to existing EMG feature vectors [200, 204], while others take a dual-stage classification approach (Figure 4.2b) [200, 205]. A separate gesture classifier can be trained for each limb position, with the most appropriate one selected during deployment based on limb

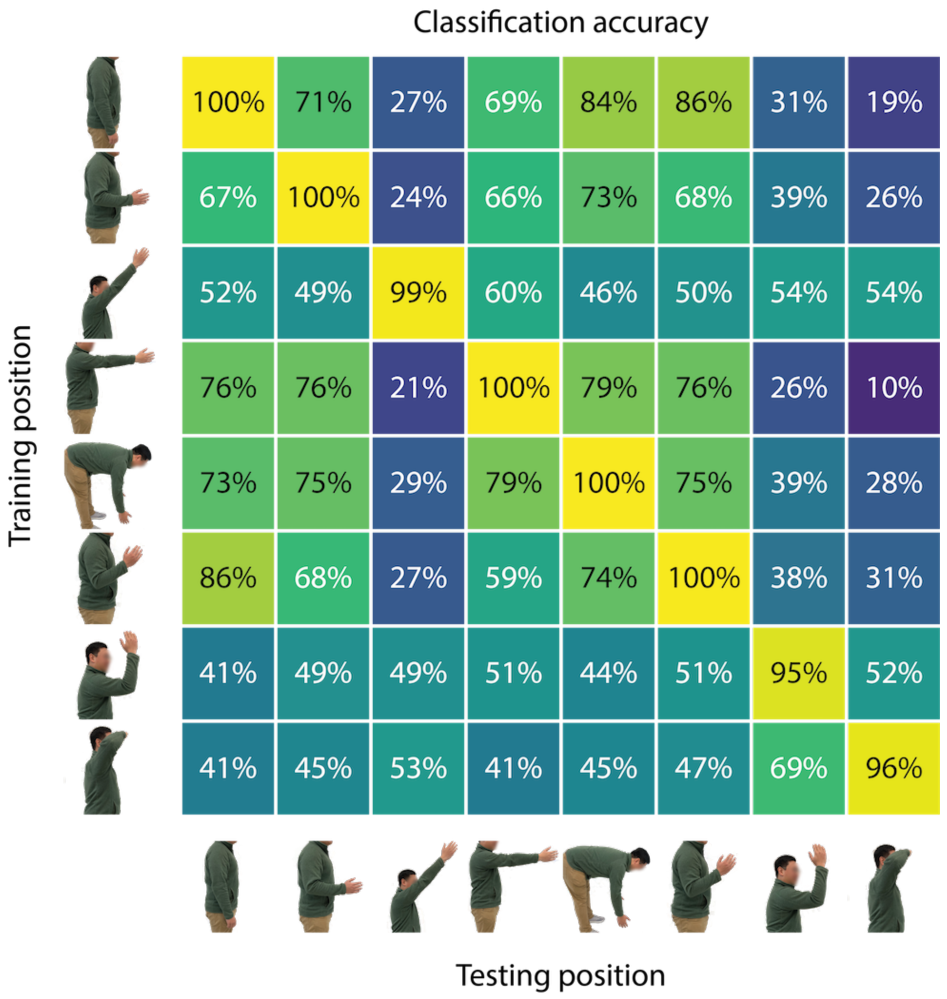


Figure 4.1: **Classification accuracy degradation across different limb positions.** Example shown for a single subject performing 13 gestures in 8 different limb positions. A separate HD classifier was trained for each limb position, and each classifier was tested in all limb positions.

position classification using accelerometer signals and either an LDA [200, 205] or SVM [206] model. A hybrid dual-stage classification approach has also been suggested, which consists of multiple classifiers each trained in a subset of limb positions, using an accelerometer for coarser limb position detection [202]. While these architectures improve classification accuracy, they come with significant implementation overhead. Memory allocation for storage of model parameters increases linearly with the number of limb positions, and an additional limb position classifier must also be implemented. These drawbacks have made dual-stage classification unattractive for in-sensor implementation.

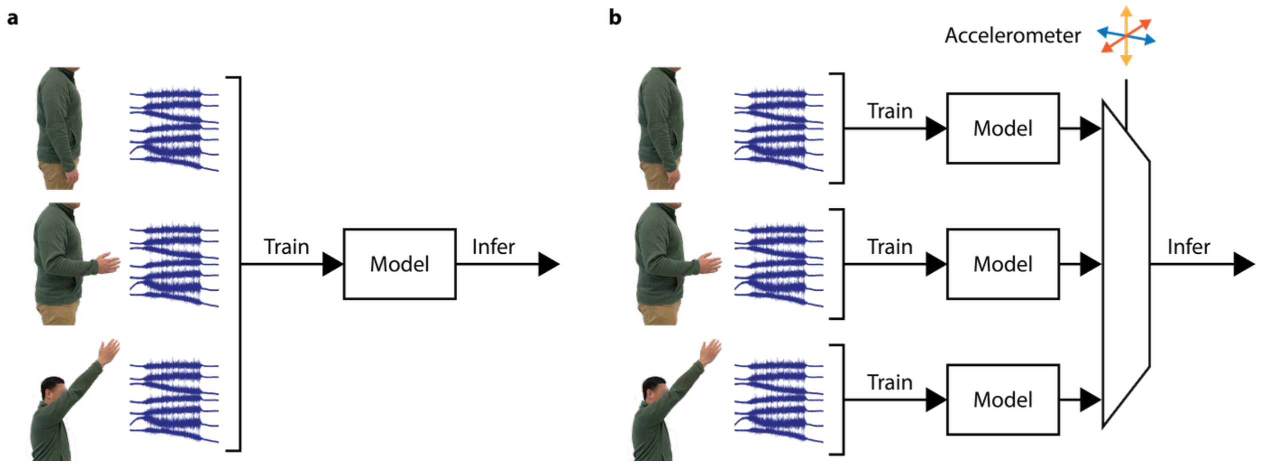


Figure 4.2: **Methods for incorporating EMG data from multiple limb positions. a)** Augmenting the dataset with multiple limb positions for training a single model. **b)** Training multiple limb position-specific models and choosing the appropriate model for inference based on limb position classification, e.g., using accelerometer signals.

For multiple limb positions, the overall training effort (for both the user and for finding generalizable model parameters) increases regardless of the methodology for incorporating the new data. As more training data is required, it becomes burdensome to the user and impractical to collect it all at once for batch training. Instead, classification models which can be incrementally trained over several sessions may be more suited for this application. When untrained limb position contexts are encountered during everyday use, smaller incremental training datasets may be collected to update the existing model. Several works have demonstrated and advocated for adaptable machine learning models and incremental learning to address EMG variation. A general framework consists of selectively updating the training dataset with new examples as new information is made available to the system (Figure 4.3) [207]. This framework is agnostic to the type of classification model used and assumes that it can be retrained in a supervised manner using the updated dataset. Adaptive, incremental training for gesture recognition has been demonstrated using both SVM [208, 209] and LDA [207, 210–213] classifiers. Much focus has been on how well models can adapt to gradual changes in the EMG over prolonged periods of use. However, when the incrementally learned contexts consist of multiple limb positions that all continue to be relevant, more attention needs to be paid to how well models also retain past information. Incrementally adapting a model to a new position must enable performance in the new context while also preserving classification accuracy when used in previous contexts.

Again, a challenge for incremental learning of multiple contexts is the increasing amount of memory required to store training examples for use in re-training or updating model parameters, as well as with the parameters themselves. There has been a lack of emphasis

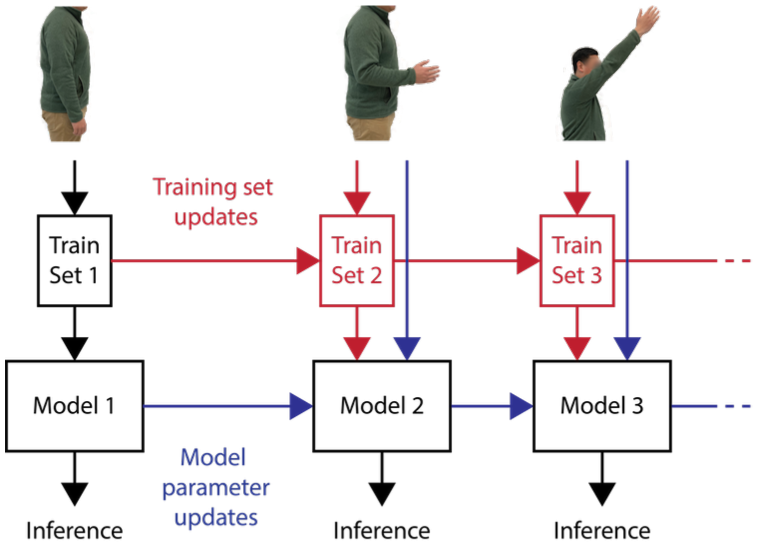


Figure 4.3: **General framework for incrementally learning multiple limb positions.** After training an initial model, updating the model through incremental training can occur in one or both ways, described by the red path and the blue path. In the red path, the training dataset is incrementally augmented, and a new model can be trained from scratch at each incremental step. In the blue path, only the model parameters are retained, and new data is incorporated by updated model parameters.

on implementing incremental learning methods with computational resource constraints in mind. Because the incremental learning framework grows a dataset that must be retained for use in subsequent training sessions, the memory footprint for storing training examples (either directly or as learned model parameters) must be considered for efficient in-sensor implementations. While model parameters effectively condense the information contained in training examples for use during inference, some portion of the original training dataset must usually be retained for training or updating the model. Online learning methods for sample-by-sample SVM training [214, 215] require previous training examples to potentially be promoted as support vectors, and models trained with stochastic gradient descent require that data from different contexts be shuffled for effective training. Some form of training example storage is needed, and it is therefore beneficial for a model's parameters to capture the necessary information for incremental training while also being able to incorporate new data regardless of the order in which it arrives (Figure 4.3).

Here, I show that methods based on HD computing are able to satisfy these requirements and can do so with computationally efficient operations for on-the-fly training. The process of learning a new example consists entirely of accumulating it into the appropriate prototype, a computation which compresses training examples into smaller model parameters and does not depend on the order in which data is presented. This makes HD classifiers inherently amenable to incremental learning. In this chapter, I will describe how HD classifier prototypes can be improved for better classification accuracy in multiple limb positions and can also be trained incrementally and in a memory-efficient manner.

I first describe the EMG and accelerometer dataset I collected from 5 subjects to evaluate the different HD classifier configurations. I then discuss two different methods for improving classification using HD computing. I briefly cover the first method, which attempts to better define class boundaries within the data by optimizing prototypes and maximizing classification margin, as opposed to simply finding class centroids through superposition. While this enabled good performance across the 8 different limb positions, it also required many iterative computations that are impractical and ill-suited for incremental learning and embedded implementation.

The second method for improving prototypes involves limb position encoding using the accelerometer sensor [168]. Properties relating orthogonal vectors in HD spaces can be leveraged to reduce the amount of interference between training examples from different contexts in superposition. I use these ideas as a foundation for a sensor fusion-based HD classification architecture where accelerometer signals are encoded into limb position context hypervectors. With this method, I show that we can effectively perform dual-stage classification while storing the position-specific parameters in superposition, rather than separately. This allows us to achieve higher classification accuracy without the drastic increase in memory footprint that normally comes from dual-stage or cascaded classifiers.

Finally, I present a range of HD classifier configurations for *incrementally* learning hand gestures in multiple limb positions. Together with the choice of a context encoding method, I investigate choices for how new training hypervectors should be superimposed into existing prototypes and how many prototypes per class are allowed. To evaluate these choices, I created an incremental training scenario in which new limb position contexts were learned one at a time in random order. I then compared the configurations using classification accuracy and memory footprint as metrics, and I implemented incrementally trained SVM and LDA models as traditional machine learning comparisons.

4.2 Limb position dataset and limits of direct superposition

To test the various methodologies for hand gesture recognition in multiple limb position contexts, I collected a new dataset with simultaneous EMG and accelerometer measurements during the performance of 13 gestures in 8 different limb positions. In this section, I describe that data collection process, as well as initial experiments demonstrating the degradation in classification accuracy caused by direct superposition of training examples from multiple limb positions to compute class prototype hypervectors.

4.2.1 Dataset collection experiments

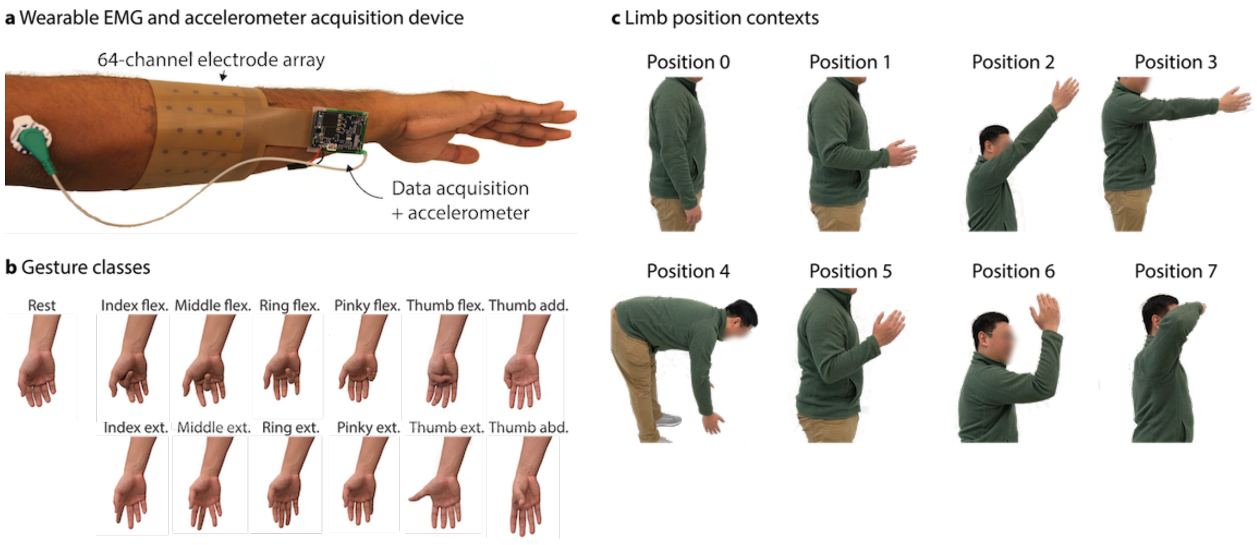


Figure 4.4: **Multi-limb-position EMG dataset collection. a)** Wearable, wireless EMG and accelerometer acquisition device. **b)** 13 gesture classes, including rest + 12 single degree-of-freedom movements. **c)** 8 static limb position contexts.

I collected a dataset from 5 subjects (2 male and 3 female) performing 13 gestures in 8 limb positions while measuring their forearm EMG and wrist-mounted accelerometer signals. Experiments were performed in strict compliance with IRB guidelines and were approved by the Committee for Protection of Human Subjects at the University of California, Berkeley. Data were acquired using the FlexEMG device enabling compact and wireless biosignal acquisition and streaming (Figure 4.4a) [135]. For this study I utilized a 64-channel electrode array implemented on a flexible printed circuit board, as opposed to the custom screen-printed silver electrode array from Chapter 3. In collaboration with Alisha Menon and Fred Burghardt, I also added an on-board accelerometer (InvenSense MPU-6050) to measure limb position. Data from the accelerometer x-, y-, and z-axes were sampled at 1 kS/s and synchronized with the EMG data acquisition. The electrode array was wrapped around the largest section of the dominant forearm to cover the extrinsic flexor and extensor muscles involved in digit movements. The device was oriented such that the accelerometer was positioned over the ulna approximately two inches from the wrist, measuring the movements of the forearm. For this study, the FlexEMG device's in-sensor learning capabilities were not used, and raw EMG and accelerometer signals were wirelessly streamed to a laptop for offline analysis. Figure 4.4b and c show the gesture classes and limb positions used for the study. The 13 gesture classes included 12 single degree-of-freedom movements of each digit along with the rest class (no movement). Eight different limb positions were chosen based on literature to mimic ones that may be encountered in everyday use [206].

The experimental procedure consisted of one recording session per limb position, in which each gesture was performed for three repetitions. A single repetition (Figure 4.5) consisted of beginning in the default limb position (Position 0) and the rest gesture, transitioning to the directed limb position within a two-second window, performing the directed gesture within a two-second window, holding the gesture for four seconds, transitioning back to the rest gesture within a two-second window, and returning to the default limb position within a two-second window. A relaxation period of three seconds followed each repetition. Directions for gesture and limb position and timing guidance were provided to the subject through a custom graphical user interface, which also logged the wirelessly streamed data. For analysis, only the central four-second steady-state period of the gesture performance was used. Each period was divided into 50 ms windows for feature extraction of the average x-, y-, and z-axis acceleration and mean absolute value of each of the 64 EMG channels.

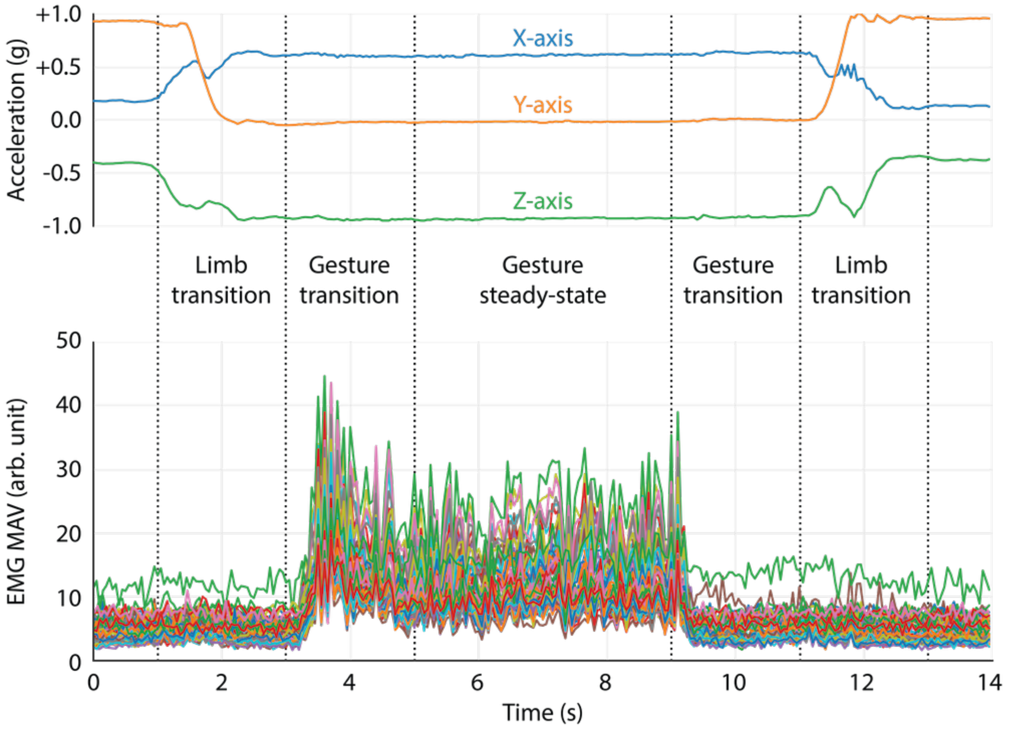


Figure 4.5: **Single data collection trial.** 64-channel EMG mean absolute value (MAV) features and 3-axis accelerometer features from an example repetition of middle finger extension in limb position 3, with annotated timing directions.

4.2.2 Interference between contexts in superposition

For all of the methods involving HD computing in this chapter, I implemented the same EMG projection algorithm as described in Section 3.3.2, except with the standard 10,000-dimensional hypervectors rather than smaller 1,000-bit ones used for FPGA implementation. As a refresher for the reader, I represent each electrode channel using an orthogonal 10,000-D bipolar hypervector. First, spatial encoding is performed by computing a bipolarized weighted sum of these hypervectors using the extracted feature values from each channel as weights. Temporal information is then encoded using a permute and multiply operation across five consecutive spatially encoded hypervectors. Thus, each encoded EMG hypervector represents 250 ms of EMG data, with an overlap of 200 ms.

Classification of query hypervectors was performed through nearest neighbor search among class prototypes. Prototypes were calculated by superimposing training examples from each class through element-wise majority, and similarity between prototypes and queries was measured using Hamming distance. To demonstrate why improvements were still needed, I implemented simple direct superposition of training examples from multiple limb positions for building class prototypes. Figure 4.6a describes this process in two steps, where context-specific prototypes were first formed through accumulation of training examples, and then generalized prototypes were formed by accumulating the context-specific associative memories. For bipolarized hypervectors, bipolarization would occur after the second accumulation. The same experiment was repeated for each of the subjects, with 5-fold cross-validation used to generate training (4 folds) and testing (1 fold) datasets. Figure 4.6b shows the classification accuracy degradation as more and more limb position contexts are stored in superposition. While average classification accuracy within a single limb position is high for all subjects, accuracy falls as more and more limb positions are stored in superposition in a single set of class prototypes.

The source of error due to direct superposition can be analyzed by inspecting the element-wise addition operation used before bipolarization, replacing Hamming similarity between two hypervectors with the dot product $(\cdot)$. The prototype stored in the associative memory for gesture g can be calculated as $AM[g] = \sum_p AM_p[g]$, where $AM_p[g]$ is the position-specific prototype for gesture g in position p. Classifying a query hypervector X with gesture label y from limb position context c is then computed as:

$$\begin{aligned}\widehat{y} &= \underset{g}{\operatorname{argmax}}\,(X \cdot AM[g]) \\ &= \underset{g}{\operatorname{argmax}}\left(X \cdot AM_c[g] + \sum_{p \neq c} X \cdot AM_p[g]\right)\end{aligned} \tag{4.1}$$

We split each dot product into a position-specific classification term and an error term from interference with other contexts. This error typically is not small, and it depends on the similarity between prototypes from different gestures and different positions. In the next two sections, we describe two methods for improving classification. Section 4.3 describes a method that, after being initialized with prototypes computed using direct superposition,

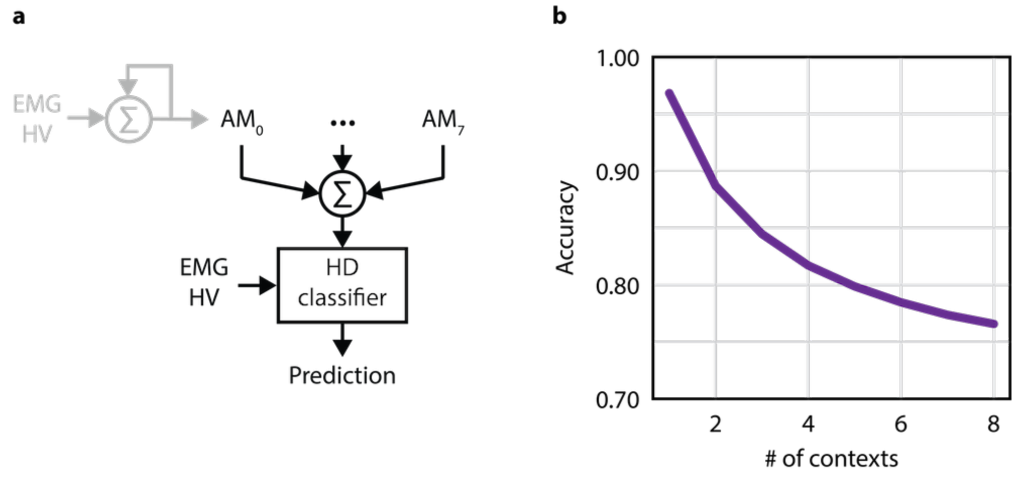


Figure 4.6: **Direct superposition of EMG training examples in multiple contexts.** **a)** Architecture describing the superposition process in two steps. For each context, training hypervectors from each gesture class are superimposed to form associative memories with context-specific class prototypes. Then, multiple context-specific associative memories can be superimposed together to form a generalized model for inference. **b)** Classification accuracy using direct superposition, measured when training and testing a single HD model in different sized subsets of limb position contexts. For each sized subset, all possible combinations of limb position contexts are taken. Accuracy is shown as an average over these different subsets and over 5 subject-specific experiments.

iteratively "corrects" and heuristically optimizes the prototypes for better class separateion. Section 4.4 then describes methods for incorporating accelerometer signals for context encoding to minimize the interference error by orthogonalizing prototypes from different limb positions.

