EMBody: A Data-Centric Toolkit for EMG-Based Interface Prototyping and Experimentation

JAKOB KAROLUS, LMU Munich, Germany
FRANCISCO KISS, University of Stuttgart, Germany
CAROLINE ECKERTH, LMU Munich, Germany
NICOLAS VIOT, LII ENAC, France
FELIX BACHMANN, LMU Munich, Germany
ALBRECHT SCHMIDT, LMU Munich, Germany
PAWEŁ W. WOŹNIAK, Utrecht University, the Netherlands

Body movements, from a short smile to a marathon run, are driven by muscle activity. Despite the fact that measuring muscle activity with electromyography (EMG) is technically well established, it is highly complex and its use in interfaces has been limited. Easy access to muscle sensing can offer new opportunities to Human-Computer Interaction (HCI) research. Off-the-shelf sensors often only provide low-level access, hence requiring expertise in signal processing and widening the gulf of execution for users without engineering skills. To address this challenge, we introduce EMBody, a data-centric toolkit for EMG-based interface prototyping and experimentation. EMBody offers multiple levels of prototyping fidelity for EMG sensing, signal processing, and data interpretation. Our data-centric toolkit encapsulates the different data representation stages, offering a wide range of customization opportunities to experts while also allowing non-technical designers to focus on creating new interaction techniques. EMBody features a lightweight form factor and wireless connectivity. Additionally, the system leverages an exploration-centered workflow by allowing rapid access to measurement data via the accompanying software. Users define a set of motions to be recognized and interactively provide example data points. The toolkit then handles signal processing and classification. The recognized movements are streamed on the local network, ready to be used by interactive applications. This paper reports on how to use EMBody and its implementation. We iteratively developed the toolkit in a series of workshops and example applications. Users who had none or very limited knowledge of EMG could rapidly create engaging functional prototypes, while experts appreciated the modularity of the software component allowing for a high degree of customization. We contribute the software and hardware components of EMBody as a tool for the research community to stimulate creative exploration of EMG systems.

CCS Concepts: • **Human-centered computing → Ubiquitous and mobile computing systems and tools**; *Interaction paradigms.*

Additional Key Words and Phrases: Electromyography; embodied interaction; toolkit; prototyping.

Authors' addresses: Jakob Karolus, jakob.karolus@ifi.lmu.de, LMU Munich, Germany; Francisco Kiss, franciscokiss@gmail.com, University of Stuttgart, Germany; Caroline Eckerth, caroline.eckerth@gmx.de, LMU Munich, Germany; Nicolas Viot, nicolasviot2@gmail.com, LII ENAC, France; Felix Bachmann, bachmann@bchmn.de, LMU Munich, Germany; Albrecht Schmidt, albrecht.schmidt@ifi.lmu.de, LMU Munich, Germany; Paweł W. Woźniak, p.w.wozniak@uu.nl, Utrecht University, the Netherlands.

2573-0142/2021/6-ART195
https://doi.org/10.1145/3457142

ACM Reference Format:
Jakob Karolus, Francisco Kiss, Caroline Eckerth, Nicolas Viot, Felix Bachmann, Albrecht Schmidt, and Paweł W. Woźniak. 2021. EMBody: A Data-Centric Toolkit for EMG-Based Interface Prototyping and Experimentation. *Proc. ACM Hum.-Comput. Interact.* 5, EICS, Article 195 (June 2021), 29 pages. https://doi.org/10.1145/3457142

1 INTRODUCTION

Movement is becoming an increasingly important part of our interactions with computers. At the end of the previous century, Paul Dourish envisioned how physically and socially enacted encounters with technology will transform our everyday lives [5]. Later, Dag Svanæs [48] proposed expanding the concept of embodiment to, inter alia, embedded perception, i.e., extending one's senses and awareness through technology. Embodied interaction is now profoundly present in commercial and research systems from gestural interaction in the kitchen [31] to increased perception through vibration [2]. Embodied interaction involves movement—a process where users implicitly contract and relax their muscles to move their bodies in particular patterns. In this paper, we investigate the means of easily allowing HCI researchers to understand our muscles in more detail in order to design for embodied interaction.

For this purpose, we use electromyography (EMG) which is a technique to record muscular activity by measuring the electrical field generated by the contractions of muscle fibers through electrodes on the skin [27]. Seminal work by Saponas et al. [37] showcased the feasibility of recognizing gestures by employing EMG around the forearm, coining the term "muscle-computer interfaces". They proposed an interaction methodology that relied directly on recognizing muscular activity rather than physical actuation. Apart from explicit interaction using EMG-based interfaces, EMG excels at providing in-depth insight into a person's muscle activation and thus human body movements [19, 51] aiding in human activity recognition.

However, building EMG systems from scratch is difficult and requires expertise from numerous domains such as sensing technology, signal processing, machine learning and interaction design. Here, off-the-shelf electrical sensors[1] can only provide the first step of this process. Users need to take care of data processing, calibration and model training. Hence, a significant amount of time needs to be invested, adversely affecting work on designing for EMG-Based interfaces. This constitutes an obstacle for a broader application of EMG-based interaction. Consequently, there is a need for new EMG tools for HCI prototyping. An EMG toolkit for HCI should offer opportunities for interaction designers, researchers, and engineers to engage with EMG-based sensing on different levels of technical complexity.

This paper introduces our data-centric toolkit for EMG—EMBody (Figure 1), which we contribute to the research community as an open-source tool. The toolkit aims to enable the design and implementation of interactive artifacts, exploring the different interaction possibilities offered by EMG and offers extensive customization support for EMG experts when conducting experiments. Consequently, EMBody was designed to *reduce authoring time and complexity*, *empower new audiences* and *enable replication and creative exploration*, cf. [22]. Practitioners are able to iterate designs that use EMG without the need for extensive preparation, allowing for quick comparison of alternative designs. Furthermore, we explicitly tailored EMBody for the needs of not only designers, but also developers and engineers. EMBody offers modular code structured in an exploration-centered data processing workflow (Figure 2). This ensures fine-grained control for experts when needed, while also providing an accessible point of entry for non-experts into EMG-based interface prototyping.

[1]Sensors that measure electrical potential between electrodes as used by EMG.

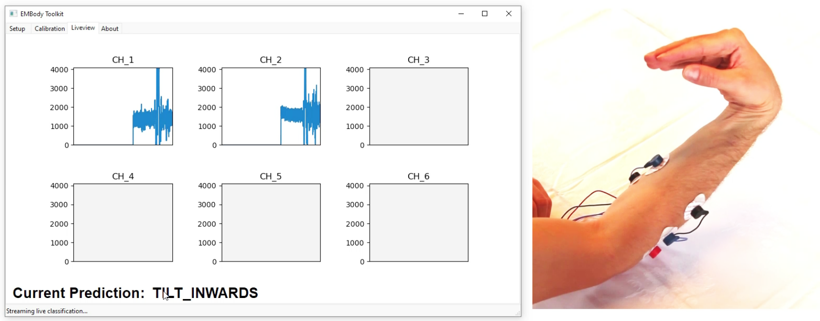

Fig. 1. Live classification stream in EMBody's software application (left). Using two EMG channels, the software predicts the current gesture (right) and provides it via a network stream for other applications to use.

We evaluated EMBody by conducting two workshops, where we showcased the capabilities of the toolkit and refined its design. We also present two example applications which serve to evaluate the feasibility of the toolkit and demonstrate its suitability for rapid prototyping of physiological user interfaces. Two use case scenarios highlight target audiences and respective workflows when working with EMBody. We further established the viability of the final toolkit through expert interviews, identifying requirements and challenges for EMG input prototyping.

This paper contributes the following: (1) the design and implementation of EMBody—a data-centric toolkit for rapid EMG-based interface prototyping and experimentation, which consists of a modular, open-source software application for interpreting electromyograms and accompanying open-source hardware; (2) the specification of a modular data processing pipeline for EMG that offers customization according to the requirements of a particular prototype; (3) a formative evaluation of the toolkit in two workshops in which teams of participants successfully built prototypes of EMG-based applications; (4) a summative evaluation of the toolkit in the form of expert interviews and (5) two example applications that illustrate how EMBody can be effectively used to build research prototypes for EMG-based interactions.

2 RELATED WORK

One major driver of embodied interaction is the increasing availability of toolkits allowing users to rapidly prototype interaction ideas. Most commonly, toolkits help ease certain steps during this process [22], from ideation and interaction design to signal acquisition and processing to higher level output generation, e.g. by means of machine learning. In the following, we reflect on prominent toolkits within the HCI domain, their purposes and architecture and take a closer look at physiological computing toolkits and electromyographic sensing.