4.3 HD prototype optimization

In prototype optimization for HD computing, the goal is to generate class prototypes through methods other than computing class centroids in order to improve classification performance. In this section, we first introduce learning vector quantization (LVQ), a prototype-based classification algorithm attempting to define optimal boundaries within the data and maximize classification margin. We then turn to a heuristic approach that is more amenable to HD computing with bipolar hypervectors.

4.3.1 Generalized learning vector quantization

Learning vector quantization, like HD computing, is a prototype-based classification algorithm [216]. The primary difference is that with HD computing, a simple superposition is used to produce prototypes, whereas for LVQ, prototypes are created through an optimization process with a cost function. The other difference is that traditional LVQ excels with floating point features and Euclidean distance, whereas HD computing is more amenable to operations in a binary or bipolar domain. Still optimization has its advantages: it attempts to actually learn boundaries between classes rather than simply trying to form class representations, and through this it optimizes classification performance.

A review of LVQ algorithms shows that they can roughly separated into two different variants: 1) margin-based, where a distance-based cost function incorporates information about the decision boundary and is to be optimized, and 2) heuristic, in which prototypes are updated through measuring the amount of error they produce when performing classification [216]. Here, we describe generalized LVQ (GLVQ), a margin-based algorithm [217]. In GLVQ, prototypes may be initialized at random or based on the data, e.g., using the class centroids. After initialization, the k class prototypes w_g for $g \in 1, 2, ...k$ are used to evaluate the cost function:

$$J = \sum_{i=1}^{n} \phi\left(\frac{d(x_i, w_{correct}) - d(x_i, w_{wrong})}{d(x_i, w_{correct}) + d(x_i, w_{wrong})}\right) \tag{4.2}$$

where x_i is an individual training example, n is the total number of training examples, $d(a, b)$ is a distance function between vectors a and b, $w_{correct}$ is the prototype representing the correct class for example x_i, w_{wrong} is the nearest incorrect prototype, and $\phi(\cdot)$ is a monotonically increasing non-linear function. This cost-function is differentiable and can thus be minimized using gradient descent.

For gesture recognition experiments, I applied GLVQ optimization to the projected EMG hypervectors and compared the performance to prototypes generated through direct superposition. Because the cost function must include a differentiable distance function, I selected Euclidean distance for the optimization process even though all of the training examples were bipolar hypervectors. Optimization also produced prototypes with floating-point values, so I implemented three different types of inference for evaluating the classification accuracy. The first kept the floating-point prototypes and used Euclidean distance as the classification distance metric, keeping to what was used in the cost function. To make the inference more similar to HD classification, I also replaced the Euclidean distance with cosine distance. Finally, to make the inference process identical to HD classification, I bipolarized the optimized prototypes and used Hamming distance as the metric for classification.

Table 4.1 shows the classification accuracy results for a single subject in all 8 limb positions when comparing the GLVQ based methods to prototypes computed using direct superposition. For this particular subject, the standard HD computing method of direct superposition was only able to achieve 75.96% classification accuracy, a major degradation when compared to the average 98.86% within-context classification for each limb position. GLVQ using floating-point prototypes and Euclidean distance for classification was able to

Prototype type	Distance metric	Accuracy
Bipolarized direct superposition	Hamming distance	75.96%
Floating-point GLVQ	Euclidean distance	95.61%
Floating-point GLVQ	Cosine distance	84.93%
Bipolarized GLVQ	Hamming distance	81.39%

Table 4.1: GLVQ prototype optimization accuracy.

mostly recover the classification accuracy to 95.61%, demonstrating the ability for GLVQ to define much better separating planes in the HD space. However, this performance depended a lot on the way prototypes were represented. If we first used cosine distance rather than Euclidean distance for the nearest-neighbor search, this dropped accuracy to 84.93%. Further simplifying the inference process by bipolarizing the prototypes and using Hamming distance to compare queries reduced classification accuracy to 81.39%. This shows that while GLVQ can find better prototypes, it also requires a much more complex inference computation and larger storage for prototypes to be effectual.

4.3.2 Adaptive HD training and LVQ for bipolar hypervectors

To find a prototype optimization method that is better suited to inference using Hamming distance, I turned to the heuristic LVQ2.1 algorithm [216] and adaptive training with HD computing [218]. The basic concept behind LVQ2.1 is to iteratively nudge the closest correct prototype and the closest incorrect prototype for each training example. The closest correct prototype is nudged in the direction towards the training example to make it more similar, with $w_{correct} \leftarrow w_{correct} + \alpha \cdot (x - w_{correct})$ where α is the learning rate. Similarly, the closest incorrect prototype is updated with $w_{wrong} \leftarrow w_{wrong} - \alpha \cdot (x - w_{wrong})$. Adaptive HD training takes a similar approach, except with superposition of training examples rather than "nudging" their values using the error vector. Upon accumulating a new training example onto the correct class's prototype, if that example is still misclassified, then it is subtracted from the prototype of the mistakenly selected class.

For my experiments, I extended this idea to a batch implementation of LVQ-style updates, similar to the method described in [219]. In the first iteration, the entire batch of training data is used to generate prototypes through direct superposition. Afterwards, all training examples are classified using the current prototypes. For each class, the prototype w_g for class g is updated as follows: First all of the examples that produced a false negative (belonging to class g but mistakenly classified with a different label) are collected, and their bipolarized centroid $w_{g,FN}$ is found. Similarly, the centroid $w_{g,FP}$ is found for all of the false positive examples that were incorrectly classified as gesture g. The prototype is then updated using a bitwise merge to approximate $w_g \leftarrow w_g + \alpha w_{g,FN} - \beta w_{g,FP}$. Here, α and β are learning rates determining the proportion of bits to merge from the respective centroids during the update.

Figure 4.7 shows the learning curve for this LVQ-style prototype update. After about 20 iterations, classification accuracy for the validation set stabilizes at around 93.18%. This performance is almost as good as GLVQ prototypes with Euclidean distance classification, but it requires only bipolar hypervectors for more efficient operation.

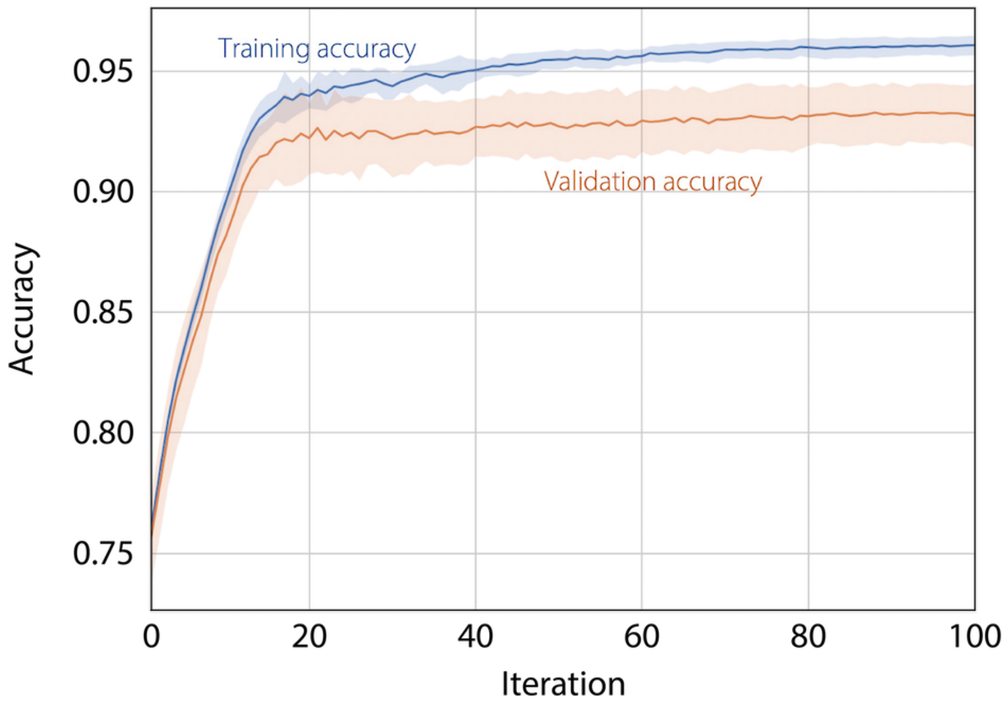


Figure 4.7: **Learning curve of heuristic LVQ for HD classification.** **a)** Architecture describing the superposition process in two steps. For each context, training hypervectors from each gesture class are superimposed to form associative memories with context-specific class prototypes. Then, multiple context-specific associative memories can be superimposed together to form a generalized model for inference. **b)** Classification accuracy using direct superposition, measured when training and testing a single HD model in different sized subsets of limb position contexts. For each sized subset, all possible combinations of limb position contexts are taken. Accuracy is shown as an average over these different subsets and over 5 subject-specific experiments.

While these optimization methods resulted in improvements over direct superposition, the iterative nature of the processes and the requirement that training data from all contexts be available during the training process makes prototype optimization unlikely to be adopted for in-sensor, incremental model training. For such situations, we must instead turn to a methodology in which superposition through incremental accumulation is the only required operation on projected hypervectors. In the next section, we discuss how this can be made possible by encoding context information with sensor fusion.

4.4 Sensor fusion for context encoding

Although the same gesture classes are used in each of the different limb positions, EMG signal variation due to context change can shift the data distribution enough that gesture recognition in multiple limb positions can be viewed as a multi-task problem. It has been shown that orthogonal, high-dimensional context vectors can be bound to neural network weights to access task-specific models stored in superposition with other models for multi-task learning with the same set of parameters [220]. A similar result has also been demonstrated for HD classification, where orthogonal hypervectors representing different tasks were used as keys to access task-specific prototypes stored in superposition [221]. These strategies could be applicable here, requiring some method of determining the limb position context during inference.

In this section, I describe a method of binding context hypervectors based on accelerometer signals we call context-based orthogonalization [168]. I present an analysis of its expected classification margin improvement, along with two architectures for projecting accelerometer-based limb position information into context hypervectors. The first, which most closely resembles dual-stage classification, requires identification of discrete limb position contexts that are each represented by an orthogonal context hypervector. Context hypervectors are selected during training and inference based on limb position labels and limb position classification using accelerometer signals, respectively. The second architecture I present directly encodes accelerometer features into context hypervectors without the need to define discrete contexts. My results show that sensor fusion for context-based orthogonalization can improve classification accuracy for my multiple limb position dataset from 76.57% to 91.66% without introducing significant implementation costs.

4.4.1 Context-based orthogonalization

Context-aware classification can be achieved using a dual-stage architecture as depicted in Figure 4.8a, which resolves the issue of interference between contexts by separating them completely. However, this method would require separate storage of context-specific parameters and scale memory footprint by the number of desired contexts. We can instead orthogonalize gesture prototypes from different limb position contexts by binding them with orthogonal context vectors prior to superposition [220, 221]. During inference, query hypervectors are also bound to their corresponding context vectors before nearest neighbor search.

Here, I show that context-based orthogonalization effectively enables dual-stage classification while still allowing for superposition of parameters and thus memory efficiency. Figure 4.8b depicts the implementation for an HD classifier. Using similar notation as in Section 4.2.2, entries in the AM are now calculated as $AM^*[g] = \sum_p CM_p \times AM_p[g]$, where CM_p is a context hypervector representing limb position p and is orthogonal to context hyptervectors

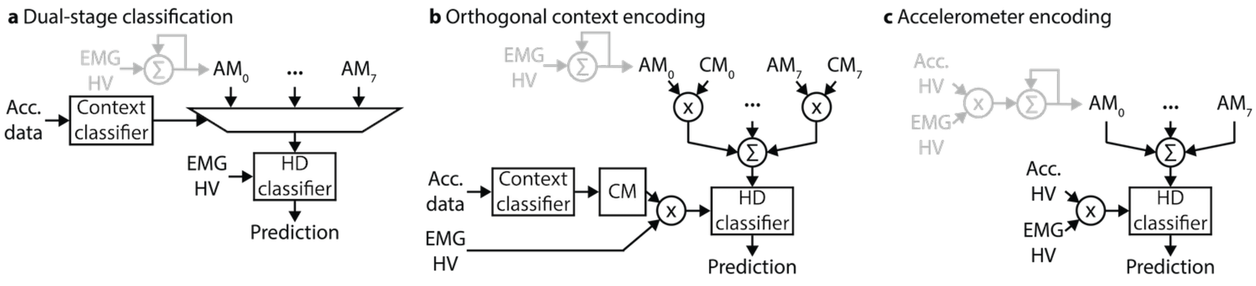


Figure 4.8: **Context-aware HD classification architectures.** **a)** Dual-stage classification with limb position classification to select position-specific prototypes. Accelerometer (acc.) data is used to classify the limb position context and choose from among the context-specific models (AM_0 through AM_7). **b)** Context-based orthogonalization using classified limb position to enable superposition of prototypes. Orthogonal context hypervectors (CM_0 through CM_7) are bound to EMG hypervectors during training and inference. **c)** Direct encoding of accelerometer signals as context vectors without limb position classification. Accelerometer context vectors are calculated on an example-by-example basis.

representing other positions. Classifying a query is now computed as:

$$\begin{aligned}\widehat{y} &= \operatorname*{argmax}_g \left(CM_c \times X \cdot AM^*[g]\right) \\ &= \operatorname*{argmax}_g \Bigg(CM_c \times CM_c \times X \cdot AM_c[g] \quad + \\ &\qquad \sum_{p \neq c} CM_c \times CM_p \times X \cdot AM_p[g]\Bigg) \\ &= \operatorname*{argmax}_g \left(X \cdot AM_c[g] + \epsilon\right)\end{aligned} \tag{4.3}$$

Because binding a bipolar hypervector with itself results in the identity vector, the first term of the dot product is again a position-specific term. Binding two random hypervectors with each other results in a new, orthogonal hypervector, diminishing the dot product error term.

4.4.2 Classification margin analysis

A classification margin can be defined by viewing the task as a problem of noisy element retrieval from the associative memory, with error due to both sample-by-sample variation and prototype superposition. Since sample-by-sample variation should be zero-mean, we neglect it in defining retrieval error for class y in context c:

$$d_{retrieve} = d\left(CM_c \times AM_c[y], \sum_p CM_p \times AM_p[y]\right) \tag{4.4}$$

where $d(a, b)$ is the Hamming distance function. We can compare this with the distance to the nearest incorrect class:

$$d_{wrong} = d\left(CM_c \times AM_c[y], \sum_p CM_p \times AM_p[g \neq y]\right) \tag{4.5}$$

to approximate the classification margin:

$$m = \left(d_{wrong} - d_{retrieve}\right) / \left(d_{wrong} + d_{retrieve}\right) \tag{4.6}$$

For the baseline case of direct superposition, we can just set all of the CM_p context vectors to be identical. I simulated a general classification task using randomly generated prototypes with average pairwise distance of d_0, regardless of class or context. Figure 4.9a shows the retrieval error and incorrect classification distance when superimposing 11 contexts with varying d_0, both for direct superposition and with context encoding. While orthogonal context encoding increases retrieval error to a constant value, it also increases incorrect classification distances by a larger amount, leading to a larger classification margin as shown in Figure 4.9b. This improvement in margin is shown for varying odd numbers of contexts in Figure 4.9c. Even numbers of contexts perform identically to the next higher odd number of contexts since the random tie-breaking required for superposition is equivalent to adding in a new, orthogonal context. There is no improvement in margin over direct superposition when there are three or fewer contexts, or in extreme cases where all prototypes are identical or orthogonal. Otherwise, orthogonal context encoding always improves margin, with the maximum improvement occurring with 11 contexts and $d_0 \approx 0.3$.

4.4.3 Sensor Fusion of Accelerometer Signals

I explored two different ways to generate context vectors using accelerometer features. First, we can perform dual-stage classification and train a first stage classifier on accelerometer signals to select between limb position context hypervectors. We can classify limb positions using a simple linear SVM, and a context memory can be created with a randomly generated, orthogonal vector representing each limb position. Figure 4.10a shows the accelerometer features for different limb positions used to select between context memory hypervectors.

Alternatively, we can directly encode accelerometer signals into context hypervectors, without the need to classify limb position (Figure 4.8d). Accelerometer signals are first quantized between $+1g$ and $-1g$ into discrete levels which can be represented by hypervectors from a continuous item memory (CIM) (Figure 4.10b). A separate memory is used for each of three accelerometer axes, and encoded quantization levels are then bound together to form a single context hypervector. The CIM preserves relationships between quantization levels, representing extreme values ($\pm 1g$) with two hypervectors that are d_{max} Hamming distance apart and representing intermediate values as a mixture of these hypervectors. This ensures similarity between examples from the same limb position, while still differentiating between hypervectors representing different positions.

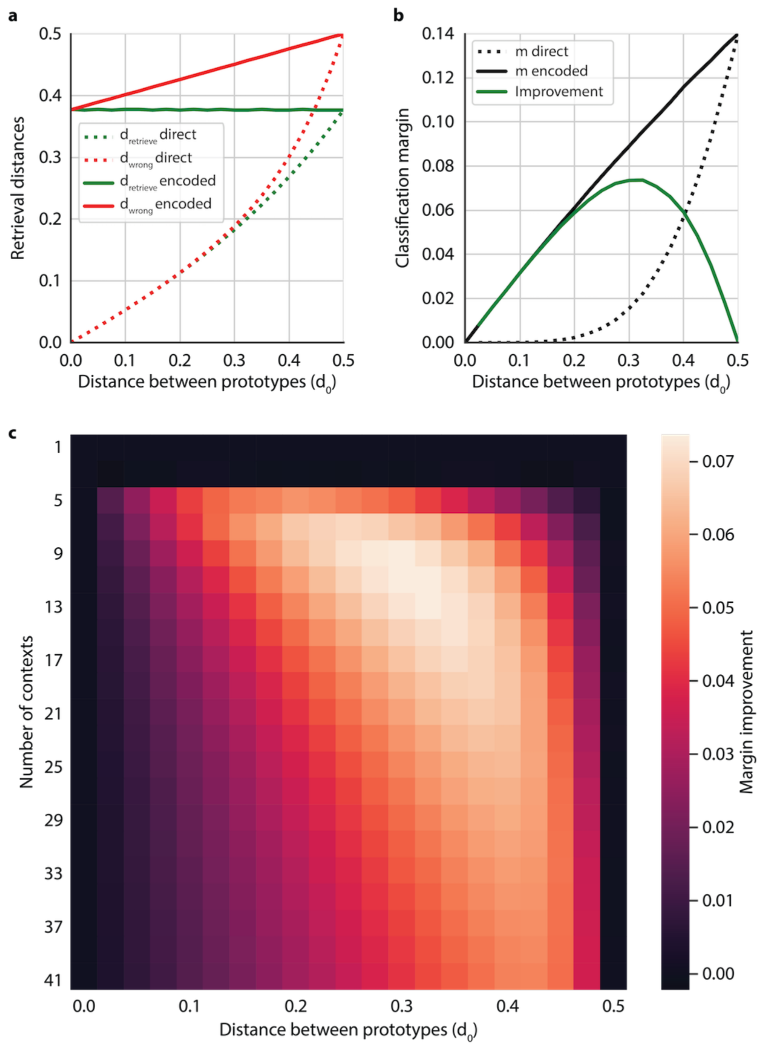


Figure 4.9: **Improvement in classification margin using orthogonal context encoding. a)** Retrieval error (green) and incorrect classification distance (red) for 11 contexts. Dashed lines are with direct superposition, and solid lines are with context-based orthogonalization. **b)** Classification margins and margin improvement due to orthogonal context encoding for 11 contexts. **c)** Margin improvement as a function of both the number of contexts and distance between classes.

4.4.4 Experiments

To provide baseline comparisons with context-based orthogonalization methods, I implemented two strategies from prior art for multi-limb position classifiers. The first was a true dual-stage classifier [202] (Figure 4.8a), where I used an SVM to classify accelerometer signals and choose an associative memory containing position-specific prototypes. The second

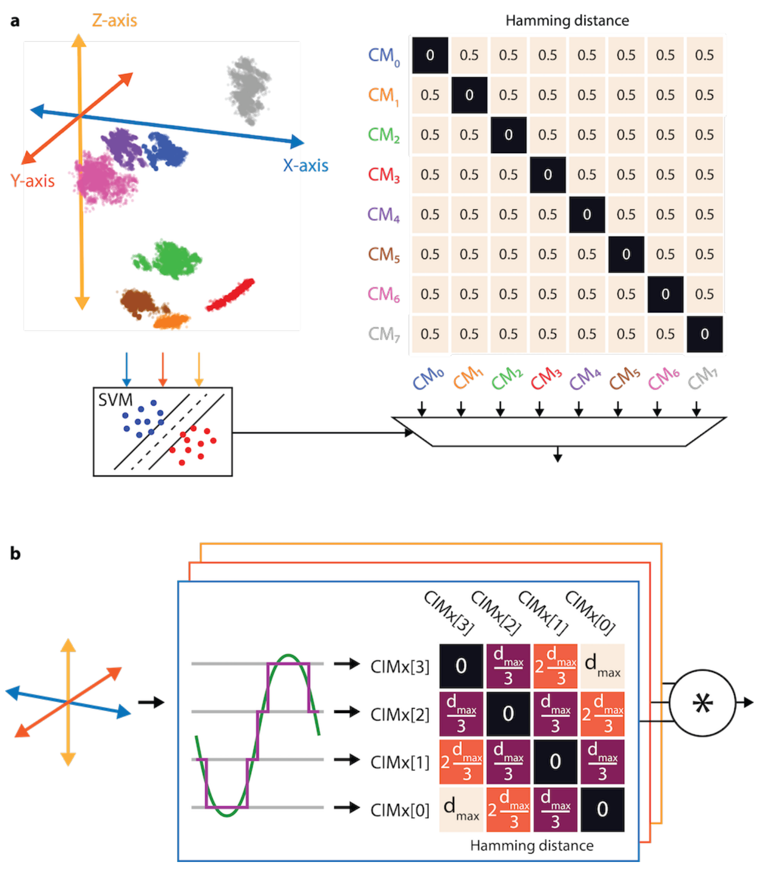


Figure 4.10: Methods for projecting accelerometer signals into hypervectors. **a)** Selection of a context vector from an orthogonal context memory based on classified limb position contexts using SVM. **b)** Using a continuous item memory to represent the quantized signal from each accelerometer axis and binding the result.

	Dual-stage classification	**Direct superposition**	**Context-based orthogonalization**	**Accelerometer CIM encoding**
Accuracy	96.56%	76.57%	91.66%	91.31%
Parameter count	104 prototypes 50 support vectors	13 prototypes	13 prototypes 50 support vectors	13 prototypes
Memory	1020.3 kb	127.0 kb	131.6 kb	126.0 kb
	Adapted from [202]	Adapted from [135]	This work	This work

Table 4.2: Comparison of limb-position aware gesture recognition approaches.

was direct superposition of each of the limb position-specific associative memories without any context information [135] (Figure 4.6a). Classification accuracy for all experiments was calculated using 10-fold cross-validation of the dataset.

The first two columns in Table 4.2 show the results for these baseline strategies. Dual-stage classification, as expected, achieved the best-case scenario classification accuracy of 96.56% across all limb positions. However, this method has the major drawback of requiring separate EMG and limb position classification modules, as well as $> 8\times$ the required parameter storage compared to a single limb position model. On the other hand, the direct superposition model is identical to a limb position-specific model in terms of implementation, but it results in an accuracy degradation of 19.99%.