2.1 Toolkits

Toolkits lower the entrance barrier for specific stages during the creation process of applications and artifacts [22]. Specialized toolkits, such as Makers' Marks [39], Sauron [38], Pineal [21], ShapeMe [53] and RetroFab [35] support technical users in working with aesthetics and form factors. The aforementioned tools allow novice users to create 3D forms or enclosures, which is a process which usually involves extensive knowledge and iterations using 3D software. By abstracting from this process, e.g. through shortcuts like embedding smartwatches as computation unit [21] or automated processes that convert physical changes to digital representations [53], these toolkits enable users to focus on designing applications and benefit from advances in sensor technology and 3D modeling. Similarly, EMBody circumvents the need for novice users to be knowledgeable in signal and EMG processing and leverages the expressive power of muscle activity in their applications.

Other toolkits specifically address these engineering challenges, such as making sense of data (EagleSense [54], SoD-Toolkit [44]) assisting less technically adept users to interpret sensor data and high-level input for their applications. Taking care of data synchronization and filtering is cumbersome and often requires expertise, especially as environments contain more and more sensors every day. A designer for a location-aware application is only interested in a person's exact location and orientation (within a room). How this information is calculated is secondary and not relevant for the application. Here, toolkits such as EagleSense [54] abstract from the technical complexity allowing fast prototyping. EMBody offers similar features. It provides an abstraction layer for novice users from low-level implementations of sensing and interpreting EMG data. If desired, users of EMBody can prototype EMG-based applications without the need to ever know anything about EMG but that it measures muscle activity.

Developing ubiquitous artifacts often involves devices that are interconnected, e.g. a sensor and an actuator. Cross-device communication can be cumbersome and is often abstracted with the help of protocols and toolkits. Examples include toolkits for web-based applications (XDStudio [29], Panelrama [55]), tangible artifacts (Calder [23], ToyVision [24], reacTIVision [17]) or most commonly: wearable devices (WatchConnect [15], Weave [3], Interactex [11], WDK [10]). These toolkits showcase the importance of cross-device compatibility and properly defined interfaces to allow for robust communication among the devices. Hence, design and implementation decisions for EMBody are informed by these works by providing a clear interface between its hardware and software components (see Section 3.3 and Section 3.4) as well as the user's application. The software component connects to the UDP stream of the EMG hardware (or any off-the-shelf electrical potential sensor on the market, see Section 2.4 for examples), processes and interprets the EMG data and provides a high-level gesture stream via UDP for the user's application.

2.2 Pipeline Architectures

A common approach to toolkit architecture is the use of a pipeline-based structure. This provides a conceptual workflow for diverse user groups and clearly communicates the toolkit's application domain. Toolkits might only provide one — but integral — step of this pipeline, such as enabling laymen to work with electric muscle stimulation [33], paper electronics [36], fostering data engagement [14] or simply enabling rapid prototyping of electronics [52].

This concept of separating individual steps of the creation process allows for a data-centric view during development. Steps can be parallelized to increase efficiency and enable outsourcing of complex processing to domain experts. Not being endemic to the prototyping and toolkit domain, the concept of pipelining can be found in other areas as well, such as visualizations [7], fabrication [43], debugging [46] and media [45]. EMBody employs this concept and defines an exploration-centered workflow for novice and amateur users when designing for EMG-based interaction. Its modular data processing pipeline allows the expert user to customize EMBody to a high degree while maintaining a low entry threshold for novices.

2.3 Physical Computing Toolkits

User interfaces which use physiological phenomena as input are an established topic in HCI. Given the amount of measurable biological signals [42] which we emit, be it electrical[2], impedance-based[3] or acoustic[4], there is still potential for new interfaces. Yet, the possibilities for non-experts to explore user input modalities for interactive systems are limited. Most often, expensive equipment and prior training is required to operate systems correctly. Hence, accessing physiological input technologies is often cumbersome and requires expert knowledge. Universally available hardware and toolkits are a first step in this direction, as they allow non-experts to familiarize themselves with the technology and explore its capabilities without having to commit to expensive hardware and training.

Past work contributed tools for interaction designers who were not familiar with the employed sensing technologies. These toolkits [9, 16, 40, 47, 49] enable designers to realize their ideas by abstracting underlying sensor complexities and reducing the need for extensive expertise, which is often the most significant barrier to entry in using physiological computing [6]. In line, EMBody abstracts from underlying sensor complexities of EMG-based interaction.

Other toolkits [8, 10, 25, 26, 50] focus on a more developer-centered approach, e.g. by providing access to low-level hardware using high-level programming languages. A key aspect that unites all these past efforts is the focus on an iterative, design-centered approach [12] to create interactive systems. EMBody continues this philosophy and allows for rapid prototyping of EMG-based interfaces, by providing abstraction were required but allowing for customization were needed.

2.4 Electromyographic Sensing

Related work has identified the necessary requirements for EMG recordings, such as the physical hardware, appropriate sampling rates and recording modes [27, 28, 37]. Suitable electrode arrangements were discussed for various usage scenarios within HCI, such as finger gestures for input [37], intimate interfaces [4], guitar tutoring [19] or augmented piano playing [18]. However, most recent work relies on expensive hardware, often diminishing the real-world applicability of these systems. Thus, there is a need for a lightweight apparatus that would allow for easier experimentation with EMG-based interfaces.

Low-cost commercial EMG products include shields[5] and sensors (MyoWare[6], BioVolt[7]) for microcontrollers, e.g. Arduino. These devices only provide a low-level API, widening the gulf of execution for users without engineering skills. A more accessible product, which enabled designers to design for EMG gestures, was the Myo armband[8], but it has been discontinued. EMBody aims to address this gap by offering rapid access to EMG data with its dedicated software, taking care of data processing and interpretation. Hence, users can rely on a robust high-level gesture stream for their applications.

3 THE EMBODY TOOLKIT

As a consequence, the need for an EMG toolkit that would enable easy exploration emerges. We began our work with an initial set of requirements motivated by Ledo et al.'s [22] definition of a toolkit:

[2]muscle and brain activity
[3]skin resistance
[4]respiration and heart flow
[5]https://www.olimex.com/Products/Duino/Shields/SHIELD-EKG-EMG/open-source-hardware
[6]http://www.advancertechnologies.com/p/myoware.html
[7]https://infusionsystems.com/catalog/product_info.php/products_id/198
[8]https://support.getmyo.com/hc/en-us

"[Toolkits are] generative platforms designed to create new interactive artifacts, provide easy access to complex algorithms, enable fast prototyping of software and hardware interfaces, and/or enable creative exploration of design spaces." [22]

Especially in the domain of physiological computing, reducing the need for extensive expertise is essential. Fairclough [6] called for easy data acquisition and abstraction from technical details. Similarly, seminal work by Schilit et al. [41] on context-aware computing highlighted the importance of a person's environment for interactive applications. As a consequence, such toolkits need to be mobile and effectively support the creative process. Last but not least, toolkits are to "empower new audience" [22]. In this work, we explicitly focus on including a wide range of professions as potential user groups.

Consequently, we strongly base our design goals for EMBody on the five goals aggregated by Ledo et al. [22]. We further take related work from past toolkit research and especially physiological sensing into account (Section 2). Based on those past works, we derive four requirements for EMG-based prototyping and experimentation. In this work, we iteratively refined, addressed and evaluated the requirements in two workshops (Section 4.1), through developing sample applications (Section 4.2) and conducting expert interviews (Section 5).

The following section describes the final set of requirements for EMG-based prototyping and experimentation. We detail the corresponding exploration-centered workflow of EMBody as well its software and hardware components, highlighting possible extension points for technically skilled users. All resources needed to build, use and modify EMBody are open-source and are available on github[9]. Finally, a closer look at the different workflows possible with EMBody is provided through two use case scenarios in Section 3.6.

3.1 Requirements for EMG-Based Prototyping and Experimentation

Throughout the development process of EMBody, we identified four main requirements for EMG-based prototyping and experimentation. We specifically address the needs of a wider range of professions, who could potentially use EMG, for our toolkit.

Mobility. Mobility is a key aspect to facilitate prototyping and exploration not only in constrained lab environment, but to allow for in-the-field exploration. EMBody offers a low-power and lightweight apparatus which is mobile and can be carried by the user. This property enables straightforward in-situ exploration of interaction scenarios ensuring high external validity. The need for highly mobile EMG-based interfaces was exemplified by past work in HCI, which advocated using EMG for interactions on the go [4].

Data Acquisition. Data Acquisition often produces technical difficulties. Off-the-shelf sensor products rarely provide an abstraction layer for this process. Most commonly data has to be directly read via the analog port of a microcontroller. To facilitate rapid prototyping, data acquisition needs to be reliable and possible from a variety of devices. In the final version of EMBody, data acquisition is moderated using the UDP protocol over an existing WiFi connection. This allows experts to use custom hardware to communicate with EMBody's software application, by adhering to the protocol (Section 3.3). The standard hardware already provides a walk-through for users to connect to existing WiFi connections for initial setup.