As points of comparison for memory footprint, I applied to our dataset a multi-class SVM classifier with RBF kernel as well as an LDA classifer (Python scikit-learn). Limb position-specific SVM models required on average 287 support vectors consisting of the same 320 floating point features used by our HD classifier to achieve comparable accuracy. For dual-stage classification of all 8 limb positions, this resulted in an SVM parameter memory of approximately 23 Mb, more than $22\times$ the size of a dual-stage HD classification model (≈ 1 Mb). If instead I trained a single SVM model on all 8 limb positions, only 362 total support vectors would be necessary. This decreases parameter memory to 3.5 Mb, which is still larger than our HD classifier formulations, and it also requires that all applicable contexts are known and accounted for at the initial training time. In contrast, HD classifier updates can be performed at any point through simple superposition of prototype hypervectors.

For LDA, individual model size is the same whether trained with a single context or on all contexts. In both cases, a 320 element mean vector for each gesture and a 320×320 element covariance matrix must be stored, which totals to 3.25 Mb per model and 26 Mb for a dual-stage classification with 8 LDA models.

Orthogonal context encoding using a limb position classifier (Figure 4.8b) provided a dramatic improvement in classification accuracy to 93.37%, while only increasing memory footprint by 3.7% compared to the direct superposition case and limb position-specific models (Table 4.2). The marginal increase in parameter memory was for storage of SVM parameters. For our tests, 99.99% accuracy in limb position classification could be achieved while using only up to 50 stored support vectors consisting of 32-bit floating point accelerometer features. Including the HD classifier prototypes, which require 10,000 bits for each of the 13 classes, the total parameter memory footprint is 131.6 kb.

Although light weight in terms of memory requirement, the use of a limb position SVM does still require implementation of the decision function, adding to the model complexity. Additionally, the SVM must be trained using discrete, labeled limb positions decided upon before deployment. This limitation motivates direct encoding of accelerometer signals to create an arbitrary number of context vectors.

I used a genetic algorithm [222] to determine optimal parameters for encoding accelerometer features into context vectors (Figure 4.8c). Parameters for each subject were chosen individually, and the number of quantization levels as well as the distance d_{max} between extreme values in the CIM were allowed to be different for each accelerometer axis. CIM-based accelerometer encoding was able to match context-based orthogonalization in classification performance with an accuracy of 91.31% while removing the required overhead for a limb position classifier (Table 4.2). Generation of CIM hypervectors can be achieved through re-use of components already existing in the main EMG projection architecture, adding very little to implementation complexity.

These results show that by encoding accelerometer signals to include information about the limb position, we can greatly improve classification accuracy of hand gestures in multiple limb positions. However, this does not solve the problem of large initial training efforts. In the next section, we detail how the HD classification model performs when trained incrementally so that a single limb position may be learned one at a time, reducing the initial effort of building the classification model.

4.5 Incremental learning with HD classifiers

Up to this point, I have described real-time, in-sensor gesture classification and simple model updates using HD computing [135], as well as accelerometer encoding for incorporation of context information to improve classification in multiple limb position contexts [168]. In this section, I combine ideas from these works to describe an incremental learning framework for HD computing. I analyze the various methods of maintaining and updating prototypes in an incremental fashion and how they impact classification accuracy and memory footprint. These methods include how accelerometer signals should be encoded to provide limb position context information, how new training hypervectors should be superimposed into existing prototypes, and also how many prototypes per class are allowed.

A key result I demonstrate is that performing the majority vote for incremental superposition in two stages not only reduces the memory required to store intermediate values between incremental training steps, but it also improves the accuracy across all seen limb position contexts. I then proceed to show how various configurations can be chosen to trade-off classification accuracy and memory footprint, ranging from the smallest model with 85.71% accuracy and 264 kb of required memory to the largest with 97.15% accuracy and 1.125 Mb of memory footprint. Finally, in comparing incrementally trained HD classifiers with traditional machine learning models, I show that HD computing can achieve comparable classification accuracies while requiring an order of magnitude smaller memory allocation, making it ideal for implementation in devices where resources are limited.

4.5.1 Incremental learning setup

For my incremental learning experiment, I considered a scenario in which we train on a sequence of m incremental training datasets $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_m$ where each set consists of $\mathcal{S}_i = \{(x_{i,j}, y_{i,j})\,|x_{i,j} \in X, y_{i,j} \in \{1,2,...,k\}, 1 \leq j \leq n_i\}$ with $x_{i,j}$ and $y_{i,j}$ being training examples and labels, X being the feature space of raw features or projected hypervectors, k being the number of classes, and n_i being the number of training examples in each dataset $\mathcal{S}_i$. Each incremental dataset consists of context-specific data, modeling a scenario in which new limb positions are incrementally learned over time. We can represent the initial classifier trained on $\mathcal{S}_1$ as CLF_1. Updated classifiers after each subsequent incremental training step are $CLF_i = f_{train}\left(\mathcal{S}_i, CLF_{i-1}\right), 1 < i \leq m$ where each updated classifier only has access to the latest dataset and the previously trained classifier parameters.

For each new incremental training dataset, we can compute candidate prototypes representing that specific context. Therefore, I use the terms "candidate prototype" and "context-specific prototype" interchangeably. For HD computing, where incremental training and batch training can differ are:

1. How candidate prototypes are incorporated into the model's prototype memory, and
2. What representations of stored prototypes (or previous training examples) are available

We can either superimpose a candidate prototype with an existing stored prototype, or, if our memory budget permits, we can store it as a new prototype such that multiple prototypes represent the same gesture. I first describe various algorithmic design choices one can make for incrementally superimposing new candidates with stored prototypes, beginning with a method that produces an identical model as with batch training, and introducing configurations with increasing optimization and approximation. I then describe the overall algorithm with the ability to either superimpose candidate prototypes or store them separately for a trade-off between memory footprint and classification performance.

4.5.2 Incremental prototype superposition

If all previous training examples are retained in memory as model "parameters", then each incremental step can consist of batch training with all data seen until this point, $\mathcal{S} = \mathcal{S}_1 \cup \mathcal{S}_2 \cup ... \cup \mathcal{S}_i$ (Figure 4.3). This storage would require $D \times n$ bits in memory, where $n = \sum_i n_i$ is the combined size of all incremental datasets. However, we ultimately only require the class centroids, and thus we only need to store their accumulated sums from each class (Figure 4.11a). I will refer to this method as *example accumulation*, and, assuming class-balanced training examples, this reduces the required memory footprint to be $D \times k \times \left(\lfloor \log_2\left(\frac{n}{k}+1\right)\rfloor + 1\right)$ bits, where $\lfloor \log_2(N+1)\rfloor + 1$ is the memory required to accurately sum N bipolar bits. For our dataset with 240 examples per class per context ($n = 240 \times m \times k = 24,960$), this amounts to a reduction from $10,000 \times 24,960 \approx 238$ Mb required to store all training examples down to $10,000 \times 13 \times 11 \approx 1.36$ Mb to only store their sums.

The first optimization we can make is to only store and accumulate bipolarized candidate prototypes at each step (Figure 4.11b). *Prototype accumulation* requires fewer bits per element than example accumulation, with overall memory footprint being $D \times k \times (\lfloor \log_2(m+1)\rfloor + 1)$. In my case with 13 gestures and 8 limb positions, we now only require $10,000 \times 13 \times 4 \approx 508$ kb of memory.

The benefit of prototype accumulation is two-fold: not only does it require less memory than example accumulation, but it also potentially improves classification performance. Example accumulation from multiple contexts with differing class cluster spread creates prototypes with unbalanced representation of the different contexts. By cluster spread, I refer

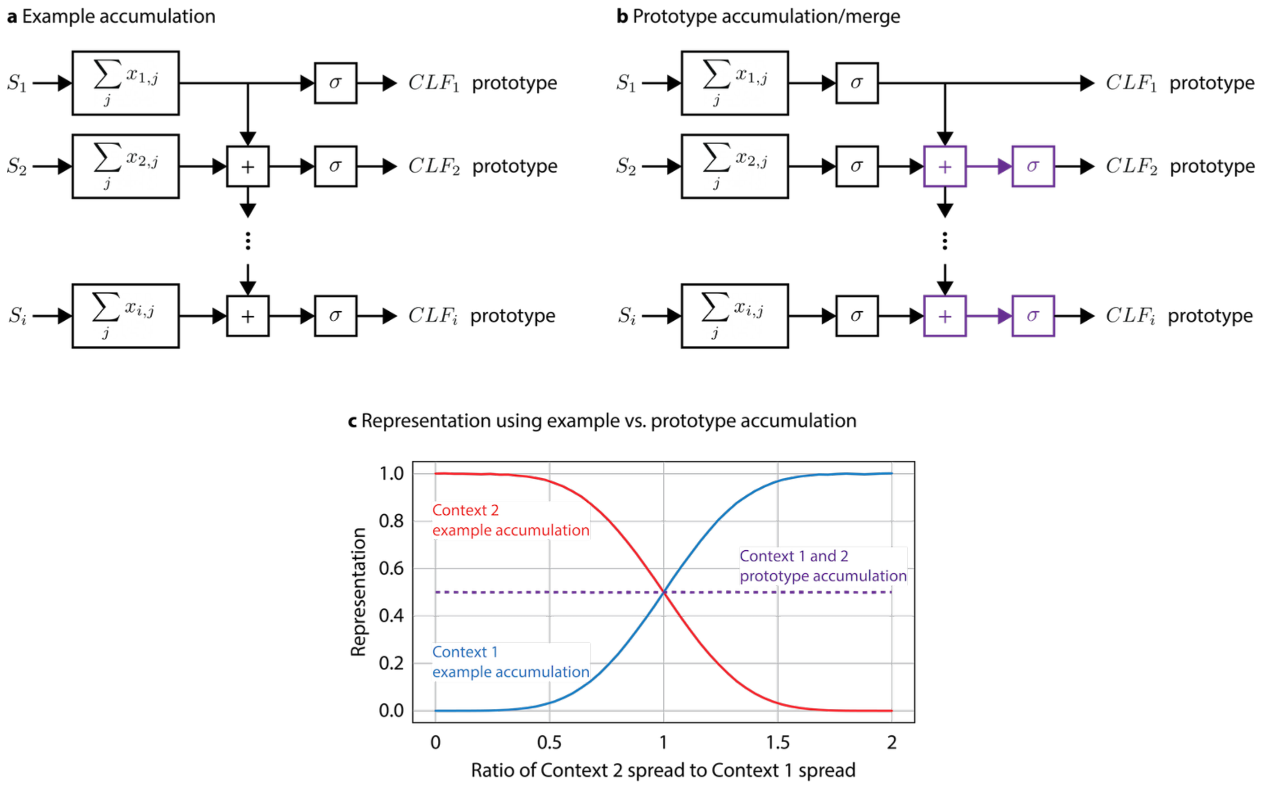


Figure 4.11: **HD classifier incremental superposition configurations.** Each row of operations in **a)** and **b)** shows updating a class prototype during an incremental training step. Vertical arrows show values that must be preserved for the next incremental training iteration. **a)** Example accumulation of all previously seen training examples. **b)** Accumulation of bipolarized prototypes from each of the sequential training datasets requires a lower bit-width representation in memory. Prototypes are accumulated using integer addition (shown) or through bit-wise merge (replacing the purple blocks). **c)** Prototype representation for two orthogonal contexts with varying cluster spread. Context 1 spread is held constant at 0.25 average pairwise Hamming distance, while Context 2 spread varies between 0 and 0.5. The x axis is the ratio of the two cluster spreads. Solid lines show representation for example accumulation, and dotted lines show representation for prototype accumulation.

to the average similarity between HVs belonging to the same class. In superposition using majority vote, classes with low spread overwhelm more spread-out classes, leading to a prototype that is more similar to the tighter class's centroid and distant from the noisier class's centroid. This results in especially bad performance when encoding limb position information with orthogonal context HVs, as the prototype may end up completely orthogonal to certain contexts it is supposed to represent. On the other hand, prototype accumulation ensures even context representation, as only a single HV from each context is included in the sum.

I simulated accumulating two clusters of data, varying the ration between cluster spread of each cluster (Figure 4.11c). The resulting prototype was always more similar to, and thus more representative of, the cluster with lower spread when using example accumulation. In contrast, prototype accumulation resulted in balanced representation.

Beyond this, we can approximate prototype accumulation to further reduce model parameter memory size. Rather than storing integer values, we instead only need 1 bit of memory per prototype element if we use bit-wise merge to approximate accumulation (Figure 4.11b). The ith candidate prototype can be merged into the stored prototype by randomly replacing $1/i$ of the stored bits with bits from the candidate. This "learning rate" of $1/i$ ensures that all seen candidate prototypes are approximately equally represented [192]. While this results in noisier prototypes, we reduce the parameter memory for our scenario to $10,000 \times 13 \approx 127$ kb. A comparative summary of these configurations is given in Table 4.3.

Configuration	**Incremental superposition operations**	**Parameter memory calculation**	**Parameter memory**[a]
Example accumulation	$\sigma\left(\sum_{l=1}^{i-1}\left(\sum_{j=1}^{n_l} x_{l,j}\right)+\sum_{j=1}^{n_i} x_{i,j}\right)$	$D \times k \times \left(\lfloor\log_2\left(\frac{n}{k}+1\right)\rfloor+1\right)$	1.36 Mb
Prototype accumulation	$\sigma\left(\sum_{l=1}^{i-1}\sigma\left(\sum_{j=1}^{n_l} x_{l,j}\right)+\sigma\left(\sum_{j=1}^{n_i} x_{i,j}\right)\right)$	$D \times k \times \left(\lfloor\log_2\left(m+1\right)\rfloor+1\right)$	508 kb
Prototype merge	$\text{merge}\left(\text{merge}_{l=1}^{i-1}\left(\sigma\left(\sum_{j=1}^{n_l} x_{l,j}\right)\right), \sigma\left(\sum_{j=1}^{n_i} x_{i,j}\right)\right)$	$D \times k$	127 kb

[a] $D = 10,000; k = 13; n = 24,960; m = 8; 1\text{ Mb} = 1024\text{ kb} = 1024^2\text{ b}$

Table 4.3: Incremental hypervector superposition configurations

4.5.3 HD computing incremental learning algorithm

The alternative to superimposing a candidate prototype is inserting it as a *separate* prototype representing the same class. Interference caused by different contexts in superposition is eliminated, and the method is akin to using an ensemble of classifiers. Whether a candidate prototype is kept separate must be consistent across all gesture classes, requiring an additional $D \times k$ bits of parameter memory. This decision can be based on a memory budget for the number of separated contexts allowed, m_{sep}. Before the budget is reached, the number of prototypes per class grows with the number of incremental training steps. After meeting the budget, one context will be selected as the target context for future superposition, either at random or based on its age. In subsequent steps, either new prototypes or an existing separate prototype will be superimposed onto the target prototype. The overall HD incremental learning algorithm is described in Algorithm 1.

Algorithm 1: HD classifier incremental training

```
Input: Memory budget of $m_{sep}$ separate prototypes per class and one superimposed prototype
       representing $m_{sup}$ contexts per class; incremental superposition configuration; sequence of
       incremental training datasets $\mathcal{S}_1, \mathcal{S}_2, ..., \mathcal{S}_m$
Result: Incrementally trained HD classification model
1  for $\mathcal{S}_i, i \in 1, 2, ..., m_{sep} + m_{sup}$ do
2      for $g \in 1, 2, ..., k$ do
3          Compute candidate prototype for $x_{i,j}, y_{i,j} \in \mathcal{S}_i \forall y_{i,j} == g$
4      end
5      if $i \leq m_{sep}$ then
6          Store candidate prototypes as new AM prototypes
7      else
8          Randomly select between new context and stored separate contexts
9          if $i = m_{sep} + 1$ then
10             Set aside selected context prototypes as target prototypes
11         else
12             Accumulate selected context prototypes onto target prototypes using incremental
                 superposition configuration
13         end
14     end
15 end
```

4.5.4 Memory allocation

In addition to parameter memory, a local scratch memory must also be maintained for storing intermediate values during feature projection to hypervectors and incremental training operations. Pseudo-random, orthogonal item memory hypervectors can be generated sequentially using a rule 90 cellular automaton with hardwired seeds and connections [192, 223]. For spatial encoding, a single 32-bit floating point accumulator can be used to generate the weighted sum for each element, which are all independent of each other. D bits are required for storage of the bipolarized spatial hypervector. For temporal encoding, we must store the previous $N_{temporal} - 1$ projected spatial hypervectors as well, totaling a memory requirement of $D \times N_{temporal} + 32$ bits for projection of EMG data.

For projection and classification of accelerometer data, we can also generate continuous item memories based on hardwired logic gates and seeds from the item memory generator [192]. A single D-bit memory is required for storage and sequential binding of the three encoded accelerometer axes. If we wish to perform orthogonal encoding, we can also implement a small HD model for context classification. This model can have a reduced hypervector dimension by employing only a subset of the bits from accelerometer encoded context hypervectors, and we found that a dimension $D_{accel} = 316$ was sufficient for a context classification accuracy of 97.36%, within 0.5% of the accuracy achieved when using the full 10,000-D hypervectors.

For computation of candidate prototypes, incoming projected hypervectors can be accumulated using a D-dimensional array of 8-bit saturating up/down counters for 240 training examples per class in each incremental training dataset. This function is shared by the parameter storage memory in the example accumulation configuration, but a separate scratch memory is required when using prototype accumulation or merge, or when multiple separate prototypes per class are allowed.

A summary of the memory footprints for the various HD classifier incremental learning configurations is given in Table 4.4. In building a configuration, an encoding method (columns) must be chosen for incorporating context information, and an incremental superposition method (rows) must be chosen to update superimposed prototypes. A list of hyperparameters is given, along with values that have been implemented for hand gesture recognition. In particular, selection of m_{sep}, the budget for separate prototypes per class plays a large role in both accuracy and memory footprint.

4.5.5 Comparison methods

As comparisons to the HD classifier, I incrementally trained two traditional machine learning algorithms—support vector machine (SVM) and linear discriminant analysis (LDA)—commonly used for EMG-based gesture recognition. I used the last $N_{temporal} = 5$ MAV features for each channel along with the 3 accelerometer axis features to build 323-element input feature vectors. In addition, the entire dataset was standardized for a mean of 0 and standard deviation of 1 when using SVM. I chose to compare models based on accuracy and memory footprint only, as computation time or complexity comparisons would depend on the exact implementations of each algorithm. For gesture recognition, classification throughput is low (e.g., 20 classifications per second) and data windows are long (e.g., 250 ms), so algorithm latency is a relatively small portion of the overall system latency. Incorporating hardware re-use and time multiplexing can make it especially difficult to make fair comparisons.

I chose a linear SVM classifier which required no kernel function evaluation and produced high classification accuracy results with our dataset. I tuned the regularization parameter C using 5-fold cross-validation in a batch setting for each subject. The optimal value $C = 0.1$ offered the best classification performance while reducing the number of support vectors, thus reducing memory footprint. Although storing individual support vectors is not required for SVM inference, it is still needed for updating incrementally trained SVM models.

		Encoding configuration		
		None	**Accelerometer**	**Orthogonal**
Superposition configuration	**Example accum.**	$D \times N_{temporal} + 32$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\lfloor \log_\mathbf{2}\left(\frac{\mathbf{n_m m_{sup}}}{\mathbf{k}} + \mathbf{1}\right)\rfloor + \mathbf{1} + \mathbf{m_{sep}}\right)$	$D \times N_{temporal} + 32$ $+\mathbf{D}$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\lfloor \log_\mathbf{2}\left(\frac{\mathbf{n_m m_{sup}}}{\mathbf{k}} + \mathbf{1}\right)\rfloor + \mathbf{1} + \mathbf{m_{sep}}\right)$	$D \times N_{temporal} + 32$ $+\mathbf{D} + \mathbf{D_{accel}} \times \mathbf{m}$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\lfloor \log_\mathbf{2}\left(\frac{\mathbf{n_m m_{sup}}}{\mathbf{k}} + \mathbf{1}\right)\rfloor + \mathbf{1} + \mathbf{m_{sep}}\right)$
	Prototype accum.	$D \times N_{temporal} + 32$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\lfloor \log_\mathbf{2}\left(\mathbf{m_{sup}} + \mathbf{1}\right)\rfloor + \mathbf{1} + \mathbf{m_{sep}}\right)$	$D \times N_{temporal} + 32$ $+\mathbf{D}$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\lfloor \log_\mathbf{2}\left(\mathbf{m_{sup}} + \mathbf{1}\right)\rfloor + \mathbf{1} + \mathbf{m_{sep}}\right)$	$D \times N_{temporal} + 32$ $+\mathbf{D} + \mathbf{D_{accel}} \times \mathbf{m}$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\lfloor \log_\mathbf{2}\left(\mathbf{m_{sup}} + \mathbf{1}\right)\rfloor + \mathbf{1} + \mathbf{m_{sep}}\right)$
	Prototype merge	$D \times N_{temporal} + 32$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\mathbf{1} + \mathbf{m_{sep}}\right)$	$D \times N_{temporal} + 32$ $+\mathbf{D}$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\mathbf{1} + \mathbf{m_{sep}}\right)$	$D \times N_{temporal} + 32$ $+\mathbf{D} + \mathbf{D_{accel}} \times \mathbf{m}$ $+D \times \left(\lfloor \log_2\left(\frac{n}{mk} + 1\right)\rfloor + 1\right)$ $+\mathbf{D} \times \mathbf{k} \times \left(\mathbf{1} + \mathbf{m_{sep}}\right)$

D = hypervector dimension = 10,000; $N_{temporal}$ = N-gram length = 5

n = total number of training examples = 24,960; m = total number of contexts/incremental steps = 8; k = total number of classes = 13

n_m = number of training examples per context/incremental step = $\frac{n}{m} = 3,120$

D_{accel} = hypervector dimension for accelerometer context classification = 316

m_{sup} = number of contexts expected for superposition; m_{sep} = number of contexts allowed to be stored separately

$m_{sup} + m_{sep} = m = 8$

Table 4.4: Memory footprint of various HDC incremental learning configurations

I first implemented a linear classifier trained using stochastic gradient descent (SGD) with hinge loss and L2 penalty, fitting the parameters for a linear SVM. This method did not require storage of previous training examples, instead updating model weights through a few iterations of gradient descent over each new incremental training dataset. Within each limb position context, data from all gesture classes were made available at one time to allow for shuffling. This necessitated a large scratch memory during each training step.

Next, I explored methods for augmenting and curating datasets for retraining. After each incremental step, I reserved the model support vectors to augment the next incremental training dataset [224]. Each new dataset could consist of either all gestures or a single gesture in a new context. Fewer total steps were required when updating with all gestures, but this relied on a larger scratch memory for temporarily storing the data. Because the overall training effort and the resulting number of support vectors grow with the size of the training dataset [215], I also explored methods to subsample from the new datasets. The first method subsampled training examples which produced errors when tested on the existing model, based on the assumption that these examples were most likely to become new support vectors [225]. I further reduced the number of examples by also randomly subsampling a percentage of the training sets, trading off generalization for reduced memory.

I then trained an ensemble of SVM models, with one model for each context. I still enabled gesture-by-gesture training for each context-specific model, as well as training set reduction by randomly subsampling. Memory footprints for the incrementally trained SVM models were measured as the number of 323×32-bit floating point support vectors retained the next step. We also estimated the training scratch memory as the size of the subsampled training dataset used. This underestimates the memory requirement, as we didn't account for data standardization or additional memory for solving the SVM quadratic programming problem and computing coefficients for candidate support vectors.