Additionally, EMBody's hardware provides up to six sensing channels without further modifications. Off-the-shelf products often need to be extended, requiring additional electronics development[10]. For EMBody, selecting individual channels is handled by the software application.

[9] https://github.com/HCUM/embody.

[10] E.g. stacking multiple shields for the Olimex board or connecting multiple MyoWare sensors.

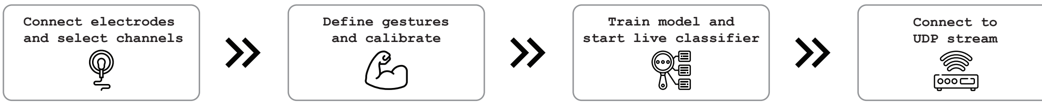

Fig. 2. Workflow when designing with EMBody.

Abstraction Layer. To tailor for novices in EMG sensing, the toolkit needs to protect this user group from unnecessary technical details including signal processing and data interpretation. For this purpose, EMBody realizes an exploration-centered workflow (Section 3.2) that guides the user through a total of four steps from an initial idea to a final working prototype. Apart from deciding on the electrode configuration (location and channels), which is supported through a manual[11], the user does not need to have any expertise in signal processing and data interpretation. EMBody features a predefined data processing pipeline including a set of filters and algorithms that take care of data interpretation, allowing the user to focus on exploration and interaction design.

Modular Structure. Conversely, experts want to have a fine level of control over the data processing pipeline. Consequently, EMBody offers gradual levels of fidelity through its modular structure. Every part of the workflow (Section 3.2) can be customized and adjusted to the user's needs. For an elaborate experiment, researchers might want to adjust the data pipeline by interchanging the classification algorithm or calculating different features. The accompanying software offers convenient extension points for this purpose, exposing various stages throughout the processing pipeline.

3.2 An Exploration-Centered Workflow

EMBody uses an exploration-centered workflow (Figure 2) guiding users from a first idea to a final prototype. This allows users to readily start exploring suitable interaction scenarios without the need for further configuration. The data-centric pipeline provides different views of the same EMG data, such as raw and filtered data, as well as generated features and final predictions. The following section introduces this workflow in detail while highlighting extension points for expert users.

3.2.1 Connect Electrodes and Select Channels. The standard firmware on EMBody's hardware offers a captive portal when connecting to its Wi-Fi network. The portal allows users to configure their preferred connection settings. Once configured, the prototype readily sends recorded data via the network. The system can be powered by any portable power source, such as a small powerbank, allowing continuous operation for multiple days.[12] Data is transmitted wirelessly without the need for additional cables apart from the electrode connections. Setting up the prototype and placing electrodes (Figure 3) is described in the enclosed manual[11].

Our toolkit allows recording up to six channels with a standard sampling rate of 250 Hz and provides sensing and recognition data via a UDP stream broadcast over the network. This enables users to track several muscles (groups) at once and recognize complex motor tasks. For technically skilled users, the firmware can be adjusted to their needs, offering a much higher sampling rate. Network capabilities are the bounding factor. Additionally, the software implements sanity checks on the received data, such as estimating sampling rate and tracking package loss during critical operations and informs the user about possible ways to solve these issues.

The EMBody live view (Figure 4) allows simultaneous tracking of up to six channels and helps the user identify faulty connections. This also allows refining electrode placement when signal

[11] Available at https://github.com/HCUM/embody/tree/master/manual.
[12] See Section 3.3 for performance details.

Fig. 3. EMBody works with a variety of electrodes. Here, they are affixed with straps to the forearm. First, the ground electrode is placed (left) on a location with little muscle fiber, e.g. close to the elbow. Afterwards, the sensing electrodes are placed on the muscle belly (right).

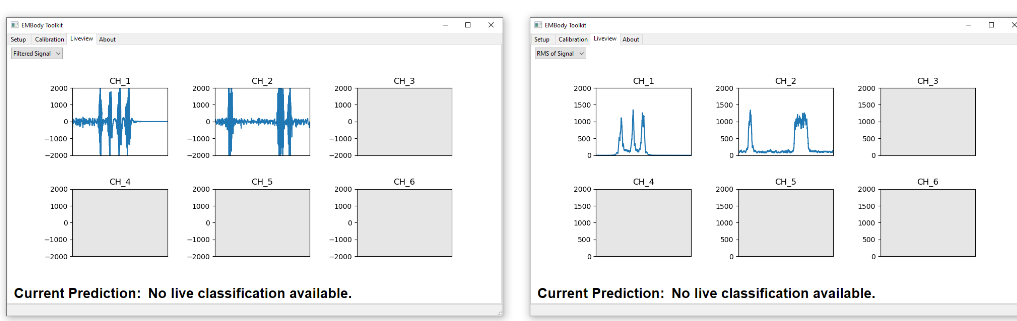

Fig. 4. Checking the EMG signal in the live view using different views of the EMG data. Filtered signal on the left; generated RMS features on the right. See Section 3.4 for algorithm details.

quality is low. The manual provides a set of guidelines on how to place electrodes to minimize noise. This first step helps users familiarize themselves with the EMG signal and discover how it reacts to their movements.

3.2.2 Define Gestures and Calibrate. Once electrode placement is completed, users provide a set of movements that they wish to recognize. EMBody will guide the user through this calibration process by instructing the user to perform the respective movements while collecting sample data for each movement (Figure 5). Additionally, EMBody verifies that the sampling rate is sufficient for further filtering steps and monitors potential package loss. If irregularities are detected, the user is advised to repeat the calibration, check for connection issues or redo the electrode setup.

During calibration, EMBody synchronizes the specified movements (the calibration labels) and incoming EMG data samples. EMBody collects more data samples for the *NULL_CLASS*, allowing the user to present motions that should not be recognized. This increases the robustness of the classifier. After completion, the recorded data is filtered and saved. An updated overview over all collected calibration data (duration per label) is displayed. For post-hoc analysis, EMBody offers an export function.

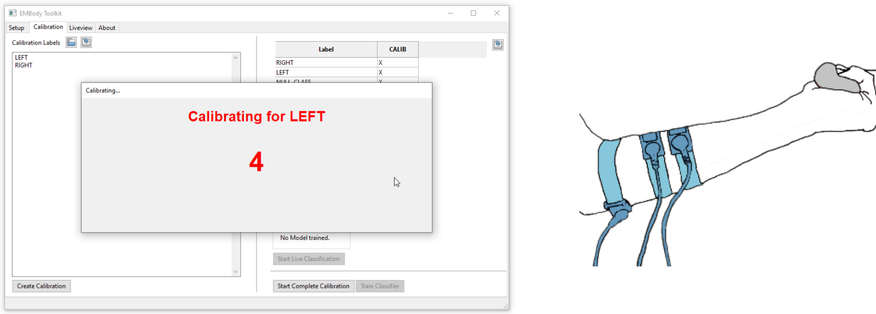

Fig. 5. EMBody calibrating for a gesture labeled *LEFT*. Corresponding gesture by the user (left arm) on the right.

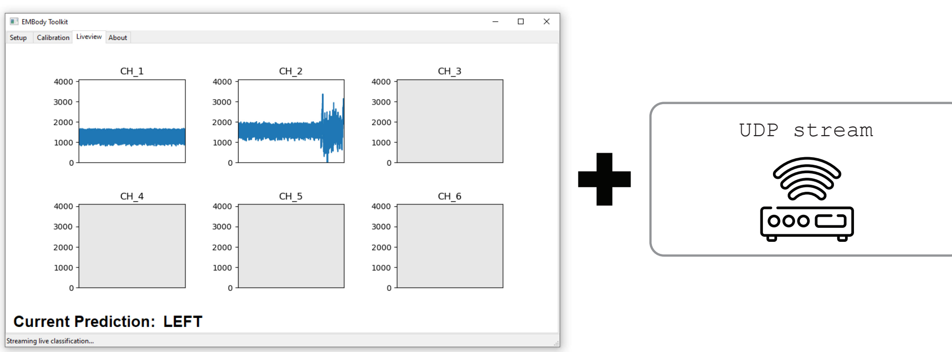

Fig. 6. Live view with live classification. Note that the classification is also streamed via UDP.

3.2.3 Train Model and Start Live Classifier. After completing a full calibration, the user is able to train a classification model using the provided discriminative method using a support vector machine (SVM). Internally, EMBody calculates Root-mean-square-based (RMS) features after filtering the data ([19, 37]). Depending on the selected amount of channels, pair-wise ratios between channels are calculated. This approach provides an indication of relative locality for the classifier. After generating the appropriate features, the software trains an SVM and evaluates the model using 10-fold cross validation on the calibration data.

While the significance of this metric is limited to the recorded calibration data, it supports the user in assessing whether the calibration movements are sufficiently distinct with respect to their recorded muscular activity. Low values[13] indicate that the chosen gestures are too similar. Thus, the user is advised to modify their gestures (backtrack in the workflow) or to provide additional sensing channels which may help distinguish the gestures (restart with electrode setup). This way, impractical electrode configurations and gesture sets can be identified quickly.

The standard feature generation and classification method of EMBody works well for short explicit EMG-based input. To recognize longer movements[14], other machine learning approaches, e.g. regression or correlation-based methods might be better suited. The modular structure of EMBody allows users to substitute and extend the classification module, providing their own training and

[13]Values lower than 80% might already be impractical for some applications.
[14]More than several seconds.

prediction routines. If the user decides to use the method as implemented, no knowledge about classifying EMG data is needed to use EMBody.

After starting live classification, EMBody switches to the live view (Figure 6) and continuously processes the incoming EMG data, generates respective features and provides a new prediction[15] every 80 *ms*.

3.2.4 Connect to UDP Stream. Live classification is displayed within the software and additionally provided as a network stream (UDP or LSL[16]). Whenever a new prediction is available, EMBody broadcasts the appropriate gesture label onto the network, which can then be accessed by any other application in the network.

3.3 Hardware

Our toolkit includes a versatile hardware system, capable of measuring up to six EMG signals in parallel and delivering them wirelessly over WiFi (Figure 7). The device senses the muscular activity primarily using dry electrodes, but is capable of working with other electrode types. Compatible electrodes must facilitate a bipolar measurement technique: apart from one reference electrode, two sensing electrodes are used to minimize the impact of noise artifacts. These electrical signals are individually processed by an analog instrumental amplifier and quantified by an Analog-Digital Converter (ADC). Finally, a microcontroller packages the data into UDP packets and transmits them using a WiFi antenna.

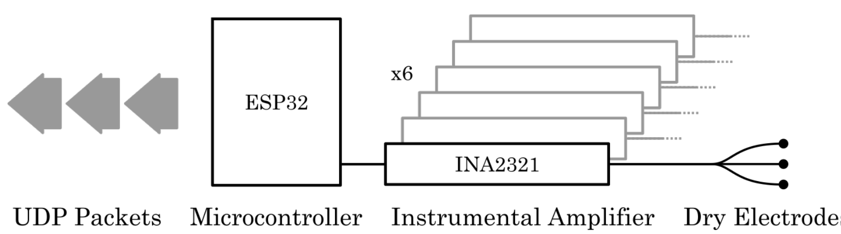

Fig. 7. Diagram of the EMG system: The system senses muscular activity with electrodes. These signals are amplified and sent in UDP packets over WiFi.

We based our design on existing circuits, aiming for compactness, wearability, and flexibility, while ensuring low noise levels and adequate data output. The instrumental amplifiers are an adaptation of an existing design[17], based on the INA2321[18], a low-power and low-cost CMOS amplifier. We used a board based on the ESP32 microcontroller, which is a low-cost and low-power *system on a chip* with integrated WiFi. Given the pin layout and the usage of WiFi, the ESP32 offers a total of six remaining ADC channels with a 12 bit resolution each, thus converting the output signals of the instrumental amplifiers to integer values from zero to 4095. Power consumption for the ESP32 with active radio transmission is approximately 240 *mA*[19], given maximum signal strength. Using a 5200 *mAh* powerbank yields up to 22 *h* of continuous operation (the power consumption of the amplifier is negligible).

During operation, the microcontroller polls all six channels and packs the measurements into frames of six values including a timestamp and broadcasts them to the connected network[20]. For

[15] Based on the provided gesture set.
[16] Lab streaming layer: https://github.com/sccn/labstreaminglayer
[17] https://github.com/BigCorvus/2-Channel-Biopotential-Amp
[18] http://www.ti.com/product/INA2321
[19] https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf
[20] Communication protocol: "timestamp;CH_x;CH_x;..."

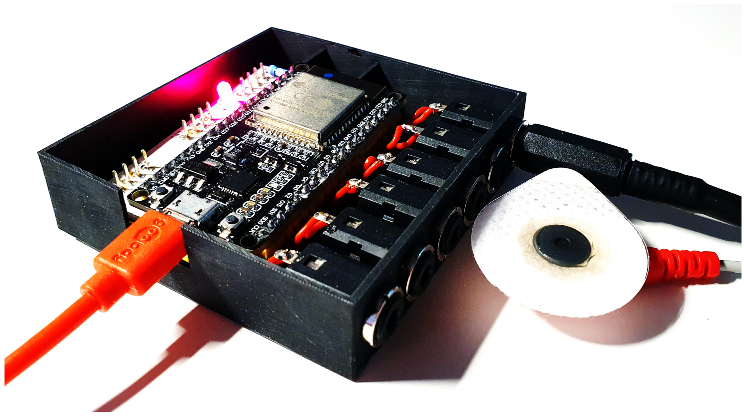

Fig. 8. EMBody's hardware prototype showing the six audio jacks on the front to connect electrodes and the microcontroller in the middle. The three amplifiers (each supporting two channels) are placed underneath the microcontroller. A lid (removed in this picture) is also provided.

EMG purposes, a sampling rate of 250 *Hz* is sufficient in most cases[21]. We provide all firmware files, allowing users to customize it to their needs.

To ensure portability and versatility, the system is mounted in a 3D-printed case with a wallet-like form factor and powered via the ESP32's USB port. Both powering the device and connecting it to a computer can be done with a micro-USB cable. Further, the inputs of the instrumental amplifiers are connected to 3.5 *mm* stereo audio jacks. This simplifies the management of the electrode cables and allows to adapt their number to specific requirements. Figure 8 depicts the hardware inside the case.

While we gladly provide the hardware upon request, the complete system can be built from our custom circuit design schematics and layouts which are available via the repository. Additionally, we provide assembly instructions, including a parts list. The available firmware and 3D design files complement the hardware resources.

3.4 Software

The EMBody software is a PC application developed in Python on Windows. It does not use native libraries, allowing it to be run on macOS and Linux as well. It follows a modular structure governed by EMBody's workflow. While it has been developed to be used with EMBody's hardware prototype in mind, the software can be used with any kind of sensor that uses the communication protocol[20]. New research probes such as PhysioSkin [30] and PolySense [13] are promising alternatives that are potentially compatible with our system. The complete source code is open-source and can be readily extended, both in terms of additional GUI elements (making use of EMBody's stream handling) and logic components (extending the data processing pipeline). The following section describes key components and highlights possibilities for extensions.

3.4.1 Data Processing Pipeline. Data filtering, processing and feature generation is encapsulated in EMBody's `ClassificationManager`. The following methods are of particular interest. Figure 9 shows the call hierarchy and information flow during the calibration and live classification phases.

[21]Proposed filters by related work [19, 37] make higher sampling rates unnecessary when using the standard pipeline of EMBody (see Section 3.4 for details).

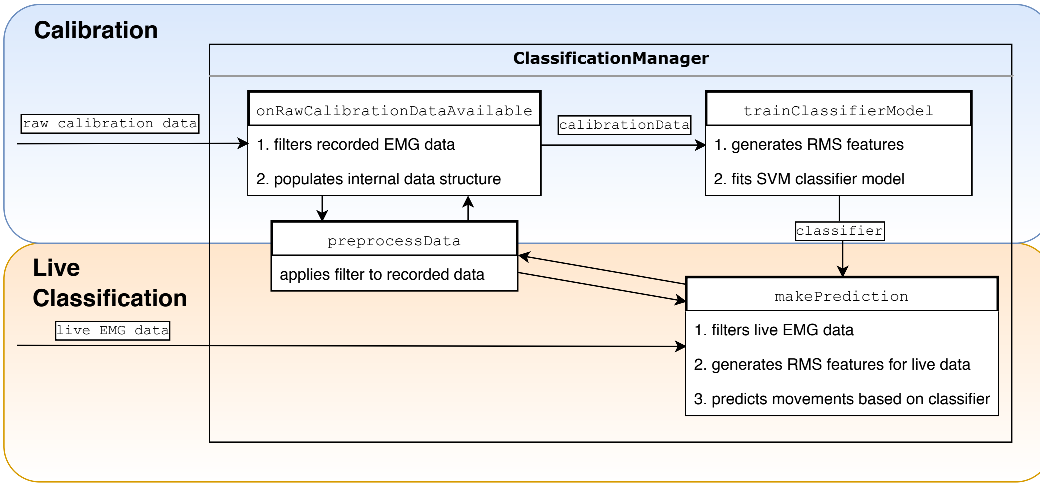

Fig. 9. Classification pipeline within `ClassificationManager`. Note the different flows for the calibration and live classification phases.