For the LDA classifier, I first considered online updates to compute model parameters, which consisted of 323×32-bit mean feature vectors for each class and an overall $323 \times 323 \times 32$-bit covariance matrix. Model parameters could be updated on a sample-by-sample basis in an exact manner, requiring only the previous parameters and scratch memory to compute the update to the covariance matrix [226]. I also compared using an ensemble of context-specific LDA classifiers, which requires the same incremental training effort, but m-times the parameter storage memory.

4.5.6 Experiments

Context-specific classifiers

I first batch trained and tested separate, context-specific classifiers to again demonstrate the necessity of incremental learning, including for the comparison SVM and LDA models. Using 5-fold cross-validation, I tested each context-specific model with data from the same context ("within context"), as well as data from all other contexts ("across contexts"). Accuracy results for the HD, SVM, and LDA classifiers are shown in Figure 4.12. All models were

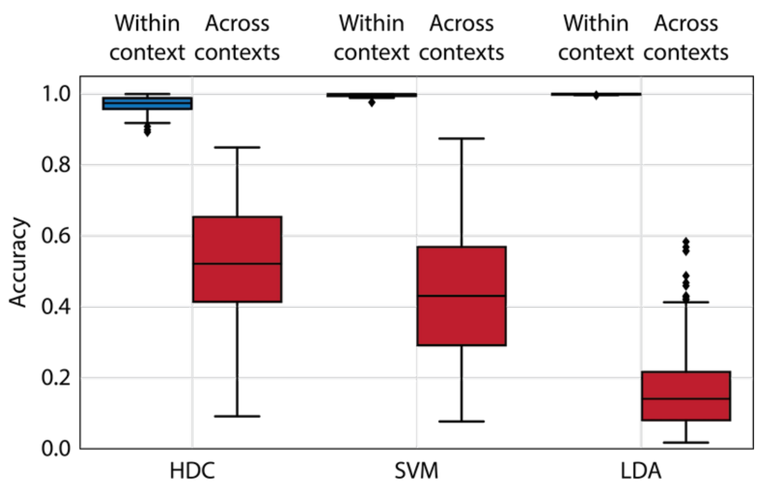


Figure 4.12: **Classification accuracy within and across limb position contexts.** Batch trained context-specific models were tested in the same context (blue) as well as in other contexts (red). Boxes show the quartiles, whiskers show the range, and individual diamonds are outliers.

able to perform well within context, with LDA being especially consistent and accurate. However, there were significant accuracy degradations when training and testing all models in different contexts, thus requiring updates for generalizable performance.

Comparison of incremental superposition methods

Figure 4.13 demonstrates the ability of HD classifiers to learn different numbers of limb position contexts through incremental superposition. For each value of m_{sup}, a model with one prototype per class was incrementally trained using all possible combinations of m_{sup} context-specific datasets. As a reference, I was able to achieve 98.95% accuracy with SVM and 93.78% accuracy with LDA when batch training and testing in all 8 limb positions. Accelerometer encoding and orthogonal encoding greatly improved classification accuracy for larger m_{sup}, with the best performance achieved using orthogonal encoding and prototype accumulation. This configuration leveraged both the reduced interference from orthogonalizing HVs of different contexts, as well as the more balanced representation of two-stage bipolarization. The effect of unbalanced representation can be clearly seen in the example accumulation results for accelerometer and orthogonal encoding of $m_{sup} = 2$ contexts, where it was much more likely for a tighter cluster from one context to “overwhelm” a noisier cluster from another. This effect was reduced as more contexts were superimposed, resulting in increasing performance with orthogonal encoding for higher m_{sup}.

There was also a slight dependence of performance on the parity of m_{sup}, with odd values outperforming even values when using context encoding. By orthogonalizing different

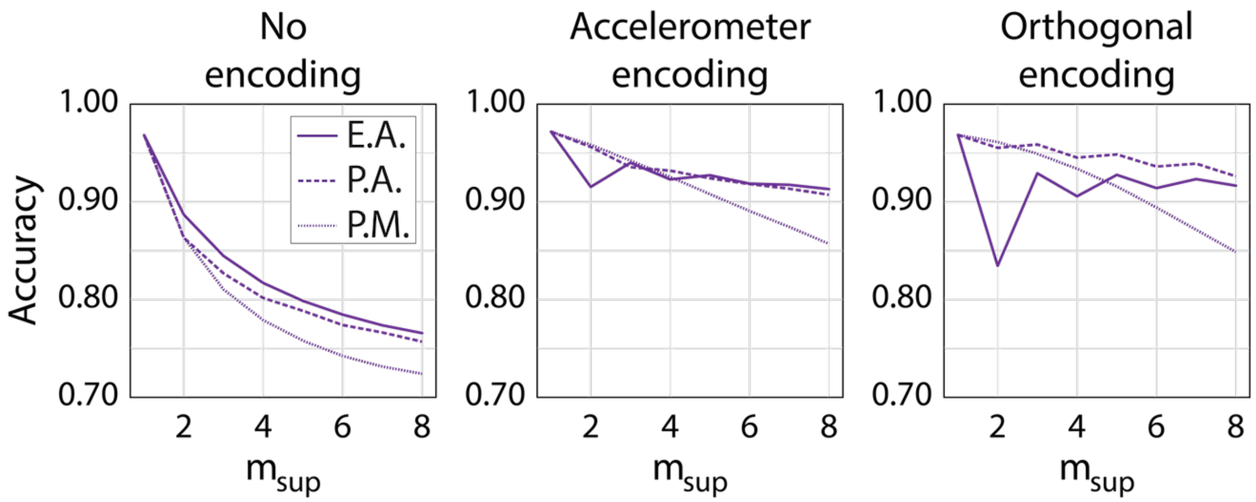


Figure 4.13: **Classification accuracy for increasing number of contexts in superposition.** E.A. = example accumulation; P.A. = prototype accumulation; P.M. = prototype merge.

contexts, the likelihood of ties in superposition was higher when summing an even number of contexts. Ties should be broken at random, but slight differences in spread caused tie-breaking to skew towards one context, reducing representation of others. Skewed tie-breaking is less likely to occur when summing an odd number of contexts.

Finally, prototype merge is shown to be a useful approximation of prototype accumulation only for small values of m_{sup}, similar to what we found with the FlexEMG in-sensor implementation [135]. Beyond $m_{sup} = 3$ or 4, the limited capacity of merging orthogonal HVs caused accuracy to degrade more.

Incremental HD classifier performance

For incremental learning performance evaluation, I first divided the gesture recognition dataset into 5 folds for leave-one-out cross-validation. I further split training and testing datasets into context-specific datasets. I generated a random ordering for the contexts and provided each context-specific training dataset to the incremental learning algorithm one at a time. After each step, the model was tested on all testing datasets corresponding to the seen contexts, and this was continued until all contexts were learned. This process was repeated for each subject individually, and at least 128 random context permutations were generated for each experiment.

I measured three metrics during the incremental training process, the first being average classification accuracy. I computed context-specific classification accuracy $a_{i,j}$ corresponding to the jth seen context after i training steps. The average classification accuracy was then computed as $A_i = \frac{1}{i}\sum_{j=1}^{i} a_{i,j}$.

While accuracy reflected the overall performance, we can better analyze model deficiencies by computing forgetting and intransigence [227]. Forgetting quantifies how much the incremental learning algorithm loses what it has learned in the past due to overwriting or interfering with relevant parameters. I calculated forgetting for the jth seen context after the ith step as $f_{i,j} = \max_{l\in\{1,2,...,i-1\}} a_{l,j} - a_{i,j}$. The first term was the best classification accuracy ever achieved by the model, and the second was the accuracy with the updated model. More positive values of $f_{i,j}$ were worse and indicated more forgetting, while negative values indicated that the model had improved for that context after incrementally training with other contexts. Average forgetting for each step was then computed as $F_i = \frac{1}{i-1}\sum_{j=1}^{i-1} f_{i,j}$.

Intransigence, on the other hand, measures the inability of the model to incrementally learn a new context, as compared to when the model was batch trained. This was computed as $I_i = a_i^* - a_{i,i}$, where a_i^* was the batch trained model accuracy using the same seen data. Like forgetting, poorer intransigence was represented by greater positive values. For the incremental HD classifier, I measured intransigence against a batch-trained, fully superimposed ($m_{sep} = 0$) HD model with example accumulation.

Figure 4.14 shows the performance metrics measured after each incremental step for the various configurations of the HD classifier. In each 3×3 grid of Figs. 4.14a-c, each configuration is represented by a single plot with different colored lines for varying m_{sep}. Metrics were averaged over all subjects, all possible context permutations, and all possible groupings of superimposed and separate contexts. Each step is shown as incrementally learning an entire new limb position context, although the training process was able to incorporate new context data one gesture at a time.

Average classification accuracy (Figure 4.14a) degraded as more contexts were stored in superposition. This is expected from the superposition analysis, with the curves representing $m_{sep} = 0$ in Figure 4.14a being identical to those in Figure 4.13. Higher m_{sep} budgets traded off memory footprint for better performance. However, storing superimposed prototypes together with separate prototypes also introduced some negative effects, as shown in the top row of Figure 4.14d. Here, I plot the metrics achieved by each configuration after completing all 8 incremental steps, demonstrating that there was at first a slight accuracy degradation when increasing m_{sep} with accelerometer encoding, contrary to intuition. This was due to certain pairs of contexts producing more similar accelerometer encoded context vectors (e.g., positions 0 and 4). With smaller, non-zero m_{sep}, there was an increased chance that one of the pair was stored separately while the other was superimposed with a large number of more distant contexts. Examples from the superimposed context could be most similar to the separate paired context prototypes, causing misclassification.

The causes of accuracy degradation over incremental training steps can be seen in the average amount of forgetting, plotted in Figure 4.14b. Compared to $m_{sep} = 0$, small non-zero values for m_{sep} caused worse forgetting since superimposed contexts were randomly selected. In these cases, there was a greater likelihood that prototypes for a context were at one point stored separately, achieving maximum within-context accuracy, and then subsequently superimposed, causing some accuracy degradation. Along with the paired context effect explained above, this caused a peak in average forgetting after all incremental steps to fall near $m_{sep} = 2$ for no encoding and accelerometer encoding, as shown in the middle row of Figure 4.14d. As I increased m_{sep} to allow for separate prototypes for all contexts, the forgetting dropped to near 0. For orthogonal encoding, there was no peak, but rather a dependence on the parity of m_{sep} when using example accumulation.

Finally, plotting intransigence shows that incrementally trained HD classifiers were for the most part more capable of incorporating new context information than the batch trained model due to their ability to store new prototypes separately (Figure 4.14c). This effect was much more pronounced when not using any context encoding. When constrained to superimpose new contexts, models exhibited higher intransigence with the number of incremental training steps. However, using prototype accumulation with orthogonal encoding resulted in negative intransigence even for $m_{sep} = 0$, suggesting that bipolarization in two stages should also be leveraged during batch training for improved performance.

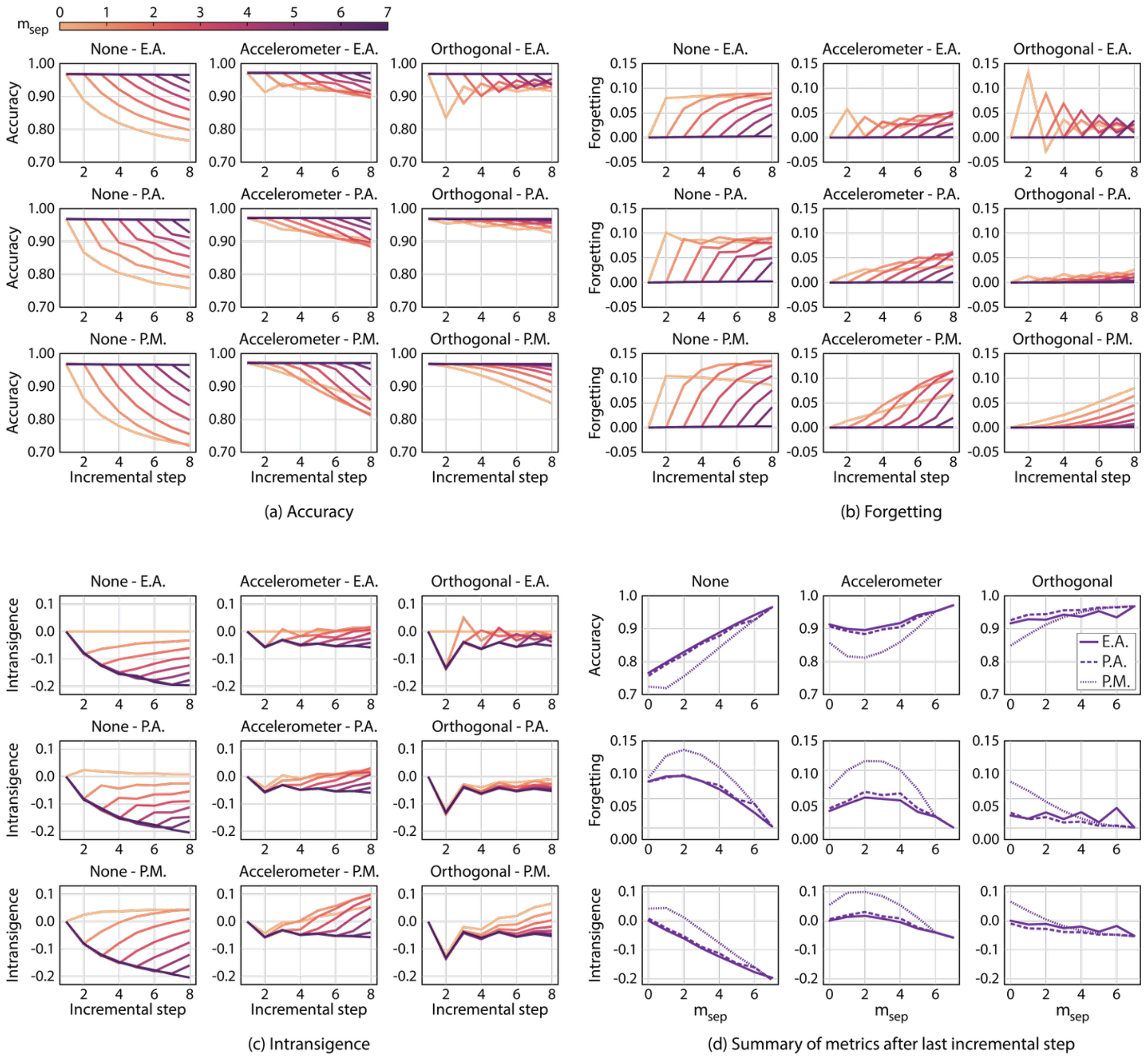


Figure 4.14: **Incremental learning performance using HD computing.** Measured performance metrics include **a)** accuracy, **b)** forgetting, and **c)** intransigence with respect to a batch trained HD classifier with $m_{sep} = 0$. Within the 3×3 grids, each row of plots represents a different incremental superposition configuration (E.A. = example accumulation; P.A. = prototype accumulation; P.M. = prototype merge) and each column represents a different context encoding method. Colors represent the value of m_{sep}. **d)** Summary of results after the last incremental step with varying m_{sep}. Each row represents a different metric, each column represents a different context encoding method, and line styles represent the incremental superposition configuration.

Incremental SVM and LDA performance

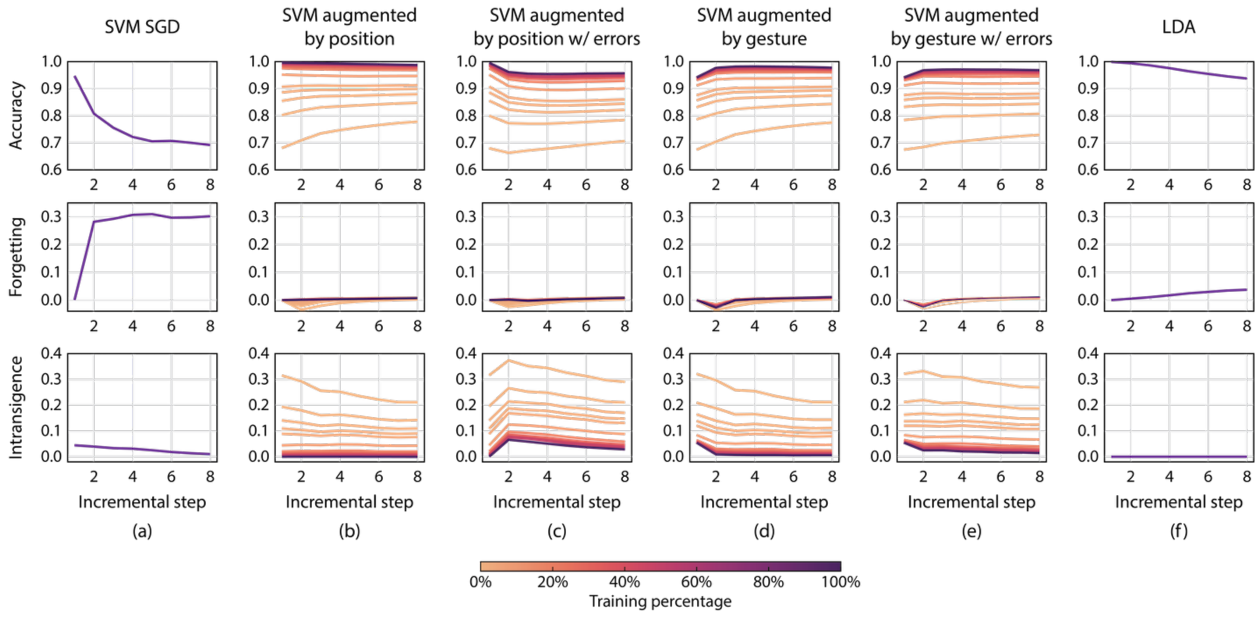


Figure 4.15: **Performance comparison with SVM and LDA classifiers.** Each column plots the accuracy, forgetting, and intransigence metrics for **a)** SVM updated through stochastic gradient descent (SVM SGD), **b)** SVM with data augmentation using entire context-specific datasets, **c)** SVM with data augmentation using misclassified examples from entire context-specific datasets, **d)** SVM with data augmentation using gesture-by-gesture datasets, **e)** SVM with data augmentation using misclassified examples from gesture-by-gesture datasets, and **f)** incrementally trained LDA. For the SVM with augmented datasets, different training percentages were used and plotted in different colors.

Figure 4.15 shows the performance of SVM and LDA in our incremental learning experiments. The SVM trained using stochastic gradient descent (SVM SGD, Figure 4.15a) experienced large amounts of accuracy degradation, with a classification accuracy of under 70% after all incremental training steps. This was almost entirely due to forgetting, a well known issue with gradient descent-trained classifiers.

Methods of dataset augmentation for incrementally update an SVM classifier (Figure 4.15b-e) produced results with similar dependencies on the percentage of training data used, with almost all error due to intransigence. Incrementally augmenting the dataset for each context (Figure 4.15b) in one step produced more consistent accuracy early on than incrementally augmenting gesture-by-gesture (Figure 4.15d). However, the performance difference was marginal after all steps. Augmenting the dataset with only misclassified examples (Figure 4.15c, e) made a larger impact when incrementally training on entire contexts at a time than gesture-by-gesture.

Finally, the performance of the LDA classifier (Figure 4.15f) showed that with accurate mean and covariance updates, classification accuracy fell gracefully due to forgetting.

Comparison of memory footprint

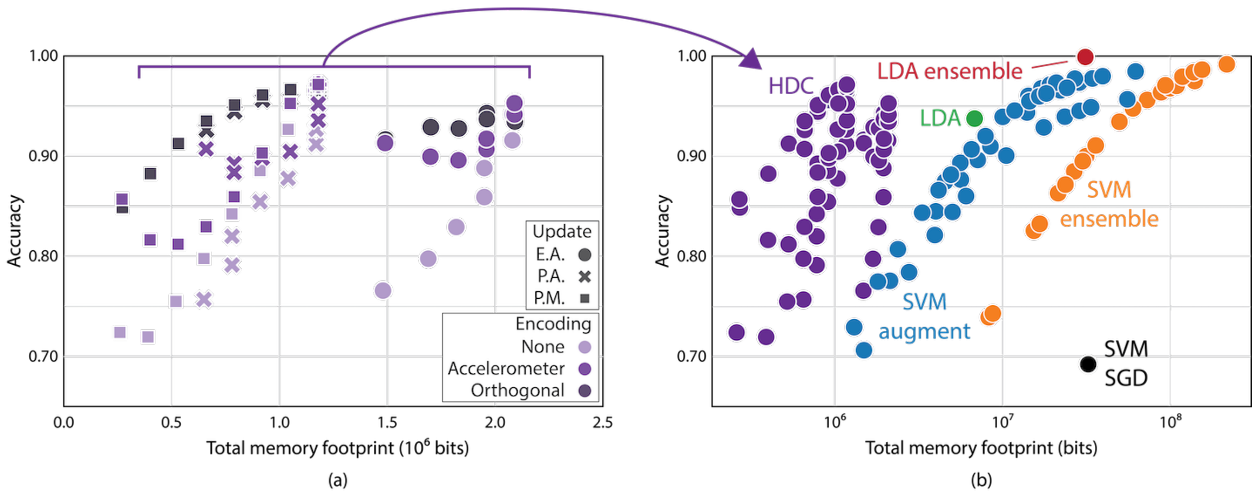


Figure 4.16: **Memory footprint and accuracy of incremental HD classifier configurations.** (a) Differentiation between different HD classifier configurations. Total memory includes both parameter storage and scratch memory for training. Shapes represent the different incremental superposition configurations (E.A. = example accumulation; P.A. = prototype accumulation; P.M. = prototype merge), and colors represent the different encoding methods. (b) Comparison of HD configurations with traditional models. HDC = incremental HD; LDA ensemble = ensemble of context-specific LDA; LDA = incrementally updated LDA; SVM ensemble = ensemble of context-specific SVM; SVM augment = incrementally retrained SVM using dataset augmentation; SVM SGD = SVM trained with stochastic gradient descent.

Figure 4.16a shows memory footprint vs. classification accuracy for incremental HD classifier configurations after all incremental steps. As expected, superposition using example accumulation required the largest memory footprint to store higher-precision accumulations. Each configuration employing example accumulation was bested by a smaller configuration using prototype accumulation or merge for equivalent or better accuracy. Between prototype accumulation and merge configurations, accuracy depended mostly on the context encoding method. The Pareto set consisted almost entirely of orthogonal encoding configurations, except for the smallest, single-prototype configuration using prototype merge and accelerometer encoding. At the other extreme, all of the configurations performed equally well when $m_{sep} = m$.

It is worth noting that while memory footprint comparison is a good baseline, it does not paint the full picture. For example, of the configurations sized approximately 600 kb, the most accurate used prototype merge with $m_{sep} = 3$ for a total of 4 prototypes per class. Slightly below this in accuracy was the model using prototype accumulation with $m_{sep} = 0$. Despite having the same memory footprint, inference for this configuration would be much quicker, requiring 1/4 the number of comparisons with a query. Since a comparison consists of $D = 10,000$ XORs followed by a popcount of D bits, this could be a substantial saving in inference computation without much sacrifice in terms of accuracy.

Finally, I compare the incremental HD classifier configurations with traditional machine learning models in Figure 4.16b. The best HD configurations were able to achieve similar accuracies while occupying over an order of magnitude smaller footprints.

The SVM trained using SGD was unable to produce acceptable results while also requiring a large memory footprint. Performance of SVMs incrementally trained through dataset augmentation was extremely dependent on the number of support vectors, which varied with training percentage. Even the smallest SVM models were larger than the best performing HD classifiers. Training an ensemble of context-specific SVMs improved the top-end classification accuracy but resulted in the largest of models we tested.

I was able to achieve the highest classification accuracy overall with an ensemble of context-specific LDA classifiers with a total memory footprint more than 26× the size of the most accurate HD model. Incrementally updating a single LDA classifier required only 3× more memory but had poorer classification performance. My comparison shows that incremental HD classifiers were considerably more memory efficient than traditional methods and made up a large portion of the Pareto set when considering both memory footprint and accuracy.