`onRawCalibrationDataAvailable`. After EMBody finishes a calibration run, it calls the method `onRawCalibrationDataAvailable` and passes all recorded samples and associated labels (see the source code for details). This method implements preprocessing steps (using `preprocessData`) and populates the internal data structure.

`preprocessData`. Closely following related work [19, 27, 37], this method applies a bandpass filter between 2 Hz and 100 Hz, attenuating long-term drifts, the DC offset and high-frequency noise as well as a bandstop filter between 49 Hz and 51 Hz in order to remove power line interference. The method returns a dataframe linking data samples to their respective class, i.e. the calibration labels. Additionally, data is grouped per calibration run[22].

`trainClassifierModel`. Implementing standard [19, 37] EMG features, this method provides epoched RMS features and their pair-wise ratios between channels. These values can be interpreted as a proxy for the intensity of muscle activity as the amplitude of the EMG signal increases when the muscular activity increases [27]. RMS is calculated using a convolutional approach and defined as

$$x_{RMS} = \sqrt{\frac{1}{n}(x_1^2 + x_2^2 + \ldots + x_n^2)}.$$

One important parameter for calculating RMS-based features is the window size n. It represents a trade-off between classification accuracy and latency, i.e. the time between acquiring EMG data and its prediction. Small windows allow for little latency, but are problematic when recognizing longer-lasting movements. Preliminary experiments confirmed that setting the window size to $n = 20$ yielded a good trade-off. Given a standard sampling rate of 250 Hz, this corresponds to a classification latency of 80 ms. These are the standard values in EMBody which can be modified when required.

[22]The *NULL_CASS* may be recorded multiple times during calibration. Grouping ensures that those samples are processed separately.

EMBody implements a support vector classification with a radial basis function kernel[23] using scikit-learn [24]. Features are scaled to unit variance and zero mean before fitting a model. A subsequent 10-fold cross validation provides a first indication of the model accuracy.

Overwriting the `trainClassifierModel` method allows the user to specify their own feature generation pipeline (cf. [34] for an overview). They can provide custom classification methods. When using sckit-learn, the user may still make use of the live prediction methods. Implementing other libraries, e.g. a correlation analysis, requires the user to also adapt *makePrediction*.

`makePrediction`. During live classification, the previously trained model is used to predict calibration labels for incoming data. Whenever a new prediction is requested[25], this method performs the preprocessing and feature generation steps and provides the respective predictions. A voting (mode-based) ensures a robust prediction, hence yielding one prediction per call. Alternatively, a list of predictions is also provided. Users may choose to work on the raw prediction data directly, or accumulate incoming data, e.g. for activity recognition, by overwriting this method. Listing 1 provides an excerpt highlighting key steps and possible extensions points.

```
def makePrediction(self, data):
  ...
  #applying filtering steps (Data is in df), change here for individual filters
  apply_bandpass_filter(df, 2.0, self.currentSamplingRate / 2.0 - 1.0, self.currentSamplingRate)
  apply_bandstop_filter(df, 49.0, 51.0, self.currentSamplingRate)

  #constructing data matrix
  X = pd.DataFrame()
  for column in df.columns:
    #add custom features here
    X['rms' + str(column)] = rms_convolution(df[column], self.windowSize)

  #automatically add pairwise ratios of all features
  addPairwiseRatios(X)

  #predict based on generated features in X using pre-trained classifier clf
  try:
    X = self.scaler.transform(X[X.columns])
    prediction = self.clf.predict(X)

    #EMBody uses voted predictions by default (over 80ms of data)
    #return prediction to get a result for each sample within data
    voted_prediction = mode(prediction)[0][0]
    self.currentPrediction = str(voted_prediction)
    return self.currentPrediction, prediction
  except ValueError:
    self.currentPrediction = None
    return None, []
```

Listing 1. Excerpt of `makePrediction` highlighting key steps.

3.4.2 *Stream Handling.* In EMBody, the class `StreamHandler` handles the incoming UDP stream from the EMG device as well as the outgoing stream of predictions. Together with the GUI elements, this class implements an observer pattern to inform and update the GUI elements. To that end, `StreamHandler` implements `StreamEventCreator`, allowing it to trigger stream events when required to inform appropriate views. Users who extend this class, or `StreamEventCreator`, are encouraged to deliver GUI updates by notifying their observers.

3.4.3 *GUI Elements.* Similarly, existing GUI elements receive updates by listening to incoming stream events, by implementing `StreamEventListener`. EMBody provides the following functional views [26]:

[23] $C = 1.0$, $\gamma = scale$

[24] https://scikit-learn.org/stable/index.html

[25] By default, every window size (80 *ms*).

[26] Please consult the manual for details at https://github.com/HCUM/embody/tree/master/manual.

- **Setup**: Connecting to the prototype, checking sampling rate and selecting channels.
- **Calibration**: Specifying (save/load) calibration labels, running calibrations and training classifier models; exporting complete calibration data (filtered).
- **Live view**: Live feed of incoming EMG data grouped by channel, additional live predictions if classifier is available.

A possible extension is including a new view, showing a live auto-correlation of the signal for different lag sizes. For this purpose, one would want to connect to the live feed of EMG data (Live view) and subclass `StreamHandler` accordingly to provide custom stream events, after processing the incoming data. A new view may listen for events and plot data when prompted. A view-less listener may simply save processed data to a file.

3.5 Technical Limitations

EMBody is custom-built and tailored for prototyping purposes. As such, it has not been designed as a precise measuring unit, but to support EMG-based interaction in prototypes. The toolkit is a trade-off between signal accuracy and accessibility for non-expert practitioners. Consequently, we note some important limitations of EMBody.

First, EMBody is not an exact measurement device. Due to its open design, it is especially vulnerable to artifacts, such as cable movements or electromagnetic noise. While the implemented filtering steps mitigate these effects, proper setup routines are still vital. Similarly, since EMBody does not provide adjustable gain settings, it represents a trade-off between being able to recognize small, fine-grained motions and extensive movements. The size of the electrodes particularly influences the resulting signal. Hence, users are encouraged to choose electrodes according to the desired usage scenario.

Second, our toolkit works best for isometric muscle activation, i.e. continuous muscle activity without visible movement [27]. Isotonic muscle contraction can be problematic, e.g. recognizing movements over a period of multiple seconds. Here, correlation based on previously calibrated templates might be more suitable. The modular structure of the accompanying software allows for adapting it for that purpose.

Third, for more complex movements, one might require more channels than EMBody can accommodate. Even using up to six channels requires extensive cable management. As EMBody was designed to allow for using all muscles, we do not include cable arrangements in the system. Custom cable solutions can help to alleviate this issue by combining multiple leads.

Finally, despite being a mobile prototype, EMBody still requires an active Wi-Fi connection for broadcasting. Subsequently, a mobile application on a smartphone providing an access point and data processing capabilities is needed for true mobility.

3.6 Target Audience

EMBody is tailored for a wider audience, supporting interaction designers, researchers and engineers. Consequently, EMBody offers different levels of depth and complexity in signal processing and classification. In the following, we outline two typical use case scenarios to showcase the diverse needs of EMBody's users. The first scenario describes how a VR interaction designer uses EMBody to realize dynamic interaction in a sword-fighting game. The latter scenario deals with a researcher collecting electromyograms throughout an experiment for prosthesis control.

3.6.1 Scenario 1: Using EMG Input in VR Prototypes. An interface designer wants to extend their VR application using the standard workflow (Figure 2). They want to sense how strongly the user is gripping the VR controller. The designer decides to include two different grip modes, normal and hard, in their sword fighting game. Thus, grip strength has a direct impact on the sword's momentum, influencing the player's ability to attack and parry. This scenario is an example for EMBody's potential user group composed of interaction designers and application developers. Their requirements include easy access to interpreted EMG data, while allowing for fast iterations among possible gestures. The underlying processing and interpretation of the EMG data is secondary and not of interest to this user group.

The designer first needs to find a suitable electrode location. A pragmatic approach to this question is to observe one's own muscle movements and place a pair of electrodes on the involved muscles. A tight grip mostly activates muscles in the forearm, hence the designer decides to place electrodes on the underside of the respective forearm (Figure 3). Afterwards, they complete the setup by connecting the electrodes to the prototype as depicted in the manual[27]. The designer specifies the two recording channels (one for each forearm) and checks the signal via the live view. They observe whether their movements trigger changes in the displayed signal as illustrated in Figure 4.

In his application, the designer is only interested in detecting a tight grip with either hand. Thus, they provide the labels: *LEFT* and *RIGHT* and create an empty calibration. EMBody automatically adds a *NULL_CLASS*, which represents any other motion. The designer starts the calibration process and provides a tight hand grip when prompted (Figure 5), making sure to relax and perform other relevant sword swinging motions in between.

EMBody reports an average accuracy of 95.6%. If need be, EMBody provides them with the means to reiterate the gesture calibration using a different electrode configuration or gesture labeling. Here, the designer is satisfied with the result and starts the live classification. EMBody now switches to the live view (Figure 6) and continuously processes the incoming EMG data, predicting if the user tightens his grip for either hand. The designer incorporates the live classification into their VR application by accessing the UDP network stream in Unity. They affix the prototype on the belt of the user and put the powerbank in their trouser pocket. As EMBody continues to relay live classifications of grip force, the designer can focus on tweaking parameters relevant for the game, e.g., how much stronger a strike should be when the sword is tightly gripped.

3.6.2 Scenario 2: Designing Experiments for EMG-based Input. A researcher familiar with Electromyography recording wants to use EMBody to find a suitable set of classification features to recognize ten different hand movements when using up to six channels connected to the forearm. Contrary to the designer scenario, this user group includes signal processing experts and researchers as well as engineers and machine learning developers. They require close control over the data processing pipeline allowing them to customize vital steps if need be. Rapid prototyping of different designs is secondary. This user group focuses on signal accuracy and body physiology, requiring detailed views of the recorded data.

Since flexion and abduction of the wrist as well as controlling finger movements involved various muscles in the forearm, the researcher decides to place the electrodes in two rings around the forearm (one closer to the elbow, one near the wrist) to capture most of the involved muscles. This also helps them to generalize their approach more easily, as exact knowledge of the forearm anatomy is thus not required to place electrodes. Contrary to scenario 1, the researcher makes use of the unipolar measurement technique[28] as shown in Figure 10. They use a custom hardware device to collect the data and send a UDP stream adhering to EMBody protocol to deliver the data.

The researcher connects their UDP stream to EMBody and selects all recording channels. In the live view (Figure 4), they observe whether their movements trigger changes in the displayed

[27] Available at https://github.com/HCUM/embody/tree/master/manual.
[28] one GND/REF electrode each, several measurement electrodes.

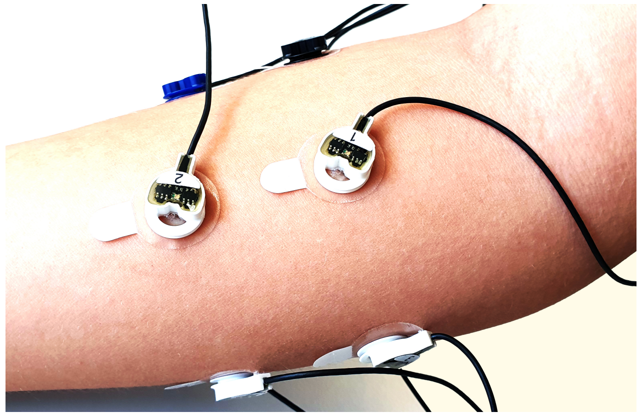

Fig. 10. Unipolar measurement with six sensing electrodes (white, two on backside) and one reference electrode (blue). Ground electrode (black) serves for noise reduction only.

signal and initially assess the viability of the current electrode configuration by comparing different channels and respective signals. The researcher notes possible improvements regarding the configuration but decides to do an initial recording. They define the ten gestures and start a complete calibration process. Using EMBody's save functionality, the researcher exports a dataset with annotated ground truth from the calibration. They repeat the process with two different electrode configurations and fine tune their classification algorithm and extracted features in their own work environment. Here, EMBody provides this user group with easy access to annotated data streams for research purposes.

After establishing a sufficient model, the researcher incorporates their classification algorithm into EMBody by extending its *ClassificationManager*. Henceforth, EMBody will use the tailored algorithm to process incoming data, taking care of signal acquisition and routing. For the actual experiment, the researcher attaches the electrodes in the optimal configuration for every participant and executes the calibration procedure. The researcher connects to EMBody live classification via UDP and relays the current prediction to the prosthesis. They record accuracy metrics in a manual task and questionnaire responses for later analysis.

4 FORMATIVE EVALUATION

Organizing two workshops which featured rapid prototyping of EMG interfaces was a key element in designing and implementing EMBody. This way we assured that the final version of our system reflected the needs of the HCI community. In this section, we illustrate how we established requirements and challenges for EMBody during the workshops. We used formative evaluation to understand the qualities necessary for EMBody to enable exploring EMG-based systems.

4.1 Workshops: Initial Feedback and Refining Requirements

We organized two experimental workshops titled "Using Physiological Sensing for Embodied Interaction" for university students in HCI. Thirty-one and 36 students, including bachelor, master, and Ph.D. students participated in the workshops. Figure 11 shows one workshop location during the hands-on sessions. Participants learned about physiological sensing (mainly EMG) and created their own EMG-controlled devices in hands-on tutorials. The workshops provided an opportunity to verify whether an initial version of the toolkit offered easy entry for EMG-based interface design. It included a hardware prototype using the Bluetooth protocol to transmit data and a set of processing

Fig. 11. Hands-on session during one of the workshops.

script to receive the data. After being presented with the system and its functionality, participants were instructed to first define their own ideas (e.g. an EMG-controlled musical instrument) and formulate a concept for their prototype. Over the course of the two-day workshops, participants successfully developed fully functional prototypes, which they evaluated in small user studies. Finally, each student team presented their work in front of the students and teachers.

The goal of the workshops was to evaluate the suitability of the toolkit for rapid prototyping and identify possible design flaws. Every participant group was able to successfully create a fully functional prototype system that employed EMG as an input modality. Examples[29] included:

- *SmartSpine*: helping the user to correctly lift heavy loads by placing electrodes on the legs and back.
- *Flappy Bird*: controlling the game Flappy Bird via flapping one's arms (electrodes on the arms).
- *Muscle PIN*: biometric authentication via muscle flex patterns (electrodes on the forearm).
- *Dance Avatar*: a puppet mimicking the user's every move; electrodes on arms and legs.
- *Lunar Lander*: a collaborative game where players control a lunar lander probe. Two players steer the probe via electrodes on the forearm.
- *Canoeing*: a four-player game where teams of two compete in a canoe competition. Each team needs to maintain a consistent paddling motion (electrodes on the forearm.)

The breath of ideas generated in the workshops as well as the fact that students at varying levels of HCI and technical competence were able to rapidly build functional systems show that the initial toolkit effectively supported rapid prototyping. Most importantly, the workshops enabled us to verify if the toolkit fulfilled the requirements and what parts of EMBody needed improvement. The prototypes were highly *mobile*—workshop participants built prototypes using different muscles and in different location. Workshop attendees were also able to effectively perform *data acquisition* as they all successfully connected muscle sensing to application input. All the members of the

[29]Selected examples are shown in EMBody's video at https://github.com/HCUM/embody.

diverse audience in the workshops were actively involved in building the prototype, thus showing that the toolkit offered an *abstraction* level that was effectively used by the participants. Finally, participants with expert signal processing knowledge were able to add advanced computation to their prototypes by taking advantage of the toolkit's *modularity*.

The workshops also enabled us to identify key areas for improvement for EMBody. First, some participants experienced issues with Bluetooth connectivity. To alleviate that issue, we redesigned the toolkit to rely solely on the WiFi connection. Second, we observed that the workshop participants spent a significant part of their prototyping time designing algorithms for recognizing movements. This was especially true for those who did not have extensive signal processing experience. Consequently, we decided that EMBody should include pre-defined gesture detection tools that could be customized by expert user. The updated software package includes a default classifier that can be used with no knowledge of EMG gesture recognition.

4.2 Sample Applications

The next step in our process was to develop systems that would enable experimental studies. To demonstrate EMBody's versatility and verify the correctness of the workflow, we built several interactive systems which used EMG input for different purposes.

4.2.1 Choosing the Right Input Control for EMG. To better understand a user's perception of their own muscle control, we endeavored to investigate how users perceive different input control mappings in a steering law experiment. The goal was to keep a moving ball as close as possible to a predefined trajectory (Figure 12). Two EMG channels were used, each controlling one horizontal direction, while the ball moved upwards on a screen. Electrodes were placed on the respective forearm. The study apparatus is depicted in Figure 12.