4.6 Discussion

In this chapter, I presented various configurations of incrementally trained HD classification models for hand gesture recognition in different limb positions. Part of the choice in configuration was whether to and how to encode accelerometer signals along with EMG signals to incorporate information about the limb position context.

I first described a sensor fusion-based method for context-aware classification of EMG signals for gesture recognition. By using accelerometer signals to bind context information to existing EMG parameters, we can achieve high classification accuracy in 8 limb positions with a model that does not increase memory requirement over a single limb position. I first evaluated the improvement in classification margin when using context-based orthogonalization for an HD classifier, and I highlighted its dependency on relationships between class prototypes and the number of contexts to be learned. I proposed two methods to generate context vectors for different limb positions using accelerometer signals, and both methods require minimal additions to a baseline HD computing gesture classifier, which can already be fully implemented on a wearable device [135].

I then offer a range of choices for incremental hypervector superposition methods, and I demonstrate how increasing the allowance for multiple class prototypes can improve classification accuracy. These choices can be made depending on the memory footprint allowed by a platform on which an in-sensor implementation is to be embedded. Incremental training experiments show that HD classifiers can be efficiently updated in multiple limb position contexts while maintaining competitive classification accuracy as compared with traditional machine learning models that require considerably larger memories.

My results point towards a way to improve reliability of gesture recognition for wearables in everyday situations, and the presented method can potentially be applied to other multi-context biosignal classification applications [228–230].

Chapter 5

Conclusions and Future Work

In this thesis, I presented two systems with which we demonstrated different forms of robustness to interference and confounding factors during deployment of a smart algorithm.

In Chapter 2, I first reviewed strategies for dealing with self-interference due to stimulation artifact in systems requiring simultaneous neural recording and stimulation. I then presented WAND, a device for closed-loop neuromodulation in which we co-designed the front-end neural recording circuitry with the back-end digital artifact cancellation algorithms to effectively remove stimulation artifacts from recorded neural signals. This enabled true closed-loop control, in which neural biomarkers could be accurately extracted in real time for analysis or classification *while* stimulation was being delivered on neighboring electrodes.

In Chapter 3, I then presented the FlexEMG system, which builds off of the hardware of WAND for the application of hand gesture recognition using electromyography (EMG). When integrating the device with a custom, screen-printed flexible electrode array and implementing a hyperdimensional computing classification algorithm in the on-board FPGA, we demonstrated real-time gesture recognition using a classification model that supported in-sensor training and updates when new situational contexts had to be learned.

Chapter 4 then expanded on the hyperdimensional computing model more, covering various ways that it could be improved to generalize across a larger number of situational contexts. In the case of performing gesture recognition in multiple limb positions, I showed how limb position information derived from on-device accelerometers could be efficiently encoded in the classifier to improve accuracy across all positions without significantly increasing the number of model parameters to be stored. I then concluded by offering a range of configurations for the hyperdimensional computing model in which we could *incrementally* train the classifier over time to learn the limb positions one-by-one. For these configurations, I discussed the trade-offs between classification accuracy, how previously learned training examples should be stored and represented for future use, and what operations should be used for updating the stored parameters.

While we have demonstrated two realistic use cases of smart wearables and implantables, there still remain a wide range of challenges to better performance of these devices. I will conclude this thesis with a brief discussion of prospective work for the future of these projects.

5.1 Enabling larger-scale closed-loop neuromodulation research

One of the goals for open-sourcing the WAND architecture was for it to function as a general-purpose research device, requiring only minor modifications to be re-optimized for new applications. Algorithmic development on the FPGA and microcontroller allows for extracting other neural biomarkers such as band powers or line-lengths used to detect seizure onset in epileptic patients [23], and new closed-loop classification and control algorithms may be conceived for integrating neural activity from more recording channels. Future work can also incorporate other features of WAND, such as the inertial sensor and multi-site stimulation, as well as more efficient algorithm design utilizing custom hardware in the FPGA.

For stimulation artifact cancellation, further research in electrode configuration and improved back-end artifact algorithms may allow recovery of the full underlying neural signal with no lost samples, a limitation WAND still faces. The closed-loop experiment demonstrated in Chapter 2 relied only on a single channel for recording and a single channel for stimulation, and it is easy to imagine a scenario where far greater channel counts are required in both the afferent and efferent directions. In these cases, more complex models of interactions between multiple stimulation and recording sites will be necessary to improve efficacy and efficiency of artifact cancellation algorithms with low implementation costs.

5.2 Improved hybrid flexible electronic devices

On the hardware side for the FlexEMG system, there is still much room for improvement in creating a more well-integrated, comfortable, and easy-to-use wearable. Beginning at the electrode interface, one of the biggest drawbacks of the current FlexEMG setup is the use of electrode gel or paste in applying the EMG electrodes to the forearm. While gel at first improves the impedance of the interface and reduces the amount of electrode lifting and motion artifact due to flexion in the array, it is tedious to apply and would be especially burdensome for everyday use. Furthermore, it does not last very long, and can eventually hurt signal quality [231]. Instead, EMG and other biosignals may be recorded using dry electrodes on the skin, simplifying the donning process considerably [1, 3, 4]. Dry electrodes have higher impedances and are more susceptible to noise, so novel electrode fabrication techniques will be required to enable creation of large scale electrode arrays, such as the one we use for forearm EMG, with geometries and surface characteristics conducive to good recording quality. Improvements in the flexible substrate will also be necessary in order to increase elasticity and enable better compression. Currently, the electrode array is held in place with tape and slight adhesion due to the electrode gel; dry electrodes will require greater compression to ensure contact and minimize the electrode impedance.

Additionally, there is much to be learned from actual in-sensor implementation (either through development on the on-board FPGA or design of new, custom application-specific integrated circuits) of the improved sensor-fusion methodology and incremental learning configurations for HD computing presented in Chapter 4. As demonstrated in Chapter 3, there was a considerable improvement in classification accuracy when subjects were given real-time visual feedback of the predicted label, enabling co-adaptation by the user and the classification model. This type of co-adaptation, in which the user can change how they perform the gestures to better match what the model expects, would likely enable better performance with fewer incremental updates to further reduce memory footprint of the model. In-sensor implementation would also enable better assessments of the hardware implementation costs for the different configurations, as well as measurements of power dissipation, which is crucial to keep at a minimum for wireless, battery-powered operation. Ultimately, co-integration of the recording front-ends and classification back-end on the same chip would be ideal, minimizing power and area and reducing component count for future placement directly on the flexible substrate alongside the electrodes.

5.3 Toward continuous, unsupervised adaptation

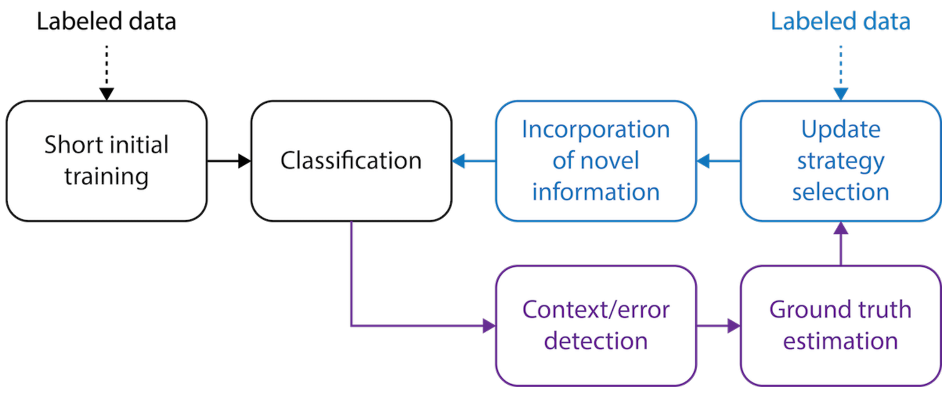


Figure 5.1: **Framework for continuous, unsupervised adaptation.** A short initial training session using labeled data is used to build the baseline classification model (black, left). Incorporation of new information for incremental learning and model updates can performed periodically using additional labeled data (right, blue). Unsupervised adaptation implemented in the form of a loop, depending on the classification model's predictions to determine when updating is necessary, estimate labels for new data, and then following the same update process (right, purple).

With respect to the gesture recognition algorithm itself, we have presented incremental learning and model update methods that require supervision—we assume that ground-truth labels can be provided for new training data. While this enables a user to "teach" the classification algorithm new instances of the different classes, a more user-friendly approach would be for the model to continuously learn and adapt in an *unsupervised* manner. This could potentially involve an additional part of the model that can detect changes in context or errors in the classification output, followed by some way of estimating the labels for new data to enable updates of model parameter. Figure 5.1 shows a high-level framework for such a method of adaptation. Following initial supervised training of the model, it can be updated either in a supervised way with new labeled data (which we have demonstrated in this thesis), or in an unsupervised way based on the predicted labels.

In this way, a gesture recognition device would no longer need supervision from the user or physician, but rather it could autonomously go about maintaining or improving its performance capabilities. The biggest challenges in enabling this unsupervised adaptation loop are the context change or error detection block and ground truth estimation block, which should work in conjunction with one another. The goal for the first block is to identify situations in which updates are needed, e.g., when new situational contexts that introduce accuracy degradation are encountered. For our example with different limb positions, this may be the detection of a unlearned limb position based on novelty detection in the accelerometer signals. Error detection may also be based directly on the stream of predicted labels as output by the model. Several works have demonstrated unsupervised adaptation strategies based on the confidence or entropy of the classifier outputs [207, 232], but these methods have not been able to match the performance of using supervised model updates. A primary reason for this is ground truth estimation. Updates are required when errors are detected, but because of the errors, the algorithm has no access to correctly labeled examples for recalculation of parameters. This suggests that a semi-supervised approach may be the most suitable, in which the user provides labels for a small amount of data, and the algorithm fills in the gaps for what remains.

Finally, a potential research avenue to explore which unifies all of the concepts in this thesis could be development of closed-loop algorithms leveraging HD computing to run on bi-directional neural interfaces. Neuromodulation applications such as seizure detection and treatment face a similar set of problems as gesture recognition, in which the target biomarkers used for detection of seizure events can change over time [233]. Hyperdimensional computing has been previously applied to the seizure detection task [234], and such classifiers can also leverage the incremental training abilities demonstrated in this work. It is easy to imagine the next generation of implantable neural interfaces for epilepsy treatment to be robust to both stimulation artifacts and the evolution of neural biomarkers over time.

Bibliography

[1] Ryan Kaveh et al. “Wireless User-Generic Ear EEG”. In: *IEEE Trans. Biomed. Circuits Syst.* 14.4 (Aug. 2020), pp. 727–737. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2020.3001265. arXiv: 2003.00657. URL: https://ieeexplore.ieee.org/document/9115876/.

[2] Zoe LaRock, Dane Finley, and Erum Ahmed. *The Digital Health Ecosystem.* Tech. rep. Insider Intelligence, 2020.

[3] Marco E. Benalcazar et al. “Hand gesture recognition using machine learning and the Myo armband”. In: *2017 25th Eur. Signal Process. Conf.* Vol. 2017-Janua. IEEE, Aug. 2017, pp. 1040–1044. ISBN: 978-0-9928626-7-1. DOI: 10.23919/EUSIPCO.2017.8081366. URL: http://ieeexplore.ieee.org/document/8081366/.

[4] Edward F. Melcer et al. “CTRL-Labs”. In: *Ext. Abstr. 2018 CHI Conf. Hum. Factors Comput. Syst.* Vol. 2018-April. New York, NY, USA: ACM, Apr. 2018, pp. 1–4. ISBN: 9781450356213. DOI: 10.1145/3170427.3186520. URL: https://dl.acm.org/doi/10.1145/3170427.3186520.

[5] Yasser Khan. “Wearable Medical Sensors Enabled by Printed Bioelectronics and Biophotonics”. PhD thesis. EECS Department, University of California, Berkeley, Dec. 2020. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2020/EECS-2020-189.html.

[6] *NeuroPace RNS System User Manual.* 2015.

[7] Michelle Paff et al. “Update on Current Technologies for Deep Brain Stimulation in Parkinson’s Disease”. In: *J. Mov. Disord.* 13.3 (Sept. 2020), pp. 185–198. ISSN: 2005-940X. DOI: 10.14802/jmd.20052. URL: http://e-jmd.org/journal/view.php?doi=10.14802/jmd.20052.

[8] Lewis Wallace. *AssistiveTouch lets users control Apple Watch by clenching a fist.* 2021. (Visited on 12/07/2021).

[9] Facebook. *Inside Facebook Reality Labs: Wrist-based interaction for the next computing platform.* 2021. (Visited on 12/07/2021).

[10] *NeuroPace RNS System — For Providers.* 2021. URL: https://neuropace.com/providers/ (visited on 05/20/2021).

[11] Scott Stanslaski et al. "Design and Validation of a Fully Implantable, Chronic, Closed-Loop Neuromodulation Device With Concurrent Sensing and Stimulation". In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 20.4 (July 2012), pp. 410–421. ISSN: 1534-4320. DOI: `10.1109/TNSRE.2012.2183617`. URL: `http://ieeexplore.ieee.org/document/6135801/`.

[12] Alim Louis Benabid et al. "Deep brain stimulation of the subthalamic nucleus for the treatment of Parkinson's disease". In: *Lancet Neurol.* 8.1 (Jan. 2009), pp. 67–81. ISSN: 14744422. DOI: `10.1016/S1474-4422(08)70291-6`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1474442208702916`.

[13] Jeff M. Bronstein et al. "Deep Brain Stimulation for Parkinson Disease". In: *Arch. Neurol.* 68.2 (Feb. 2011). ISSN: 0003-9942. DOI: `10.1001/archneurol.2010.260`. URL: `http://archneur.jamanetwork.com/article.aspx?doi=10.1001/archneurol.2010.260`.

[14] Cheng-Hsiang Cheng et al. "A fully integrated closed-loop neuromodulation SoC with wireless power and bi-directional data telemetry for real-time human epileptic seizure control". In: *2017 Symp. VLSI Circuits*. IEEE, June 2017, pp. C44–C45. ISBN: 978-4-86348-614-0. DOI: `10.23919/VLSIC.2017.8008541`. URL: `http://ieeexplore.ieee.org/document/8008541/`.

[15] Hossein Kassiri et al. "All-wireless 64-channel 0.013mm 2 /ch closed-loop neurostimulator with rail-to-rail DC offset removal". In: *2017 IEEE Int. Solid-State Circuits Conf.* Vol. 60. IEEE, Feb. 2017, pp. 452–453. ISBN: 978-1-5090-3758-2. DOI: `10.1109/ISSCC.2017.7870456`. URL: `http://ieeexplore.ieee.org/document/7870456/`.

[16] Martha J. Morrell. "Responsive cortical stimulation for the treatment of medically intractable partial epilepsy". In: *Neurology* 77.13 (Sept. 2011), pp. 1295–1304. ISSN: 0028-3878. DOI: `10.1212/WNL.0b013e3182302056`. URL: `http://www.neurology.org/cgi/doi/10.1212/WNL.0b013e3182302056`.

[17] *Deep Brain Stimulation Systems - Activa Platform.* 2021. URL: `https://www.medtronic.com/us-en/healthcare-professionals/products/neurological/deep-brain-stimulation-systems/activa-platform.html`.

[18] Eric R Kandel et al. *Principles of Neural Science.* 5th Edit. McGraw-Hill Companies, 2013. ISBN: 978-0-07-139011-8. URL: `https://neurology.mhmedical.com/book.aspx?bookid=1049`.

[19] Milin Zhang et al. "Electronic neural interfaces". In: *Nat. Electron.* 3.4 (Apr. 2020), pp. 191–200. ISSN: 2520-1131. DOI: `10.1038/s41928-020-0390-3`. URL: `https://doi.org/10.1038/s41928-020-0390-3%20http://www.nature.com/articles/s41928-020-0390-3`.

[20] Keith Mathieson et al. "Photovoltaic retinal prosthesis with high pixel density". In: *Nat. Photonics* 6.6 (June 2012), pp. 391–397. ISSN: 1749-4885. DOI: `10.1038/nphoton.2012.104`. URL: `http://www.nature.com/articles/nphoton.2012.104`.

[21] Marcus Yip et al. “A Fully-Implantable Cochlear Implant SoC With Piezoelectric Middle-Ear Sensor and Arbitrary Waveform Neural Stimulation”. In: *IEEE J. Solid-State Circuits* 50.1 (Jan. 2015), pp. 214–229. ISSN: 0018-9200. DOI: 10.1109/JSSC.2014.2355822. URL: http://ieeexplore.ieee.org/document/6910323/.

[22] Todd M. Herrington, Jennifer J. Cheng, and Emad N. Eskandar. “Mechanisms of deep brain stimulation”. In: *J. Neurophysiol.* 115.1 (Jan. 2016), pp. 19–38. ISSN: 0022-3077. DOI: 10.1152/jn.00281.2015. URL: https://www.physiology.org/doi/10.1152/jn.00281.2015.

[23] Felice T Sun, Martha J Morrell, and Robert E Wharen. “Responsive cortical stimulation for the treatment of epilepsy”. In: *Neurotherapeutics* 5.1 (Jan. 2008), pp. 68–74. ISSN: 1933-7213. DOI: 10.1016/j.nurt.2007.10.069. URL: http://ovidsp.ovid.com/ovidweb.cgi?T=JS%7B%5C&%7DPAGE=reference%7B%5C&%7DD=emed8%7B%5C&%7DNEWS=N%7B%5C&%7DAN=2007620738%7B%5C%%7D5Cnhttp://www.sciencedirect.com/science/article/pii/S1933721307002577%20http://link.springer.com/10.1016/j.nurt.2007.10.069.

[24] Ben H. Bonham and Leonid M. Litvak. “Current focusing and steering: Modeling, physiology, and psychophysics”. In: *Hear. Res.* 242.1-2 (Aug. 2008), pp. 141–153. ISSN: 03785955. DOI: 10.1016/j.heares.2008.03.006. URL: https://linkinghub.elsevier.com/retrieve/pii/S0378595508000610.

[25] Sergey D. Stavisky et al. “A high performing brain–machine interface driven by low-frequency local field potentials alone and together with spikes”. In: *J. Neural Eng.* 12.3 (June 2015), p. 036009. ISSN: 1741-2560. DOI: 10.1088/1741-2560/12/3/036009. URL: https://iopscience.iop.org/article/10.1088/1741-2560/12/3/036009.

[26] Xiaoxuan Jia and Adam Kohn. “Gamma Rhythms in the Brain”. In: *PLoS Biol.* 9.4 (Apr. 2011), e1001045. ISSN: 1545-7885. DOI: 10.1371/journal.pbio.1001045. URL: https://dx.plos.org/10.1371/journal.pbio.1001045.

[27] Keyoumars Ashkan et al. “Insights into the mechanisms of deep brain stimulation”. In: *Nat. Rev. Neurol.* 13.9 (Sept. 2017), pp. 548–554. ISSN: 1759-4758. DOI: 10.1038/nrneurol.2017.105. URL: http://www.nature.com/articles/nrneurol.2017.105.

[28] Michael Eisenstein. “Electrotherapy: Shock value”. In: *Nature* 538.7626 (Oct. 2016), S10–S12. ISSN: 0028-0836. DOI: 10.1038/538S10a. URL: http://www.nature.com/articles/538S10a.

[29] Boris Rosin et al. “Closed-Loop Deep Brain Stimulation Is Superior in Ameliorating Parkinsonism”. In: *Neuron* 72.2 (Oct. 2011), pp. 370–384. ISSN: 08966273. DOI: 10.1016/j.neuron.2011.08.023. URL: https://linkinghub.elsevier.com/retrieve/pii/S0896627311007768.

[30] Nicole C. Swann et al. “Adaptive deep brain stimulation for Parkinson’s disease using motor cortex sensing”. In: *J. Neural Eng.* 15.4 (Aug. 2018), p. 046006. ISSN: 1741-2560. DOI: 10.1088/1741-2552/aabc9b. URL: https://iopscience.iop.org/article/10.1088/1741-2552/aabc9b.

[31] Daniel A. Wagenaar and Steve M. Potter. “Real-time multi-channel stimulus artifact suppression by local curve fitting”. In: *J. Neurosci. Methods* 120.2 (Oct. 2002), pp. 113–120. ISSN: 01650270. DOI: 10.1016/S0165-0270(02)00149-8. URL: https://linkinghub.elsevier.com/retrieve/pii/S0165027002001498.

[32] Andy Zhou, Benjamin C. Johnson, and Rikky Muller. “Toward true closed-loop neuromodulation: artifact-free recording during stimulation”. In: *Curr. Opin. Neurobiol.* 50 (June 2018), pp. 119–127. ISSN: 09594388. DOI: 10.1016/j.conb.2018.01.012. URL: https://linkinghub.elsevier.com/retrieve/pii/S095943881730243X.

[33] Daniel R. Merrill, Marom Bikson, and John G.R. Jefferys. “Electrical stimulation of excitable tissue: design of efficacious and safe protocols”. In: *J. Neurosci. Methods* 141.2 (Feb. 2005), pp. 171–198. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2004.10.020. URL: https://linkinghub.elsevier.com/retrieve/pii/S0165027004003826.

[34] Daniel A. Wagenaar, Jerome Pine, and Steve M. Potter. “Effective parameters for stimulation of dissociated cultures using multi-electrode arrays”. In: *J. Neurosci. Methods* 138.1-2 (Sept. 2004), pp. 27–37. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2004.03.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S0165027004001141.

[35] Chun Wang et al. “Characteristics of electrode impedance and stimulation efficacy of a chronic cortical implant using novel annulus electrodes in rat motor cortex”. In: *J. Neural Eng.* 10.4 (Aug. 2013), p. 046010. ISSN: 1741-2560. DOI: 10.1088/1741-2560/10/4/046010. URL: https://iopscience.iop.org/article/10.1088/1741-2560/10/4/046010.

[36] Edgar A. Brown et al. “Stimulus-Artifact Elimination in a Multi-Electrode System”. In: *IEEE Trans. Biomed. Circuits Syst.* 2.1 (Mar. 2008), pp. 10–21. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2008.918285. URL: http://ieeexplore.ieee.org/document/4464125/.

[37] R.A. Blum et al. “An Integrated System for Simultaneous, Multichannel Neuronal Stimulation and Recording”. In: *IEEE Trans. Circuits Syst. I Regul. Pap.* 54.12 (Dec. 2007), pp. 2608–2618. ISSN: 1549-8328. DOI: 10.1109/TCSI.2007.906071. URL: http://ieeexplore.ieee.org/document/4358608/.

[38] R.A. Blum et al. “Models of stimulation artifacts applied to integrated circuit design”. In: *26th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* Vol. 4. IEEE, 2004, pp. 4075–4078. ISBN: 0-7803-8439-3. DOI: 10.1109/IEMBS.2004.1404137. URL: http://ieeexplore.ieee.org/document/1404137/.

[39] Emilia Noorsal et al. "A Neural Stimulator Frontend With High-Voltage Compliance and Programmable Pulse Shape for Epiretinal Implants". In: *IEEE J. Solid-State Circuits* 47.1 (Jan. 2012), pp. 244–256. ISSN: 0018-9200. DOI: 10.1109/JSSC.2011.2164667. URL: http://ieeexplore.ieee.org/document/6025221/.

[40] Elliot Greenwald et al. "A CMOS neurostimulator with on-chip DAC calibration and charge balancing". In: *2013 IEEE Biomed. Circuits Syst. Conf.* IEEE, Oct. 2013, pp. 89–92. ISBN: 978-1-4799-1471-5. DOI: 10.1109/BioCAS.2013.6679646. URL: http://ieeexplore.ieee.org/document/6679646/.