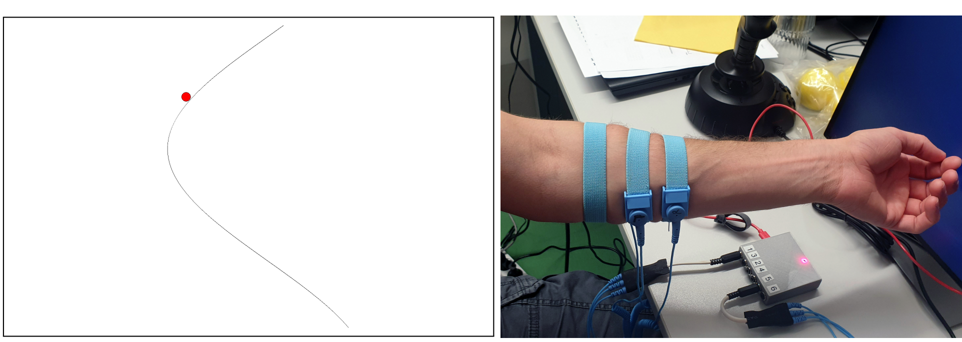

Fig. 12. Study stimulus (left) and electrode placement (right) for EMG input controls.

Here, EMBody allowed us to test a series of electrode locations and suitable muscle groups as well as various input mapping functions in a rapid fashion for a final study. We conducted a within-subject experiment using three different modalities to control the ball: a joystick as baseline and two EMG-based controls (position and rate control). While position control directly changed the ball's position based on the recorded power of muscle activation (controlling its velocity), rate control influenced the acceleration in either direction. Different ball speeds were introduced as an additional independent variable.

We measured the average deviation from the given line as well as participants' responses to our questionnaire about ease of use, their perceived fatigue and their feeling of control[30].

[30] All on a visual-analog scale from 0 to 100.

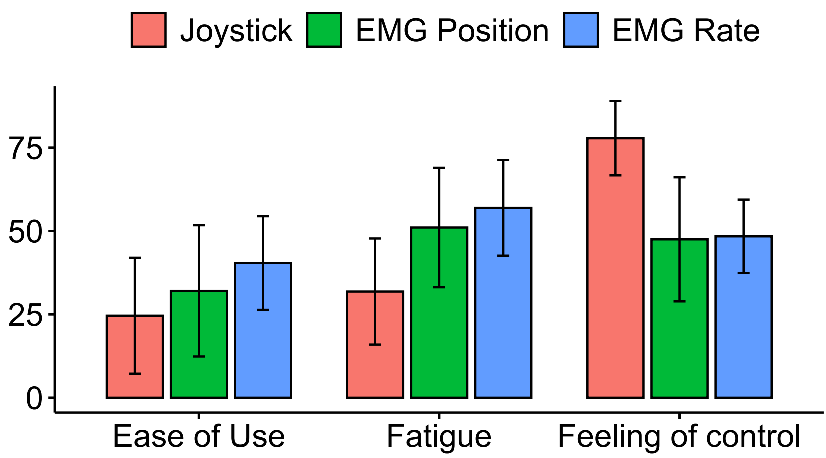

Fig. 13. Averaged responses for our questionnaires (ease of use, fatigue, feeling of control) given modality. Visual-analog scale from 0 to 100.

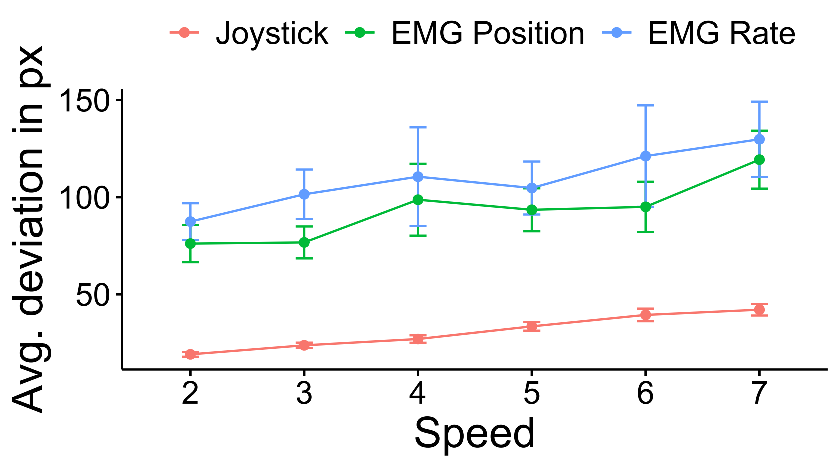

Fig. 14. Average Deviation (L2 norm) in pixels given different speeds and modalities.

Figure 13 illustrates the questionnaire responses w.r.t. the modality. While the Joystick clearly outperformed the EMG-based controls in terms of control and fatigue (lower is better), EMG-based inputs were preferred in terms of ease of use. In a preliminary evaluation (one-way ANOVA and Tukey posthoc comparisons) of ten participants ($8m/2f$), we found that the feeling of control via the joystick was significantly higher ($F(2,27) = 7.7, p < 0.01$) than for both EMG-based controls. Additionally, rate control was significantly ($F(2,27) = 3.4, p < 0.05$) more fatiguing than the joystick.

The superior control of the joystick baseline can be seen in Figure 14 (significantly different to position and rate control). We observed a significant linear effect[31] of ball speed for all modalities.

[31]Linear mixed model analysis. Fixed: modality, speed. Random: trialnumber, participant.

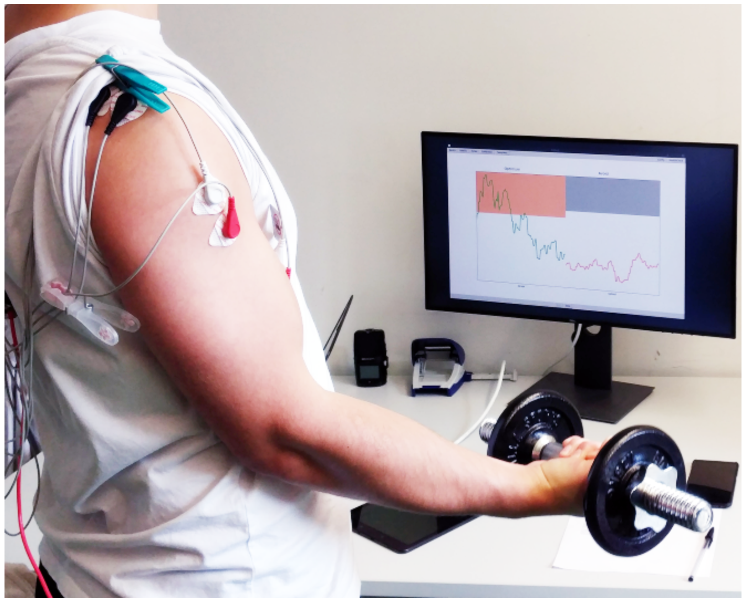

Fig. 15. Study apparatus for the fitness exercise. The monitor displays visual feedback during the exercise.

There was no interaction effect between speed and modality. Additionally, position control did not significantly outperform rate control for EMG-based input.

While EMG-based input was lacking in control for steering tasks, there was a tendency for improved ease of use. In line with this finding, we believe that EMG is better suited as a secondary input modality, e.g. for hands-free interaction. Here, EMBody supports designers in prototyping the right placement for electrodes while ensuring adequate control.

4.2.2 Taking a Look Inside. Besides explicit interaction, EMBody allows exploring EMG for implicit interaction. Here, we evaluated to what degree insights into one's own muscle activation can be beneficial in learning motor tasks. To that end, we built an assistive system for a fitness exercise (bicep curl), which provided visual and audio feedback for two muscle groups. The study apparatus, showing one of the feedback options in the background, is shown in Figure 15. Here, we used EMBody to quickly connect recognized muscle motions to alternative feedback modalities via the UDP stream. This allowed us to focus on possible designs for the feedback.

We designed alternatives for visual feedback: abstract (bars indicating muscle power) and raw (a time series plot which showed a smoothed EMG signal). Auditory Feedback included sounds for correct and incorrect execution, which was detected using an adapted version of EMBody's machine learning algorithm. Here, EMBody allowed us to evaluate several feedback options for a specific task, while relying on the same mobile signal acquisition system. Changing feedback was easily possible and only depended on how we wanted to present the muscular activity. Preliminary results showed that an abstract representation of muscle activity was better understood by users. Expert users approved of using auditory feedback to control for their movements. Interestingly, despite its low-cost components, the system was able to outperform an expert reviewer (a professional fitness coach) during the performed fitness exercise, detecting incorrect execution in nearly all cases. This showcases not only the potential of physiological sensing for interaction, but, most importantly, assures that our low-cost toolkit provides adequate signal-to-noise ratio for prototyping purposes.