[41] Ji-Jon Sit and Rahul Sarpeshkar. "A Low-Power Blocking-Capacitor-Free Charge-Balanced Electrode-Stimulator Chip With Less Than 6 nA DC Error for 1-mA Full-Scale Stimulation". In: *IEEE Trans. Biomed. Circuits Syst.* 1.3 (Sept. 2007), pp. 172–183. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2007.911631. URL: http://ieeexplore.ieee.org/document/4404854/.

[42] Benjamin C. Johnson et al. "An implantable 700μW 64-channel neuromodulation IC for simultaneous recording and stimulation with rapid artifact recovery". In: *2017 Symp. VLSI Circuits*. IEEE, June 2017, pp. C48–C49. ISBN: 978-4-86348-614-0. DOI: 10.23919/VLSIC.2017.8008543. URL: http://ieeexplore.ieee.org/document/8008543/.

[43] Eric Pepin et al. "A high-voltage compliant, electrode-invariant neural stimulator front-end in 65nm bulk-CMOS". In: *ESSCIRC Conf. 2016 42nd Eur. Solid-State Circuits Conf.* Vol. 2016-Octob. IEEE, Sept. 2016, pp. 229–232. ISBN: 978-1-5090-2972-3. DOI: 10.1109/ESSCIRC.2016.7598284. URL: http://ieeexplore.ieee.org/document/7598284/.

[44] Ruslana Shulyzki et al. "320-Channel Active Probe for High-Resolution Neuromonitoring and Responsive Neurostimulation". In: *IEEE Trans. Biomed. Circuits Syst.* 9.1 (Feb. 2015), pp. 34–49. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2014.2312552. URL: http://ieeexplore.ieee.org/document/6825915/.

[45] Andreas Bahmer, Otto Peter, and Uwe Baumann. "Recording and analysis of electrically evoked compound action potentials (ECAPs) with MED-EL cochlear implants and different artifact reduction strategies in Matlab". In: *J. Neurosci. Methods* 191.1 (Aug. 2010), pp. 66–74. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2010.06.008. URL: https://linkinghub.elsevier.com/retrieve/pii/S0165027010003110.

[46] Philip Chu et al. "Equalization for intracortical microstimulation artifact reduction". In: *2013 35th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* IEEE, July 2013, pp. 245–248. ISBN: 978-1-4577-0216-7. DOI: 10.1109/EMBC.2013.6609483. URL: http://ieeexplore.ieee.org/document/6609483/.

[47] Erik J. Peterson et al. "Stimulation artifact rejection in closed-loop, distributed neural interfaces". In: *ESSCIRC Conf. 2016 42nd Eur. Solid-State Circuits Conf.* Vol. 2016-Octob. IEEE, Sept. 2016, pp. 233–236. ISBN: 978-1-5090-2972-3. DOI: `10.1109/ESSCIRC.2016.7598285`. URL: `http://ieeexplore.ieee.org/document/7598285/`.

[48] John D. Rolston, Robert E. Gross, and Steve M. Potter. "A low-cost multielectrode system for data acquisition enabling real-time closed-loop processing with rapid recovery from stimulation artifacts". In: *Front. Neuroeng.* 2.JUL (2009). ISSN: 16626443. DOI: `10.3389/neuro.16.012.2009`. URL: `http://journal.frontiersin.org/article/10.3389/neuro.16.012.2009/abstract`.

[49] W. A. Smith et al. "A scalable, highly-multiplexed delta-encoded digital feedback ECoG recording amplifier with common and differential-mode artifact suppression". In: *2017 Symp. VLSI Circuits.* IEEE, June 2017, pp. C172–C173. ISBN: 978-4-86348-614-0. DOI: `10.23919/VLSIC.2017.8008470`. URL: `http://ieeexplore.ieee.org/document/8008470/`.

[50] Stanislav Culaclii et al. "A hybrid hardware and software approach for cancelling stimulus artifacts during same-electrode neural stimulation and recording". In: *2016 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* Vol. 2016-Octob. IEEE, Aug. 2016, pp. 6190–6193. ISBN: 978-1-4577-0220-4. DOI: `10.1109/EMBC.2016.7592142`. URL: `http://ieeexplore.ieee.org/document/7592142/`.

[51] Adam E. Mendrela et al. "A Bidirectional Neural Interface Circuit With Active Stimulation Artifact Cancellation and Cross-Channel Common-Mode Noise Suppression". In: *IEEE J. Solid-State Circuits* 51.4 (Apr. 2016), pp. 955–965. ISSN: 0018-9200. DOI: `10.1109/JSSC.2015.2506651`. URL: `http://ieeexplore.ieee.org/document/7370773/`.

[52] Sudip Nag et al. "Sensing of Stimulus Artifact Suppressed Signals From Electrode Interfaces". In: *IEEE Sens. J.* 15.7 (July 2015), pp. 3734–3742. ISSN: 1530-437X. DOI: `10.1109/JSEN.2015.2399248`. URL: `http://ieeexplore.ieee.org/document/7029606/`.

[53] Paweł Hottowy et al. "Properties and application of a multichannel integrated circuit for low-artifact, patterned electrical stimulation of neural tissue". In: *J. Neural Eng.* 9.6 (Dec. 2012), p. 066005. ISSN: 1741-2560. DOI: `10.1088/1741-2560/9/6/066005`. URL: `https://iopscience.iop.org/article/10.1088/1741-2560/9/6/066005`.

[54] Subramaniam Venkatraman et al. "A System for Neural Recording and Closed-Loop Intracortical Microstimulation in Awake Rodents". In: *IEEE Trans. Biomed. Eng.* 56.1 (Jan. 2009), pp. 15–22. ISSN: 0018-9294. DOI: `10.1109/TBME.2008.2005944`. URL: `http://ieeexplore.ieee.org/document/4633669/`.

[55] G.A. DeMichele and P.R. Troyk. “Stimulus-resistant neural recording amplifier”. In: *Proc. 25th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. (IEEE Cat. No.03CH37439).* Vol. 4. IEEE, 2003, pp. 3329–3332. ISBN: 0-7803-7789-3. DOI: 10.1109/IEMBS.2003.1280857. URL: http://ieeexplore.ieee.org/document/1280857/.

[56] Yasuhiko Jimbo et al. “A system for MEA-based multisite stimulation”. In: *IEEE Trans. Biomed. Eng.* 50.2 (Feb. 2003), pp. 241–248. ISSN: 0018-9294. DOI: 10.1109/TBME.2002.805470. URL: http://ieeexplore.ieee.org/document/1185148/.

[57] F. Heer et al. “Single-chip microelectronic system to interface with living cells”. In: *Biosens. Bioelectron.* 22.11 (May 2007), pp. 2546–2553. ISSN: 09565663. DOI: 10.1016/j.bios.2006.10.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S0956566306004891.

[58] Vijay Viswam et al. “2048 action potential recording channels with 2.4 μVrms noise and stimulation artifact suppression”. In: *2016 IEEE Biomed. Circuits Syst. Conf.* IEEE, Oct. 2016, pp. 136–139. ISBN: 978-1-5090-2959-4. DOI: 10.1109/BioCAS.2016.7833750. URL: http://ieeexplore.ieee.org/document/7833750/.

[59] Thomas Wichmann and Annaelle Devergnas. “A novel device to suppress electrical stimulus artifacts in electrophysiological experiments”. In: *J. Neurosci. Methods* 201.1 (Sept. 2011), pp. 1–8. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2011.06.026. URL: https://linkinghub.elsevier.com/retrieve/pii/S016502701100361X.

[60] Cornelia Hartmann et al. “Closed-Loop Control of Myoelectric Prostheses With Electrotactile Feedback: Influence of Stimulation Artifact and Blanking”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 23.5 (Sept. 2015), pp. 807–816. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2014.2357175. URL: http://ieeexplore.ieee.org/document/6898020/.

[61] Erwin B. Montgomery, John T. Gale, and He Huang. “Methods for isolating extracellular action potentials and removing stimulus artifacts from microelectrode recordings of neurons requiring minimal operator intervention”. In: *J. Neurosci. Methods* 144.1 (May 2005), pp. 107–125. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2004.10.017. URL: https://linkinghub.elsevier.com/retrieve/pii/S0165027004003838.

[62] Leon F. Heffer and James B. Fallon. “A novel stimulus artifact removal technique for high-rate electrical stimulation”. In: *J. Neurosci. Methods* 170.2 (May 2008), pp. 277–284. ISSN: 01650270. DOI: 10.1016/j.jneumeth.2008.01.023. URL: https://linkinghub.elsevier.com/retrieve/pii/S0165027008000794.

[63] Ulrich Hoffmann et al. “Detection and removal of stimulation artifacts in electroencephalogram recordings”. In: *2011 Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* IEEE, Aug. 2011, pp. 7159–7162. ISBN: 978-1-4577-1589-1. DOI: 10.1109/IEMBS.2011.6091809. URL: http://ieeexplore.ieee.org/document/6091809/.

[64] Colin Waddell et al. "Deep Brain Stimulation Artifact Removal Through Under-Sampling and Cubic-Spline Interpolation". In: *2009 2nd Int. Congr. Image Signal Process.* IEEE, Oct. 2009, pp. 1–5. ISBN: 978-1-4244-4129-7. DOI: `10.1109/CISP.2009.5301199`. URL: `http://ieeexplore.ieee.org/document/5301199/`.

[65] Kanokwan Limnuson et al. "Real-Time Stimulus Artifact Rejection Via Template Subtraction". In: *IEEE Trans. Biomed. Circuits Syst.* 8.3 (June 2014), pp. 391–400. ISSN: 1932-4545. DOI: `10.1109/TBCAS.2013.2274574`. URL: `http://ieeexplore.ieee.org/document/6605583/`.

[66] Yaara Erez et al. "Generalized framework for stimulus artifact removal". In: *J. Neurosci. Methods* 191.1 (Aug. 2010), pp. 45–59. ISSN: 01650270. DOI: `10.1016/j.jneumeth.2010.06.005`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0165027010003079`.

[67] Takao Hashimoto, Christopher M. Elder, and Jerrold L. Vitek. "A template subtraction method for stimulus artifact removal in high-frequency deep brain stimulation". In: *J. Neurosci. Methods* 113.2 (Jan. 2002), pp. 181–186. ISSN: 01650270. DOI: `10.1016/S0165-0270(01)00491-5`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0165027001004915`.

[68] Xing Qian et al. "A Method for Removal of Deep Brain Stimulation Artifact From Local Field Potentials". In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 25.12 (Dec. 2017), pp. 2217–2226. ISSN: 1534-4320. DOI: `10.1109/TNSRE.2016.2613412`. URL: `https://ieeexplore.ieee.org/document/7575734/`.

[69] Limin Sun and Hermann Hinrichs. "Moving average template subtraction to remove stimulation artefacts in EEGs and LFPs recorded during deep brain stimulation". In: *J. Neurosci. Methods* 266 (June 2016), pp. 126–136. ISSN: 01650270. DOI: `10.1016/j.jneumeth.2016.03.020`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0165027016300267`.

[70] Lena Trebaul et al. "Stimulation artifact correction method for estimation of early cortico-cortical evoked potentials". In: *J. Neurosci. Methods* 264 (May 2016), pp. 94–102. ISSN: 01650270. DOI: `10.1016/j.jneumeth.2016.03.002`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0165027016300085`.

[71] Ke Zeng et al. "An EEMD-ICA Approach to Enhancing Artifact Rejection for Noisy Multivariate Neural Data". In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 24.6 (June 2016), pp. 630–638. ISSN: 1534-4320. DOI: `10.1109/TNSRE.2015.2496334`. URL: `http://ieeexplore.ieee.org/document/7317785/`.

[72] Tarik Al-ani et al. "Automatic removal of high-amplitude stimulus artefact from neuronal signal recorded in the subthalamic nucleus". In: *J. Neurosci. Methods* 198.1 (May 2011), pp. 135–146. ISSN: 01650270. DOI: `10.1016/j.jneumeth.2011.03.022`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0165027011001701`.

[73] Yiliang Lu et al. “Using independent component analysis to remove artifacts in visual cortex responses elicited by electrical stimulation of the optic nerve”. In: *J. Neural Eng.* 9.2 (Apr. 2012), p. 026002. ISSN: 1741-2560. DOI: `10.1088/1741-2560/9/2/026002`. URL: `https://iopscience.iop.org/article/10.1088/1741-2560/9/2/026002`.

[74] Andy Zhou et al. “A wireless and artefact-free 128-channel neuromodulation device for closed-loop stimulation and recording in non-human primates”. In: *Nat. Biomed. Eng.* 3.1 (Jan. 2019), pp. 15–26. ISSN: 2157-846X. DOI: `10.1038/s41551-018-0323-x`. URL: `https://doi.org/10.1038/s41551-018-0323-x%20http://www.nature.com/articles/s41551-018-0323-x`.

[75] Rikky Muller et al. “A Minimally Invasive 64-Channel Wireless μECoG Implant”. In: *IEEE J. Solid-State Circuits* 50.1 (Jan. 2015), pp. 344–359. ISSN: 0018-9200. DOI: `10.1109/JSSC.2014.2364824`. URL: `http://ieeexplore.ieee.org/document/6964818/`.

[76] Reid R. Harrison et al. “A Low-Power Integrated Circuit for a Wireless 100-Electrode Neural Recording System”. In: *IEEE J. Solid-State Circuits* 42.1 (Jan. 2007), pp. 123–133. ISSN: 0018-9200. DOI: `10.1109/JSSC.2006.886567`. URL: `http://ieeexplore.ieee.org/document/4039585/`.

[77] Rikky Muller, Simone Gambini, and Jan M. Rabaey. “A 0.013 mm2, 5 µW, DC-Coupled Neural Signal Acquisition IC With 0.5 V Supply”. In: *IEEE J. Solid-State Circuits* 47.1 (Jan. 2012), pp. 232–243. ISSN: 0018-9200. DOI: `10.1109/JSSC.2011.2163552`. URL: `http://ieeexplore.ieee.org/document/6015500/`.

[78] Fan Zhang, Jeremy Holleman, and Brian P. Otis. “Design of Ultra-Low Power Biopotential Amplifiers for Biosignal Acquisition Applications”. In: *IEEE Trans. Biomed. Circuits Syst.* 6.4 (Aug. 2012), pp. 344–355. ISSN: 1932-4545. DOI: `10.1109/TBCAS.2011.2177089`. URL: `http://ieeexplore.ieee.org/document/6129415/`.

[79] Ben Johnson and Alyosha Molnar. “An Orthogonal Current-Reuse Amplifier for Multi-Channel Sensing”. In: *IEEE J. Solid-State Circuits* 48.6 (June 2013), pp. 1487–1496. ISSN: 0018-9200. DOI: `10.1109/JSSC.2013.2257478`. URL: `http://ieeexplore.ieee.org/document/6510003/`.

[80] R.R. Harrison and Cameron Charles. “A low-power low-noise cmos for amplifier neural recording applications”. In: *IEEE J. Solid-State Circuits* 38.6 (June 2003), pp. 958–965. ISSN: 0018-9200. DOI: `10.1109/JSSC.2003.811979`. URL: `http://ieeexplore.ieee.org/document/1201998/`.

[81] Carolina Mora Lopez et al. “A 966-electrode neural probe with 384 configurable channels in 0.13µm SOI CMOS”. In: *2016 IEEE Int. Solid-State Circuits Conf.* Vol. 59. IEEE, Jan. 2016, pp. 392–393. ISBN: 978-1-4673-9466-6. DOI: `10.1109/ISSCC.2016.7418072`. URL: `http://ieeexplore.ieee.org/document/7418072/`.

[82] Ben Johnson et al. “A 50µm pitch, 1120-channel, 20kHz frame rate microelectrode array for slice recording”. In: *2013 IEEE Biomed. Circuits Syst. Conf.* IEEE, Oct. 2013, pp. 109–112. ISBN: 978-1-4799-1471-5. DOI: 10.1109/BioCAS.2013.6679651. URL: http://ieeexplore.ieee.org/document/6679651/.

[83] Sung-Yun Park, Jihyun Cho, and Euisik Yoon. “3.37 μW/Ch modular scalable neural recording system with embedded lossless compression for dynamic power reduction”. In: *2017 Symp. VLSI Circuits.* IEEE, June 2017, pp. C168–C169. ISBN: 978-4-86348-614-0. DOI: 10.23919/VLSIC.2017.8008468. URL: http://ieeexplore.ieee.org/document/8008468/.

[84] R.H. Olsson III et al. “Band-Tunable and Multiplexed Integrated Circuits for Simultaneous Recording and Stimulation With Microelectrode Arrays”. In: *IEEE Trans. Biomed. Eng.* 52.7 (July 2005), pp. 1303–1311. ISSN: 0018-9294. DOI: 10.1109/TBME.2005.847540. URL: http://ieeexplore.ieee.org/document/1440609/.

[85] Sina Basir-Kazeruni et al. “A blind Adaptive Stimulation Artifact Rejection (ASAR) engine for closed-loop implantable neuromodulation systems”. In: *2017 8th Int. IEEE/EMBS Conf. Neural Eng.* IEEE, May 2017, pp. 186–189. ISBN: 978-1-5090-4603-4. DOI: 10.1109/NER.2017.8008322. URL: http://ieeexplore.ieee.org/document/8008322/.

[86] Ming Yin et al. “Wireless Neurosensor for Full-Spectrum Electrophysiology Recordings during Free Behavior”. In: *Neuron* 84.6 (Dec. 2014), pp. 1170–1182. ISSN: 08966273. DOI: 10.1016/j.neuron.2014.11.010. URL: https://linkinghub.elsevier.com/retrieve/pii/S0896627314010101.

[87] Hua Gao et al. “HermesE: A 96-Channel Full Data Rate Direct Neural Interface in 0.13 µm CMOS”. In: *IEEE J. Solid-State Circuits* 47.4 (Apr. 2012), pp. 1043–1055. ISSN: 0018-9200. DOI: 10.1109/JSSC.2012.2185338. URL: http://ieeexplore.ieee.org/document/6158616/.

[88] Stavros Zanos et al. “The Neurochip-2: An Autonomous Head-Fixed Computer for Recording and Stimulating in Freely Behaving Monkeys”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 19.4 (Aug. 2011), pp. 427–435. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2011.2158007. URL: http://ieeexplore.ieee.org/document/5782994/.

[89] Xilin Liu et al. “The PennBMBI: A general purpose wireless Brain-Machine-Brain Interface system for unrestrained animals”. In: *2014 IEEE Int. Symp. Circuits Syst.* IEEE, June 2014, pp. 650–653. ISBN: 978-1-4799-3432-4. DOI: 10.1109/ISCAS.2014.6865219. URL: http://ieeexplore.ieee.org/document/6865219/.

[90] Arezu Bagheri et al. “Massively-Parallel Neuromonitoring and Neurostimulation Rodent Headset With Nanotextured Flexible Microelectrodes”. In: *IEEE Trans. Biomed. Circuits Syst.* 7.5 (Oct. 2013), pp. 601–609. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2013.2281772. URL: http://ieeexplore.ieee.org/document/6631481/.

[91] Hossein Kassiri et al. “Closed-Loop Neurostimulators: A Survey and A Seizure-Predicting Design Example for Intractable Epilepsy Treatment”. In: *IEEE Trans. Biomed. Circuits Syst.* 11.5 (Oct. 2017), pp. 1026–1040. ISSN: 1932-4545. DOI: 10.1109/TBCAS.2017.2694638. URL: http://ieeexplore.ieee.org/document/7982670/.

[92] Karim Abdelhalim et al. “64-Channel UWB Wireless Neural Vector Analyzer SOC With a Closed-Loop Phase Synchrony-Triggered Neurostimulator”. In: *IEEE J. Solid-State Circuits* 48.10 (Oct. 2013), pp. 2494–2510. ISSN: 0018-9200. DOI: 10.1109/JSSC.2013.2272952. URL: http://ieeexplore.ieee.org/document/6572893/.

[93] Hossein Kassiri et al. “Rail-to-Rail-Input Dual-Radio 64-Channel Closed-Loop Neurostimulator”. In: *IEEE J. Solid-State Circuits* 52.11 (2017), pp. 1–18. ISSN: 0018-9200. DOI: 10.1109/JSSC.2017.2749426. URL: http://ieeexplore.ieee.org/document/8068946/.

[94] Hossein Kassiri et al. “Battery-less Tri-band-Radio Neuro-monitor and Responsive Neurostimulator for Diagnostics and Treatment of Neurological Disorders”. In: *IEEE J. Solid-State Circuits* 51.5 (May 2016), pp. 1274–1289. ISSN: 0018-9200. DOI: 10.1109/JSSC.2016.2528999. URL: http://ieeexplore.ieee.org/document/7433928/.

[95] Hyo-Gyuem Rhew et al. “A Fully Self-Contained Logarithmic Closed-Loop Deep Brain Stimulation SoC With Wireless Telemetry and Wireless Power Management”. In: *IEEE J. Solid-State Circuits* 49.10 (Oct. 2014), pp. 2213–2227. ISSN: 0018-9200. DOI: 10.1109/JSSC.2014.2346779. URL: https://ieeexplore.ieee.org/document/6887365.

[96] Wei-Ming Chen et al. “A Fully Integrated 8-Channel Closed-Loop Neural-Prosthetic CMOS SoC for Real-Time Epileptic Seizure Control”. In: *IEEE J. Solid-State Circuits* 49.1 (Jan. 2014), pp. 232–247. ISSN: 0018-9200. DOI: 10.1109/JSSC.2013.2284346. URL: http://ieeexplore.ieee.org/document/6637111/.

[97] Nicole C. Swann et al. “Gamma Oscillations in the Hyperkinetic State Detected with Chronic Human Brain Recordings in Parkinson’s Disease”. In: *J. Neurosci.* 36.24 (June 2016), pp. 6445–6458. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.1128-16.2016. URL: https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.1128-16.2016.

[98] A. A. Kuhn et al. “High-Frequency Stimulation of the Subthalamic Nucleus Suppresses Oscillatory Activity in Patients with Parkinson’s Disease in Parallel with Improvement in Motor Performance”. In: *J. Neurosci.* 28.24 (June 2008), pp. 6165–6173. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.0282-08.2008. URL: https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.0282-08.2008.

[99] Fumiaki Yoshida et al. "Value of subthalamic nucleus local field potentials recordings in predicting stimulation parameters for deep brain stimulation in Parkinson's disease". In: *J. Neurol. Neurosurg. Psychiatry* 81.8 (Aug. 2010), pp. 885–889. ISSN: 0022-3050. DOI: 10.1136/jnnp.2009.190918. URL: https://jnnp.bmj.com/lookup/doi/10.1136/jnnp.2009.190918.

[100] Moran Weinberger et al. "Beta Oscillatory Activity in the Subthalamic Nucleus and Its Relation to Dopaminergic Response in Parkinson's Disease". In: *J. Neurophysiol.* 96.6 (Dec. 2006), pp. 3248–3256. ISSN: 0022-3077. DOI: 10.1152/jn.00697.2006. URL: https://www.physiology.org/doi/10.1152/jn.00697.2006.

[101] Casey H. Halpern et al. "Deep brain stimulation for epilepsy". In: *Neurotherapeutics* 5.1 (Jan. 2008), pp. 59–67. ISSN: 1933-7213. DOI: 10.1016/j.nurt.2007.10.065. URL: http://link.springer.com/10.1016/j.nurt.2007.10.065.

[102] Kelvin So et al. "Subject-specific modulation of local field potential spectral power during brain–machine interface control in primates". In: *J. Neural Eng.* 11.2 (Apr. 2014), p. 026002. ISSN: 1741-2560. DOI: 10.1088/1741-2560/11/2/026002. URL: https://iopscience.iop.org/article/10.1088/1741-2560/11/2/026002.

[103] Robert D. Flint et al. "Accurate decoding of reaching movements from field potentials in the absence of spikes". In: *J. Neural Eng.* 9.4 (Aug. 2012), p. 046006. ISSN: 1741-2560. DOI: 10.1088/1741-2560/9/4/046006. URL: https://iopscience.iop.org/article/10.1088/1741-2560/9/4/046006.

[104] Robert D. Flint et al. "Local field potentials allow accurate decoding of muscle activity". In: *J. Neurophysiol.* 108.1 (July 2012), pp. 18–24. ISSN: 0022-3077. DOI: 10.1152/jn.00832.2011. URL: https://www.physiology.org/doi/10.1152/jn.00832.2011.

[105] Preeya Khanna and Jose M. Carmena. "Beta band oscillations in motor cortex reflect neural population signals that delay movement onset". In: *Elife* 6 (May 2017). ISSN: 2050-084X. DOI: 10.7554/eLife.24573. URL: https://elifesciences.org/articles/24573.

[106] Doug Rubino, Kay A. Robbins, and Nicholas G. Hatsopoulos. "Propagating waves mediate information transfer in the motor cortex". In: *Nat. Neurosci.* 9.12 (Dec. 2006), pp. 1549–1557. ISSN: 1097-6256. DOI: 10.1038/nn1802. URL: http://www.nature.com/articles/nn1802.

[107] J. N. Sanes and J. P. Donoghue. "Oscillations in local field potentials of the primate motor cortex during voluntary movement." In: *Proc. Natl. Acad. Sci.* 90.10 (May 1993), pp. 4470–4474. ISSN: 0027-8424. DOI: 10.1073/pnas.90.10.4470. URL: http://www.pnas.org/cgi/doi/10.1073/pnas.90.10.4470.

[108] Carsten Mehring et al. "Inference of hand movements from local field potentials in monkey motor cortex". In: *Nat. Neurosci.* 6.12 (Dec. 2003), pp. 1253–1254. ISSN: 1097-6256. DOI: 10.1038/nn1158. URL: http://www.nature.com/articles/nn1158.

[109] Daniel K. Leventhal et al. "Basal Ganglia Beta Oscillations Accompany Cue Utilization". In: *Neuron* 73.3 (Feb. 2012), pp. 523–536. ISSN: 08966273. DOI: 10.1016/j.neuron.2011.11.032. URL: https://linkinghub.elsevier.com/retrieve/pii/S0896627312000360.

[110] Mikael Lundqvist et al. "Gamma and Beta Bursts Underlie Working Memory". In: *Neuron* 90.1 (Apr. 2016), pp. 152–164. ISSN: 08966273. DOI: 10.1016/j.neuron.2016.02.028. URL: https://linkinghub.elsevier.com/retrieve/pii/S0896627316001458.

[111] Bijan Pesaran et al. "Temporal structure in neuronal activity during working memory in macaque parietal cortex". In: *Nat. Neurosci.* 5.8 (Aug. 2002), pp. 805–811. ISSN: 1097-6256. DOI: 10.1038/nn890. URL: http://www.nature.com/articles/nn890.

[112] Maria C. Dadarlat, Joseph E. O'Doherty, and Philip N. Sabes. "A learning-based approach to artificial sensory feedback leads to optimal integration". In: *Nat. Neurosci.* 18.1 (Jan. 2015), pp. 138–144. ISSN: 1097-6256. DOI: 10.1038/nn.3883. URL: http://www.nature.com/articles/nn.3883.

[113] Miguel Pais-Vieira et al. "A Brain-to-Brain Interface for Real-Time Sharing of Sensorimotor Information". In: *Sci. Rep.* 3.1 (Dec. 2013), p. 1319. ISSN: 2045-2322. DOI: 10.1038/srep01319. URL: http://www.nature.com/articles/srep01319.

[114] N. A. Fitzsimmons et al. "Primate Reaching Cued by Multichannel Spatiotemporal Cortical Microstimulation". In: *J. Neurosci.* 27.21 (May 2007), pp. 5593–5602. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.5297-06.2007. URL: https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.5297-06.2007.

[115] Joseph E. O'Doherty et al. "Active tactile exploration using a brain–machine–brain interface". In: *Nature* 479.7372 (Nov. 2011), pp. 228–231. ISSN: 0028-0836. DOI: 10.1038/nature10489. URL: http://www.nature.com/articles/nature10489.

[116] Joseph E. O'Doherty et al. "Virtual Active Touch Using Randomly Patterned Intracortical Microstimulation". In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 20.1 (Jan. 2012), pp. 85–93. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2011.2166807. URL: http://ieeexplore.ieee.org/document/6114258/.

[117] Boubker Zaaimi et al. "Multi-electrode stimulation in somatosensory cortex increases probability of detection". In: *J. Neural Eng.* 10.5 (Oct. 2013), p. 056013. ISSN: 1741-2560. DOI: 10.1088/1741-2560/10/5/056013. URL: https://iopscience.iop.org/article/10.1088/1741-2560/10/5/056013.

[118] Ali Moin et al. "Powering and communication for OMNI: A distributed and modular closed-loop neuromodulation device". In: *2016 38th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* Vol. 2016-Octob. IEEE, Aug. 2016, pp. 4471–4474. ISBN: 978-1-4577-0220-4. DOI: 10.1109/EMBC.2016.7591720. URL: http://ieeexplore.ieee.org/document/7591720/.

[119] Ryan T. Canolty, Karunesh Ganguly, and Jose M. Carmena. “Task-Dependent Changes in Cross-Level Coupling between Single Neurons and Oscillatory Activity in Multiscale Networks”. In: *PLoS Comput. Biol.* 8.12 (Dec. 2012). Ed. by Olaf Sporns, e1002809. ISSN: 1553-7358. DOI: 10.1371/journal.pcbi.1002809. URL: https://dx.plos.org/10.1371/journal.pcbi.1002809.

[120] Maryam Saleh et al. “Fast and Slow Oscillations in Human Primary Motor Cortex Predict Oncoming Behaviorally Relevant Cues”. In: *Neuron* 65.4 (Feb. 2010), pp. 461–471. ISSN: 08966273. DOI: 10.1016/j.neuron.2010.02.001. URL: https://linkinghub.elsevier.com/retrieve/pii/S0896627310000619.

[121] Annette Bastian, Gregor Schoner, and Alexa Riehle. “Preshaping and continuous evolution of motor cortical representations during movement preparation”. In: *Eur. J. Neurosci.* 18.7 (Oct. 2003), pp. 2047–2058. ISSN: 0953-816X. DOI: 10.1046/j.1460-9568.2003.02906.x. URL: http://doi.wiley.com/10.1046/j.1460-9568.2003.02906.x.

[122] Mark M. Churchland. “Neural Variability in Premotor Cortex Provides a Signature of Motor Preparation”. In: *J. Neurosci.* 26.14 (Apr. 2006), pp. 3697–3712. ISSN: 0270-6474. DOI: 10.1523/JNEUROSCI.3762-05.2006. URL: https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.3762-05.2006.

[123] Mark M. Churchland and Krishna V. Shenoy. “Delay of Movement Caused by Disruption of Cortical Preparatory Activity”. In: *J. Neurophysiol.* 97.1 (Jan. 2007), pp. 348–359. ISSN: 0022-3077. DOI: 10.1152/jn.00808.2006. URL: https://www.physiology.org/doi/10.1152/jn.00808.2006.

[124] John P. Donoghue et al. “Neural Discharge and Local Field Potential Oscillations in Primate Motor Cortex During Voluntary Movements”. In: *J. Neurophysiol.* 79.1 (Jan. 1998), pp. 159–173. ISSN: 0022-3077. DOI: 10.1152/jn.1998.79.1.159. URL: https://www.physiology.org/doi/10.1152/jn.1998.79.1.159.

[125] Jose L. Cantero, Mercedes Atienza, and Rosa M. Salas. “Human alpha oscillations in wakefulness, drowsiness period, and REM sleep: different electroencephalographic phenomena within the alpha band”. In: *Neurophysiol. Clin. Neurophysiol.* 32.1 (Jan. 2002), pp. 54–71. ISSN: 09877053. DOI: 10.1016/S0987-7053(01)00289-1. URL: https://linkinghub.elsevier.com/retrieve/pii/S0987705301002891.

[126] Hartmut Schulz. “Rethinking Sleep Analysis”. In: *J. Clin. Sleep Med.* 04.02 (Apr. 2008), pp. 99–103. ISSN: 1550-9389. DOI: 10.5664/jcsm.27124. URL: http://jcsm.aasm.org/doi/10.5664/jcsm.27124.

[127] A. G. Rouse et al. “A chronic generalized bi-directional brain–machine interface”. In: *J. Neural Eng.* 8.3 (June 2011), p. 036018. ISSN: 1741-2560. DOI: 10.1088/1741-2560/8/3/036018. URL: https://iopscience.iop.org/article/10.1088/1741-2560/8/3/036018.

[128] Al-Thaddeus Avestruz et al. “A 5 μW/Channel Spectral Analysis IC for Chronic Bidirectional Brain–Machine Interfaces”. In: *IEEE J. Solid-State Circuits* 43.12 (Dec. 2008), pp. 3006–3024. ISSN: 0018-9200. DOI: 10.1109/JSSC.2008.2006460. URL: http://ieeexplore.ieee.org/document/4684654/.

[129] M. Tariqus Salam, Jose Luis Perez Velazquez, and Roman Genov. “Seizure Suppression Efficacy of Closed-Loop Versus Open-Loop Deep Brain Stimulation in a Rodent Model of Epilepsy”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 24.6 (June 2016), pp. 710–719. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2015.2498973. URL: http://ieeexplore.ieee.org/document/7323863/.

[130] Barbara Jobst and George Thomas. “Critical review of the responsive neurostimulator system for epilepsy”. In: *Med. Devices Evid. Res.* 8 (Oct. 2015), p. 405. ISSN: 1179-1470. DOI: 10.2147/MDER.S62853. URL: https://www.dovepress.com/critical-review-of-the-responsive-neurostimulator-system-for-epilepsy-peer-reviewed-article-MDER.

[131] Scott F. Lempka et al. “Characterization of the stimulus waveforms generated by implantable pulse generators for deep brain stimulation”. In: *Clin. Neurophysiol.* 129.4 (Apr. 2018), pp. 731–742. ISSN: 13882457. DOI: 10.1016/j.clinph.2018.01.015. URL: https://linkinghub.elsevier.com/retrieve/pii/S1388245718300300.

[132] M. Reza Pazhouhandeh et al. “Track-and-Zoom Neural Analog-to-Digital Converter With Blind Stimulation Artifact Rejection”. In: *IEEE J. Solid-State Circuits* 55.7 (July 2020), pp. 1984–1997. ISSN: 0018-9200. DOI: 10.1109/JSSC.2020.2991526. URL: https://ieeexplore.ieee.org/document/9091529/.

[133] John P. Uehlin et al. “A Single-Chip Bidirectional Neural Interface With High-Voltage Stimulation and Adaptive Artifact Cancellation in Standard CMOS”. In: *IEEE J. Solid-State Circuits* 55.7 (July 2020), pp. 1749–1761. ISSN: 0018-9200. DOI: 10.1109/JSSC.2020.2991524. URL: https://ieeexplore.ieee.org/document/9094199/.

[134] Ali Moin et al. “An EMG Gesture Recognition System with Flexible High-Density Sensors and Brain-Inspired High-Dimensional Classifier”. In: *2018 IEEE Int. Symp. Circuits Syst.* Vol. 2018-May. IEEE, Apr. 2018, pp. 1–5. ISBN: 978-1-5386-4881-0. DOI: 10.1109/ISCAS.2018.8351613. URL: https://ieeexplore.ieee.org/document/8351613/.

[135] Ali Moin et al. “A wearable biosensing system with in-sensor adaptive machine learning for hand gesture recognition”. In: *Nat. Electron.* 4.1 (Jan. 2021), pp. 54–63. ISSN: 2520-1131. DOI: 10.1038/s41928-020-00510-8. URL: https://doi.org/10.1038/s41928-020-00510-8%20http://www.nature.com/articles/s41928-020-00510-8.

[136] Simone Benatti et al. "A Prosthetic Hand Body Area Controller Based on Efficient Pattern Recognition Control Strategies". In: *Sensors* 17.4 (Apr. 2017), p. 869. ISSN: 1424-8220. DOI: 10.3390/s17040869. URL: http://www.mdpi.com/1424-8220/17/4/869.

[137] Xu Zhang et al. "Hand gesture recognition and virtual game control based on 3D accelerometer and EMG sensors". In: *Proc. 14th Int. Conf. Intell. user interfaces*. New York, NY, USA: ACM, Feb. 2009, pp. 401–406. ISBN: 9781605581682. DOI: 10.1145/1502650.1502708. URL: https://dl.acm.org/doi/10.1145/1502650.1502708.

[138] Giulio Marin, Fabio Dominio, and Pietro Zanuttigh. "Hand gesture recognition with leap motion and kinect devices". In: *2014 IEEE Int. Conf. Image Process.* IEEE, Oct. 2014, pp. 1565–1569. ISBN: 978-1-4799-5751-4. DOI: 10.1109/ICIP.2014.7025313. URL: http://ieeexplore.ieee.org/document/7025313/.

[139] L. Almeida et al. "Towards natural interaction in immersive reality with a cyberglove". In: *2019 IEEE Int. Conf. Syst. Man Cybern.* Vol. 2019-Octob. IEEE, Oct. 2019, pp. 2653–2658. ISBN: 978-1-7281-4569-3. DOI: 10.1109/SMC.2019.8914239. URL: https://ieeexplore.ieee.org/document/8914239/.

[140] *MoCap Glove System*. MoCapGlove.10.12.V2. CyberGlove Systems. 2010.

[141] Ji-Hae Kim et al. "deepGesture: Deep learning-based gesture recognition scheme using motion sensors". In: *Displays* 55 (Dec. 2018), pp. 38–45. ISSN: 01419382. DOI: 10.1016/j.displa.2018.08.001. URL: https://linkinghub.elsevier.com/retrieve/pii/S0141938217302032.

[142] Eleanor Criswell and Jeffrey R Cram. *Cram's Introduction to Surface Electromyography*. 2nd. Sudbury, MA: Jones and Bartlett, 2011.

[143] Max Ortiz-Catalan, B. Hakansson, and R. Branemark. "An osseointegrated human-machine gateway for long-term sensory feedback and motor control of artificial limbs". In: *Sci. Transl. Med.* 6.257 (Oct. 2014), 257re6–257re6. ISSN: 1946-6234. DOI: 10.1126/scitranslmed.3008933. URL: https://stm.sciencemag.org/lookup/doi/10.1126/scitranslmed.3008933.

[144] Pierre Morel et al. "Long-term decoding of movement force and direction with a wireless myoelectric implant". In: *J. Neural Eng.* 13.1 (Feb. 2016), p. 016002. ISSN: 1741-2560. DOI: 10.1088/1741-2560/13/1/016002. URL: https://iopscience.iop.org/article/10.1088/1741-2560/13/1/016002.

[145] Soren Lewis et al. "Fully Implantable Multi-Channel Measurement System for Acquisition of Muscle Activity". In: *IEEE Trans. Instrum. Meas.* 62.7 (July 2013), pp. 1972–1981. ISSN: 0018-9456. DOI: 10.1109/TIM.2013.2253992. URL: http://ieeexplore.ieee.org/document/6514117/.

[146] Richard F. Weir et al. "Implantable Myoelectric Sensors (IMESs) for Intramuscular Electromyogram Recording". In: *IEEE Trans. Biomed. Eng.* 56.1 (Jan. 2009), pp. 159–171. ISSN: 0018-9294. DOI: 10.1109/TBME.2008.2005942. URL: http://ieeexplore.ieee.org/document/4633666/.

[147] T. Scott Saponas et al. "Making muscle-computer interfaces more practical". In: *Proc. 28th Int. Conf. Hum. factors Comput. Syst. - CHI '10*. Vol. 2. New York, New York, USA: ACM Press, 2010, p. 851. ISBN: 9781605589299. DOI: 10.1145/1753326.1753451. URL: http://portal.acm.org/citation.cfm?doid=1753326.1753451.

[148] Yu Du et al. "Surface EMG-Based Inter-Session Gesture Recognition Enhanced by Deep Domain Adaptation". In: *Sensors* 17.3 (Feb. 2017), p. 458. ISSN: 1424-8220. DOI: 10.3390/s17030458. URL: http://www.mdpi.com/1424-8220/17/3/458.

[149] Christoph Amma et al. "Advancing Muscle-Computer Interfaces with High-Density Electromyography". In: *Proc. 33rd Annu. ACM Conf. Hum. Factors Comput. Syst.* Vol. 2015-April. New York, NY, USA: ACM, Apr. 2015, pp. 929–938. ISBN: 9781450331456. DOI: 10.1145/2702123.2702501. URL: https://dl.acm.org/doi/10.1145/2702123.2702501.

[150] Gea Drost et al. "Clinical applications of high-density surface EMG: A systematic review". In: *J. Electromyogr. Kinesiol.* 16.6 (Dec. 2006), pp. 586–602. ISSN: 10506411. DOI: 10.1016/j.jelekin.2006.09.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S1050641106001209.

[151] Stegeman Dick F. et al. "High-density Surface EMG: Techniques and Applications at a Motor Unit Level". In: *Biocybern. Biomed. Eng.* 32.3 (Jan. 2012), pp. 3–27. ISSN: 02085216. DOI: 10.1016/S0208-5216(12)70039-6. URL: https://linkinghub.elsevier.com/retrieve/pii/S0208521612700396.

[152] Maoqi Chen and Ping Zhou. "A Novel Framework Based on FastICA for High Density Surface EMG Decomposition". In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 24.1 (Jan. 2016), pp. 117–127. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2015.2412038. URL: https://ieeexplore.ieee.org/document/7058391/.

[153] Dario Farina et al. "Principles of Motor Unit Physiology Evolve With Advances in Technology". In: *Physiology* 31.2 (Mar. 2016), pp. 83–94. ISSN: 1548-9213. DOI: 10.1152/physiol.00040.2015. URL: https://www.physiology.org/doi/10.1152/physiol.00040.2015.

[154] Bert U. Kleine et al. "Using two-dimensional spatial information in decomposition of surface EMG signals". In: *J. Electromyogr. Kinesiol.* 17.5 (Oct. 2007), pp. 535–548. ISSN: 10506411. DOI: 10.1016/j.jelekin.2006.05.003. URL: https://linkinghub.elsevier.com/retrieve/pii/S1050641106000800.

[155] Vojko Glaser, Ales Holobar, and Damjan Zazula. “Real-Time Motor Unit Identification From High-Density Surface EMG”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 21.6 (Nov. 2013), pp. 949–958. ISSN: 1534-4320. DOI: `10.1109/TNSRE.2013.2247631`. URL: `https://ieeexplore.ieee.org/document/6475191/`.

[156] T. Walley Williams. “Practical Methods for Controlling Powered Upper-Extremity Prostheses”. In: *Assist. Technol.* 2.1 (Mar. 1990), pp. 3–18. ISSN: 1040-0435. DOI: `10.1080/10400435.1990.10132142`. URL: `http://www.tandfonline.com/doi/abs/10.1080/10400435.1990.10132142`.

[157] Dario Farina et al. “The Extraction of Neural Information from the Surface EMG for the Control of Upper-Limb Prostheses: Emerging Avenues and Challenges”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 22.4 (July 2014), pp. 797–809. ISSN: 1534-4320. DOI: `10.1109/TNSRE.2014.2305111`. URL: `https://ieeexplore.ieee.org/document/6737308/`.

[158] Dario Farina et al. “Decoding the neural drive to muscles from the surface electromyogram”. In: *Clin. Neurophysiol.* 121.10 (Oct. 2010), pp. 1616–1623. ISSN: 13882457. DOI: `10.1016/j.clinph.2009.10.040`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1388245710003457`.

[159] Ning Jiang, K.B. Englehart, and P.A. Parker. “Extracting Simultaneous and Proportional Neural Control Information for Multiple-DOF Prostheses From the Surface Electromyographic Signal”. In: *IEEE Trans. Biomed. Eng.* 56.4 (Apr. 2009), pp. 1070–1080. ISSN: 0018-9294. DOI: `10.1109/TBME.2008.2007967`. URL: `http://ieeexplore.ieee.org/document/4663628/`.

[160] Yikun Gu et al. “Robust EMG pattern recognition in the presence of confounding factors: features, classifiers and adaptive learning”. In: *Expert Syst. Appl.* 96 (Apr. 2018), pp. 208–217. ISSN: 09574174. DOI: `10.1016/j.eswa.2017.11.049`. URL: `https://doi.org/10.1016/j.eswa.2017.11.049%20https://linkinghub.elsevier.com/retrieve/pii/S0957417417308060`.

[161] M.A. Oskoei and Huosheng Hu. “Support Vector Machine-Based Classification Scheme for Myoelectric Control Applied to Upper Limb”. In: *IEEE Trans. Biomed. Eng.* 55.8 (Aug. 2008), pp. 1956–1965. ISSN: 0018-9294. DOI: `10.1109/TBME.2008.919734`. URL: `http://ieeexplore.ieee.org/document/4463647/`.

[162] Simone Benatti et al. “A Versatile Embedded Platform for EMG Acquisition and Gesture Recognition”. In: *IEEE Trans. Biomed. Circuits Syst.* 9.5 (Oct. 2015), pp. 620–630. ISSN: 1932-4545. DOI: `10.1109/TBCAS.2015.2476555`. URL: `http://ieeexplore.ieee.org/document/7303979/`.

[163] Karthik Sivarama Krishnan et al. “Recognition of human arm gestures using Myo armband for the game of hand cricket”. In: *2017 IEEE Int. Symp. Robot. Intell. Sensors*. IEEE, Oct. 2017, pp. 389–394. ISBN: 978-1-5386-1342-9. DOI: `10.1109/IRIS.2017.8250154`. URL: `http://ieeexplore.ieee.org/document/8250154/`.

[164] Haoshi Zhang et al. “An adaptation strategy of using LDA classifier for EMG pattern recognition”. In: *2013 35th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* IEEE, July 2013, pp. 4267–4270. ISBN: 978-1-4577-0216-7. DOI: 10.1109/EMBC.2013.6610488. URL: http://ieeexplore.ieee.org/document/6610488/.

[165] Weidong Geng et al. “Gesture recognition by instantaneous surface EMG images”. In: *Sci. Rep.* 6.1 (Dec. 2016), p. 36571. ISSN: 2045-2322. DOI: 10.1038/srep36571. URL: http://www.nature.com/articles/srep36571.

[166] Xilin Liu et al. “The Virtual Trackpad: An Electromyography-Based, Wireless, Real-Time, Low-Power, Embedded Hand-Gesture-Recognition System Using an Event-Driven Artificial Neural Network”. In: *IEEE Trans. Circuits Syst. II Express Briefs* 64.11 (Nov. 2017), pp. 1257–1261. ISSN: 1549-7747. DOI: 10.1109/TCSII.2016.2635674. URL: https://ieeexplore.ieee.org/document/7769179/.

[167] Abbas Rahimi et al. “Hyperdimensional biosignal processing: A case study for EMG-based hand gesture recognition”. In: *2016 IEEE Int. Conf. Rebooting Comput.* IEEE, Oct. 2016, pp. 1–8. ISBN: 978-1-5090-1370-8. DOI: 10.1109/ICRC.2016.7738683. URL: http://ieeexplore.ieee.org/document/7738683/.

[168] Andy Zhou, Rikky Muller, and Jan Rabaey. “Memory-Efficient, Limb Position-Aware Hand Gesture Recognition using Hyperdimensional Computing”. In: *Proc. TinyML Res. Symp. (TinyML Res. Symp. ’21).* New York, NY, USA, 2021.

[169] Aaron J. Young, Levi J. Hargrove, and Todd A. Kuiken. “The Effects of Electrode Size and Orientation on the Sensitivity of Myoelectric Pattern Recognition Systems to Electrode Shift”. In: *IEEE Trans. Biomed. Eng.* 58.9 (Sept. 2011), pp. 2537–2544. ISSN: 0018-9294. DOI: 10.1109/TBME.2011.2159216. URL: http://ieeexplore.ieee.org/document/5872014/.

[170] Xiaorong Zhang and He Huang. “A real-time, practical sensor fault-tolerant module for robust EMG pattern recognition”. In: *J. Neuroeng. Rehabil.* 12.1 (2015), p. 18. ISSN: 1743-0003. DOI: 10.1186/s12984-015-0011-y. URL: http://www.jneuroengrehab.com/content/12/1/18.

[171] Levi Hargrove, Kevin Englehart, and Bernard Hudgins. “A training strategy to reduce classification degradation due to electrode displacements in pattern recognition based myoelectric control”. In: *Biomed. Signal Process. Control* 3.2 (Apr. 2008), pp. 175–180. ISSN: 17468094. DOI: 10.1016/j.bspc.2007.11.005. URL: https://linkinghub.elsevier.com/retrieve/pii/S1746809407001012.

[172] Dennis Tkach, He Huang, and Todd A. Kuiken. “Study of stability of time-domain features for electromyographic pattern recognition”. In: *J. Neuroeng. Rehabil.* 7.1 (2010), p. 21. ISSN: 1743-0003. DOI: 10.1186/1743-0003-7-21. URL: http://jneuroengrehab.biomedcentral.com/articles/10.1186/1743-0003-7-21.

[173] Bojan Milosevic, Elisabetta Farella, and Simone Benatti. "Exploring Arm Posture and Temporal Variability in Myoelectric Hand Gesture Recognition". In: *2018 7th IEEE Int. Conf. Biomed. Robot. Biomechatronics*. Vol. 2018-Augus. IEEE, Aug. 2018, pp. 1032–1037. ISBN: 978-1-5386-8183-1. DOI: 10.1109/BIOROB.2018.8487838. URL: https://ieeexplore.ieee.org/document/8487838/.

[174] Hui-Wang Cui et al. "Ultra-fast photonic curing of electrically conductive adhesives fabricated from vinyl ester resin and silver micro-flakes for printed electronics". In: *RSC Adv.* 4.31 (2014), pp. 15914–15922. ISSN: 2046-2069. DOI: 10.1039/C4RA00292J. URL: http://xlink.rsc.org/?DOI=C4RA00292J.

[175] Giacinto Luigi Cerone, Alberto Botter, and Marco Gazzoni. "A Modular, Smart, and Wearable System for High Density sEMG Detection". In: *IEEE Trans. Biomed. Eng.* 66.12 (Dec. 2019), pp. 3371–3380. ISSN: 0018-9294. DOI: 10.1109/TBME.2019.2904398. URL: https://ieeexplore.ieee.org/document/8664610/.

[176] B. G. Lapatki et al. "A thin, flexible multielectrode grid for high-density surface EMG". In: *J. Appl. Physiol.* 96.1 (Jan. 2004), pp. 327–336. ISSN: 8750-7587. DOI: 10.1152/japplphysiol.00521.2003. URL: https://www.physiology.org/doi/10.1152/japplphysiol.00521.2003.

[177] Manfredo Atzori et al. "Electromyography data for non-invasive naturally-controlled robotic hand prostheses". In: *Sci. Data* 1.1 (Dec. 2014), p. 140053. ISSN: 2052-4463. DOI: 10.1038/sdata.2014.53. URL: http://www.nature.com/articles/sdata201453.

[178] Manfredo Atzori et al. "Building the Ninapro database: A resource for the biorobotics community". In: *2012 4th IEEE RAS EMBS Int. Conf. Biomed. Robot. Biomechatronics*. IEEE, June 2012, pp. 1258–1265. ISBN: 978-1-4577-1200-5. DOI: 10.1109/BioRob.2012.6290287. URL: http://ieeexplore.ieee.org/document/6290287/.

[179] Stefano Pizzolato et al. "Comparison of six electromyography acquisition setups on hand movement classification tasks". In: *PLoS One* 12.10 (Oct. 2017). Ed. by Dingguo Zhang, e0186132. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0186132. URL: https://dx.plos.org/10.1371/journal.pone.0186132.

[180] Pentti Kanerva. "Hyperdimensional Computing: An Introduction to Computing in Distributed Representation with High-Dimensional Random Vectors". In: *Cognit. Comput.* 1.2 (June 2009), pp. 139–159. ISSN: 1866-9956. DOI: 10.1007/s12559-009-9009-8. URL: http://link.springer.com/10.1007/s12559-009-9009-8.

[181] Abbas Rahimi et al. "Hyperdimensional Computing for Blind and One-Shot Classification of EEG Error-Related Potentials". In: *Mob. Networks Appl.* 25.5 (Oct. 2020), pp. 1958–1969. ISSN: 1383-469X. DOI: 10.1007/s11036-017-0942-6. URL: http://link.springer.com/10.1007/s11036-017-0942-6.

[182] Alessio Burrello et al. "One-shot Learning for iEEG Seizure Detection Using End-to-end Binary Operations: Local Binary Patterns with Hyperdimensional Computing". In: *2018 IEEE Biomed. Circuits Syst. Conf.* IEEE, Oct. 2018, pp. 1–4. ISBN: 978-1-5386-3603-9. DOI: `10.1109/BIOCAS.2018.8584751`. URL: `https://ieeexplore.ieee.org/document/8584751/`.

[183] Abbas Rahimi et al. "High-Dimensional Computing as a Nanoscalable Paradigm". In: *IEEE Trans. Circuits Syst. I Regul. Pap.* 64.9 (Sept. 2017), pp. 2508–2521. ISSN: 1549-8328. DOI: `10.1109/TCSI.2017.2705051`. URL: `http://ieeexplore.ieee.org/document/7942066/`.

[184] Angkoon Phinyomark et al. "Navigating features: a topologically informed chart of electromyographic features space". In: *J. R. Soc. Interface* 14.137 (Dec. 2017), p. 20170734. ISSN: 1742-5689. DOI: `10.1098/rsif.2017.0734`. URL: `http://dx.doi.org/10.1098/rsif.2017.0734%20https://dx.doi.org/10.6084/m9.%20https://royalsocietypublishing.org/doi/10.1098/rsif.2017.0734`.

[185] Lauren 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". In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 19.2 (Apr. 2011), pp. 186–192. ISSN: 1534-4320. DOI: `10.1109/TNSRE.2010.2100828`. URL: `http://ieeexplore.ieee.org/document/5676233/%20https://ieeexplore.ieee.org/document/5676233/`.

[186] K. Englehart and B. Hudgins. "A robust, real-time control scheme for multifunction myoelectric control". In: *IEEE Trans. Biomed. Eng.* 50.7 (July 2003), pp. 848–854. ISSN: 0018-9294. DOI: `10.1109/TBME.2003.813539`. URL: `http://ieeexplore.ieee.org/document/1206493/`.

[187] Quanquan Gu, Zhenhui Li, and Jiawei Han. "Generalized fisher score for feature selection". In: *Proc. 27th Conf. Uncertain. Artif. Intell. UAI 2011.* 2011, pp. 266–273. arXiv: `1202.3725`.

[188] Jie Liu. "Adaptive myoelectric pattern recognition toward improved multifunctional prosthesis control". In: *Med. Eng. Phys.* 37.4 (Apr. 2015), pp. 424–430. ISSN: 13504533. DOI: `10.1016/j.medengphy.2015.02.005`. URL: `https://linkinghub.elsevier.com/retrieve/pii/S1350453315000399`.

[189] C.P. Diehl and Gert Cauwenberghs. "Svm incremental learning, adaptation and optimization". In: *Proc. Int. Jt. Conf. Neural Networks, 2003.* Vol. 4. IEEE, 2003, pp. 2685–2690. ISBN: 0-7803-7898-9. DOI: `10.1109/IJCNN.2003.1223991`. URL: `http://ieeexplore.ieee.org/document/1223991/`.

[190] Xiangyang Zhu et al. "Cascaded Adaptation Framework for Fast Calibration of Myoelectric Control". In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 25.3 (Mar. 2017), pp. 254–264. ISSN: 1534-4320. DOI: `10.1109/TNSRE.2016.2562180`. URL: `https://ieeexplore.ieee.org/document/7464360/`.

[191] Pentti Kanerva. “Binary spatter-coding of ordered K-tuples”. In: *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics).* Vol. 1112 LNCS. 1996, pp. 869–873. DOI: `10.1007/3-540-61510-5_146`. URL: `http://link.springer.com/10.1007/3-540-61510-5%7B%5C_%7D146`.

[192] Manuel Schmuck, Luca Benini, and Abbas Rahimi. “Hardware Optimizations of Dense Binary Hyperdimensional Computing: Rematerialization of Hypervectors, Binarized Bundling, and Combinational Associative Memory”. In: *ACM J. Emerg. Technol. Comput. Syst.* 15.4 (Dec. 2019), pp. 1–25. ISSN: 1550-4832. DOI: `10.1145/3314326`. URL: `https://dl.acm.org/doi/10.1145/3314326`.

[193] Sidharth Pancholi and Amit M. Joshi. “Electromyography-Based Hand Gesture Recognition System for Upper Limb Amputees”. In: *IEEE Sensors Lett.* 3.3 (Mar. 2019), pp. 1–4. ISSN: 2475-1472. DOI: `10.1109/LSENS.2019.2898257`. URL: `https://ieeexplore.ieee.org/document/8637799/`.

[194] Simone Benatti et al. “Online Learning and Classification of EMG-Based Gestures on a Parallel Ultra-Low Power Platform Using Hyperdimensional Computing”. In: *IEEE Trans. Biomed. Circuits Syst.* 13.3 (June 2019), pp. 516–528. ISSN: 1932-4545. DOI: `10.1109/TBCAS.2019.2914476`. URL: `https://ieeexplore.ieee.org/document/8704957/`.

[195] Dapeng Yang et al. “EMG pattern recognition and grasping force estimation: Improvement to the myocontrol of multi-DOF prosthetic hands”. In: *2009 IEEE/RSJ Int. Conf. Intell. Robot. Syst.* IEEE, Oct. 2009, pp. 516–521. ISBN: 978-1-4244-3803-7. DOI: `10.1109/IROS.2009.5354544`. URL: `http://ieeexplore.ieee.org/document/5354544/`.

[196] K. Englehart and B. Hudgins. “A robust, real-time control scheme for multifunction myoelectric control”. In: *IEEE Trans. Biomed. Eng.* 50.7 (July 2003), pp. 848–854. ISSN: 0018-9294. DOI: `10.1109/TBME.2003.813539`. URL: `http://ieeexplore.ieee.org/document/1206493/`.

[197] Simone Benatti et al. “Online Learning and Classification of EMG-Based Gestures on a Parallel Ultra-Low Power Platform Using Hyperdimensional Computing”. In: *IEEE Trans. Biomed. Circuits Syst.* (2019). ISSN: 19409990. DOI: `10.1109/TBCAS.2019.2914476`.

[198] Ning Jiang et al. “Myoelectric Control of Artificial Limbs—Is There a Need to Change Focus? [In the Spotlight]”. In: *IEEE Signal Process. Mag.* 29.5 (Sept. 2012), pp. 152–150. ISSN: 1053-5888. DOI: `10.1109/MSP.2012.2203480`. URL: `http://ieeexplore.ieee.org/document/6279589/`.

[199] Paul Kaufmann, Kevin Englehart, and Marco Platzner. “Fluctuating emg signals: Investigating long-term effects of pattern matching algorithms”. In: *2010 Annu. Int. Conf. IEEE Eng. Med. Biol.* IEEE, Aug. 2010, pp. 6357–6360. ISBN: 978-1-4244-

4123-5. DOI: 10.1109/IEMBS.2010.5627288. URL: http://ieeexplore.ieee.org/document/5627288/.

[200] Anders Fougner et al. “Resolving the limb position effect in myoelectric pattern recognition”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* (2011). ISSN: 15344320. DOI: 10.1109/TNSRE.2011.2163529.

[201] Bojan Milosevic, Elisabetta Farella, and Simone Benatti. “Exploring Arm Posture and Temporal Variability in Myoelectric Hand Gesture Recognition”. In: *2018 7th IEEE Int. Conf. Biomed. Robot. Biomechatronics.* IEEE, Aug. 2018, pp. 1032–1037. ISBN: 978-1-5386-8183-1. DOI: 10.1109/BIOROB.2018.8487838. URL: https://ieeexplore.ieee.org/document/8487838/.

[202] A. Radmand, E. Scheme, and K. Englehart. “A characterization of the effect of limb position on EMG features to guide the development of effective prosthetic control schemes”. In: *2014 36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.* IEEE, Aug. 2014, pp. 662–667. ISBN: 978-1-4244-7929-0. DOI: 10.1109/EMBC.2014.6943678. URL: http://ieeexplore.ieee.org/document/6943678/.

[203] Ali Moin et al. “Analysis of Contraction Effort Level in EMG-Based Gesture Recognition Using Hyperdimensional Computing”. In: *2019 IEEE Biomed. Circuits Syst. Conf.* IEEE, Oct. 2019, pp. 1–4. ISBN: 978-1-5090-0617-5. DOI: 10.1109/BIOCAS.2019.8919214. URL: https://ieeexplore.ieee.org/document/8919214/.

[204] Waseem Shahzad et al. “Enhanced Performance for Multi-Forearm Movement Decoding Using Hybrid IMU–sEMG Interface”. In: *Front. Neurorobot.* 13 (July 2019). ISSN: 1662-5218. DOI: 10.3389/fnbot.2019.00043. URL: https://www.frontiersin.org/article/10.3389/fnbot.2019.00043/full.

[205] Yanjuan Geng, Ping Zhou, and Guanglin Li. “Toward attenuating the impact of arm positions on electromyography pattern-recognition based motion classification in transradial amputees”. In: *J. Neuroeng. Rehabil.* 9.1 (2012), p. 74. ISSN: 1743-0003. DOI: 10.1186/1743-0003-9-74. URL: http://jneuroengrehab.biomedcentral.com/articles/10.1186/1743-0003-9-74.

[206] E. Scheme et al. “Examining the adverse effects of limb position on pattern recognition based myoelectric control”. In: *2010 Annu. Int. Conf. IEEE Eng. Med. Biol.* IEEE, Aug. 2010, pp. 6337–6340. ISBN: 978-1-4244-4123-5. DOI: 10.1109/IEMBS.2010.5627638. URL: http://ieeexplore.ieee.org/document/5627638/.

[207] Jonathon W. Sensinger, Blair A. Lock, and Todd A. Kuiken. “Adaptive Pattern Recognition of Myoelectric Signals: Exploration of Conceptual Framework and Practical Algorithms”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 17.3 (June 2009), pp. 270–278. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2009.2023282. URL: https://ieeexplore.ieee.org/document/5061575/.

[208] Dapeng Yang et al. “Adaptive learning of multi-finger motion recognition based on support vector machine”. In: *2013 IEEE Int. Conf. Robot. Biomimetics*. IEEE, Dec. 2013, pp. 2231–2238. ISBN: 978-1-4799-2744-9. DOI: 10.1109/ROBIO.2013.6739801. URL: http://ieeexplore.ieee.org/document/6739801/.

[209] Qi Huang et al. “A Novel Unsupervised Adaptive Learning Method for Long-Term Electromyography (EMG) Pattern Recognition”. In: *Sensors* 17.6 (June 2017), p. 1370. ISSN: 1424-8220. DOI: 10.3390/s17061370. URL: http://www.mdpi.com/1424-8220/17/6/1370.

[210] Marina M.C. Vidovic et al. “Improving the Robustness of Myoelectric Pattern Recognition for Upper Limb Prostheses by Covariate Shift Adaptation”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* 24.9 (Sept. 2016), pp. 961–970. ISSN: 1534-4320. DOI: 10.1109/TNSRE.2015.2492619. URL: https://ieeexplore.ieee.org/document/7302056/.

[211] Jianwei Liu et al. “Reduced Daily Recalibration of Myoelectric Prosthesis Classifiers Based on Domain Adaptation”. In: *IEEE J. Biomed. Heal. Informatics* 20.1 (Jan. 2016), pp. 166–176. ISSN: 2168-2194. DOI: 10.1109/JBHI.2014.2380454. URL: http://ieeexplore.ieee.org/document/6985518/.

[212] Xinpu Chen, Dingguo Zhang, and Xiangyang Zhu. “Application of a self-enhancing classification method to electromyography pattern recognition for multifunctional prosthesis control”. In: *J. Neuroeng. Rehabil.* 10.1 (2013), p. 44. ISSN: 1743-0003. DOI: 10.1186/1743-0003-10-44. URL: http://jneuroengrehab.biomedcentral.com/articles/10.1186/1743-0003-10-44.

[213] Yikun Gu et al. “Robust EMG pattern recognition in the presence of confounding factors: features, classifiers and adaptive learning”. In: *Expert Syst. Appl.* (2018). ISSN: 09574174. DOI: 10.1016/j.eswa.2017.11.049.

[214] C.P. Diehl and Gert Cauwenberghs. “Svm incremental learning, adaptation and optimization”. In: *Proc. Int. Jt. Conf. Neural Networks, 2003.* Vol. 4. IEEE, 2003, pp. 2685–2690. ISBN: 0-7803-7898-9. DOI: 10.1109/IJCNN.2003.1223991. URL: http://ieeexplore.ieee.org/document/1223991/.

[215] Antoine Bordes et al. “Fast kernel classifiers with online and active learning”. In: *J. Mach. Learn. Res.* 6 (2005). ISSN: 15337928.

[216] David Nova and Pablo A. Estévez. “A review of learning vector quantization classifiers”. In: *Neural Comput. Appl.* 25.3-4 (Sept. 2014), pp. 511–524. ISSN: 0941-0643. DOI: 10.1007/s00521-013-1535-3. arXiv: 1509.07093. URL: http://link.springer.com/10.1007/s00521-013-1535-3.

[217] Atsushi Sato and Keiji Yamada. “Generalized Learning Vector Quantization”. In: *Adv. Neural Inf. Process. Syst.* (1996), pp. 423–429.

[218] Mohsen Imani et al. “AdaptHD: Adaptive Efficient Training for Brain-Inspired Hyperdimensional Computing”. In: *2019 IEEE Biomed. Circuits Syst. Conf.* IEEE, Oct. 2019, pp. 1–4. ISBN: 978-1-5090-0617-5. DOI: 10.1109/BIOCAS.2019.8918974. URL: https://ieeexplore.ieee.org/document/8918974/.

[219] Mansour Kheffache. “Energy-Efficient Detection of Atrial Fibrillation in the Context of Resource-Restrained Devices”. PhD thesis. Lulea University of Technology, 2019.

[220] Brian Cheung et al. “Superposition of many models into one”. In: *Adv. Neural Inf. Process. Syst.* 2019.

[221] Cheng-Yang Chang, Yu-Chuan Chuang, and An-Yeu (Andy) Wu. “Task-Projected Hyperdimensional Computing for Multi-task Learning”. In: *IFIP Adv. Inf. Commun. Technol.* 2020, pp. 241–251. DOI: 10.1007/978-3-030-49161-1_21. URL: http://link.springer.com/10.1007/978-3-030-49161-1%7B%5C_%7D21.

[222] Darrell Whitley. “A genetic algorithm tutorial”. In: *Stat. Comput.* 4.2 (June 1994). ISSN: 0960-3174. DOI: 10.1007/BF00175354. URL: http://link.springer.com/10.1007/BF00175354.

[223] Alisha Menon et al. *Efficient emotion recognition using hyperdimensional computing with combinatorial channel encoding and cellular automata.* 2021. arXiv: 2104.02804 [cs.ET].

[224] Nadeem Ahmed Syed, Huan Liu, and Kah Kay Sung. “Handling concept drifts in incremental learning with support vector machines”. In: *Proc. fifth ACM SIGKDD Int. Conf. Knowl. Discov. data Min. - KDD ’99.* New York, New York, USA: ACM Press, 1999, pp. 317–321. ISBN: 1581131437. DOI: 10.1145/312129.312267. URL: http://portal.acm.org/citation.cfm?doid=312129.312267.

[225] Carlotta Domeniconi and Dimitrios Gunopulos. “Incremental support vector machine construction”. In: *Proc. 2001 IEEE Int. Conf. Data Min.* IEEE Comput. Soc, 2001, pp. 589–592. ISBN: 0-7695-1119-8. DOI: 10.1109/ICDM.2001.989572. URL: http://ieeexplore.ieee.org/document/989572/.

[226] Xinpu Chen, Dingguo Zhang, and Xiangyang Zhu. “Application of a self-enhancing classification method to electromyography pattern recognition for multifunctional prosthesis control”. In: *J. Neuroeng. Rehabil.* 10.1 (2013), p. 44. ISSN: 1743-0003. DOI: 10.1186/1743-0003-10-44. URL: http://jneuroengrehab.biomedcentral.com/articles/10.1186/1743-0003-10-44.

[227] Arslan Chaudhry et al. “Riemannian Walk for Incremental Learning: Understanding Forgetting and Intransigence”. In: *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics).* Vol. 11215 LNCS. 2018, pp. 556–572. DOI: 10.1007/978-3-030-01252-6_33. URL: http://link.springer.com/10.1007/978-3-030-01252-6%7B%5C_%7D33.

[228] He Huang et al. “Continuous locomotion-mode identification for prosthetic legs based on neuromuscular - Mechanical fusion”. In: *IEEE Trans. Biomed. Eng.* (2011). ISSN: 00189294. DOI: 10.1109/TBME.2011.2161671.

[229] Lin Du et al. “Toward Design of an Environment-Aware Adaptive Locomotion-Mode-Recognition System”. In: *IEEE Trans. Biomed. Eng.* 59.10 (Oct. 2012), pp. 2716–2725. ISSN: 0018-9294. DOI: 10.1109/TBME.2012.2208641. URL: http://ieeexplore.ieee.org/document/6239577/.

[230] Elaine A. Corbett, Konrad P. Kording, and Eric J. Perreault. “Real-time evaluation of a noninvasive neuroprosthetic interface for control of reach”. In: *IEEE Trans. Neural Syst. Rehabil. Eng.* (2013). ISSN: 15344320. DOI: 10.1109/TNSRE.2013.2251664.

[231] A. Searle and L. Kirkup. “A direct comparison of wet, dry and insulating bioelectric recording electrodes”. In: *Physiol. Meas.* 21.2 (May 2000), pp. 271–283. ISSN: 0967-3334. DOI: 10.1088/0967-3334/21/2/307. URL: https://iopscience.iop.org/article/10.1088/0967-3334/21/2/307.

[232] Osamu Fukuda et al. “A human-assisting manipulator teleoperated by EMG signals and arm motions”. In: *IEEE Trans. Robot. Autom.* 19.2 (Apr. 2003), pp. 210–222. ISSN: 1042-296X. DOI: 10.1109/TRA.2003.808873. URL: http://ieeexplore.ieee.org/document/1192150/.

[233] Adelson Chua, Michael I. Jordan, and Rikky Muller. “A 1.5nJ/cls Unsupervised On-line Learning Classifier for Seizure Detection”. In: *2021 Symp. VLSI Circuits*. IEEE, June 2021, pp. 1–2. ISBN: 978-4-86348-780-2. DOI: 10.23919/VLSICircuits52068.2021.9492392. URL: https://ieeexplore.ieee.org/document/9492392/.

[234] Alessio Burrello et al. “Hyperdimensional Computing With Local Binary Patterns: One-Shot Learning of Seizure Onset and Identification of Ictogenic Brain Regions Using Short-Time iEEG Recordings”. In: *IEEE Trans. Biomed. Eng.* 67.2 (Feb. 2020), pp. 601–613. ISSN: 0018-9294. DOI: 10.1109/TBME.2019.2919137. URL: https://ieeexplore.ieee.org/document/8723166/.