5 SUMMATIVE EVALUATION

During the development process, we iteratively refined EMBody to address upcoming challenges. In line with Ledo et al. [22] who suggested multiple evaluation strategies and goals for HCI toolkits, we first evaluated an instance of the *usage* of EMBody in workshops, as described above. In order to establish the capabilities of the final version of the prototype, we further evaluated the toolkit through a series of expert interviews.

5.1 Participants

We recruited five HCI experts who participated in at least one of our workshops. All participants were male and aged $\bar{x} = 27.6\,y$. No remuneration was provided for the interview. Table 1 details the profiles of the participants. We chose interviewees so that they would be member of the primary target audience of the toolkit—HCI researchers with varying level of technical knowledge and different research foci. The participants had varying levels of experience with prototyping. Apart from one, all interviewees were prototyping at least once a month involving microcontrollers, AR/VR applications and small electronics projects.

ID	Age	Gender	Profession	Areas of expertise	Prot. experience	Prot. frequency
P1	31	male	PhD student	UX design, software development	7	Once a month
P2	29	male	Postdoc	HCI, sports, human physiology	7	Twice a week
P3	30	male	PhD student	HCI, augmented/virtual reality	6	Once a month
P4	22	male	Student	HCI, participatory design	5	Once a week
P5	26	male	PhD student	Machine learning, NLP	3	Once a year

Table 1. Participant profiles in the interviews, including their areas of expertise as well as prototyping experience (7-item likert scale) and frequency.

5.2 Interview script

At the start of the interview, we asked the participants about their experience in the workshops, specifically what challenges they faces while realizing their project. Afterwards, participants watched a video of the final version of EMBody[32]. We then inquired about the participants' initial perceptions of EMBody and its exploration-centered workflow. Next, we asked about the challenges and opportunities they saw in using the toolkit. Finally, we discussed possible applications of EMBody in the participants' research work.

5.3 Analysis

All five interviews were recorded (total duration $1:42\,h$) and transcribed verbatim. We wanted to conduct a focused analysis of a moderate volume of qualitative data. Consequently, we used the pragmatic approach to thematic analysis [1]. We established an initial coding tree by open-coding a representative 20% of the material by two researchers and aligning the codes. The rest of the interviews were then split between the coders and analyzed by a single researcher. In a final session, we refined codes and identified recurring themes in the data.

[32] Available at https://github.com/HCUM/embody.

5.4 Results

The final discussion resulted in the following high-level themes: Gradual Levels of Fidelity, Target Audiences, Transparency and Challenges in Working with EMG. We further detail the contents of each theme in the following.

5.4.1 Gradual Levels of Fidelity. EMBody aims to support a wide range of user groups, from novices to experts (Section 3). In our interviews, participants appreciated the simplicity of starting to work with EMG-based interaction as the toolkit allowed them for focus on designing gesture and movements without the need to bother with signal interpretation:

> *I could simply just start with coding my interaction [...] as a designer don't have to deal with the signal and all.(P2)*

> *This new approach would also enable a bit more non-technical people who actually don't care about a signal, just about the application to get into EMGs sensing. For example, people from non-computational fields that just want to try out something, I think, for them it's really more accessible then. (P5)*

It also became evident that experts benefited from the modular data processing pipeline. Moreover, they would appreciate an in-depth classification report already in EMBody's base version:

> *I think it's important to be able to get the raw data, maybe someone wants to do some work on machine-learning algorithm on it and do something else with this data. I think it's important to have most or all data and have this abstraction. Different people can use it differently, depending on the flow. (P3)*

> *I want it to be as accurate as possible, so I would want to have access to everything to be able to customize everything. (P4)*

5.4.2 Target Audiences. Participants remarked on the suitability of EMBody for different audiences, such as people with no experience in computer science. The exploration-centered workflow allowed them to quickly grasp the idea of EMG-based interaction and supported an easy entry:

> *What would they have to do? Let's say the box itself, the board itself would be nicely presented in a nice cover. You just have to plug it in. Easy, that's something you do all the time. Connect it to wifi, so that probably you just immediately opens the wifi and you can connect it from the computer. That's something you always do, so that should be possible I guess. Then afterwards, installing the application. That's easy to do with the installer. You don't have to compile it. Then that would be obviously very feasible as well (P1)*

However, participants noted that the final version of EMBody focused more on curious audience and researchers who wanted to conduct EMG experiments. Here, participants discussed the possibility of introducing different user modes within the application:

> *I think this is one of the disadvantages and one of the issues. It's not optimized for the public (P2)*

> *What I can imagine is that you have these abstractions layers for expert users that really want to see the signal. They're really interested in how this classification actually works, and you have something, let's say, novice mode where you don't see that. (P2)*

5.4.3 Transparency. Presenting users with the live signal and the respective gesture predictions allowed users to "get a feel of what the signals looks like" (P4). This greatly increased the transparency of the toolkit and its data processing pipeline:

When you're doing live prediction, it's nice to have the signal in front of you and the prediction just next to it, so you know if you've done something wrong. (P4)

I think you get a lot of transparency towards the user, whoever that may be. Could be a regular user, could be a researcher. You see how the signal's actually working, you see the signal, and so on (P2)

Furthermore, EMBody's workflow was immediately recognized, closely following related procedures for prototyping and experimenting with physiological signals:

I think we used the very same procedure. We first set up the prototype, so we connected electrodes, then we– well, first, the gesture we wanted to use were defined beforehand but we still had to calibrate. After the calibration, we trained the model and we started live classification to use a prototype. It's the same workflow. (P4)

Interviewees appreciated that all of EMBody's source code and documentation is open-source, ensuring transparency of the workflow and algorithms:

Yes. I think that's very convenient to have. Also, it's open-source. They can want to extend it I guess. (P1)

(Talking about Myo armband) It was not open source. This is maybe a key difference between the commercial product and your project. They did several gestures, but their accuracy was not very well. It was not really clear how they classified it and so the product failed and the company is no more (P2)

5.4.4 Challenges in Working with EMG. Participants further commented on the challenges they experienced with EMG as a modality. First, electrode location was critical in achieving consistent results and not always straightforward:

There it was like not really clear, "Where do I have to put these two electrodes and where do I have to put the ground electrodes? What does this actually mean? Does this has an influence where I put the ground electrode?" (P2)

Second, the lack of generalizability over multiple persons that is inherent in EMG was difficult to address when prototyping with multiple users:

Overall, what worked very well is that when you put it approximately at the same place, it worked very well again, but only if you put it on the same person. (P2)

Having access to six channels also meant dealing with a lot of cables for the electrodes. Interviewees remarked that this could place a heavy burden on users.

I don't know if you used all of these six channels, but it could be quite heavy on the participant. (P4)

6 DISCUSSION

From an initial set of requirements, we further refined and addressed challenges for EMG-based prototyping and experimentation resulting in the final version of EMBody. Through various stages of evaluations, we confirmed that EMBody meets the requirements. However, our work also highlights ongoing challenging in EMG-based interaction.

6.1 Mobility and Data Acquisition

EMBody's goal is to provide a *mobile* platform, allowing for easy *data acquisition* of muscle activity via electromyograms. Throughout the workshops, we identified that the Bluetooth protocol used in an early version was unreliable and intractable, especially when working in groups and with multiple devices. Hence, the final version of EMBody relies on a WiFi connection using the UDP protocol. This constitutes a compromise between reliability, the necessary setup time and resources. While relying on WiFi for connectivity meant the necessity of increased power (Section 3.3), we found this to be negligible and could keep a small form factor. Our presented sample applications make use of this new prototype, showcasing that *mobile* scenarios, such as fitness exercises, are possible without obstructing the user.

An added benefit of switching to WiFi was the fact that we could realize a simple setup process via a captive portal, allowing users to easily configure the device during first use, without the need to flash the firmware of the microcontroller. Additionally, it simplified prototyping when working with multiple receiver applications. We confirmed the feasibility of this *data acquisition* setup and connectivity via WiFi in the presented sample applications.

Switching to a more powerful microcontroller also meant that we could increase the channel count to a maximum of six channels. We found that the original version using one EMG shield[33] per channel was very cumbersome to use when employing more than two channels. The final version of EMBody natively supports up to six channels for *data acquisition*, thus enabling simultaneous exploration of electrode configurations. The live view of EMBody's software application conveniently allows for visual debugging of these configurations as confirmed by our sample applications and interviews.

6.2 Abstraction Layers through Modular Structure

One major objective during EMBody's development process was to make it accessible to a wide range of user groups, including novices as well as experts in physiological sensing, but also tailoring to different professions, such as designers, developers and engineers. We realized this objective using a modular structure encapsulating EMBody's data-centric processing pipeline (Section 3.4). The software grants experts a high degree of control over how data is processed and interpreted. Likewise, novices are aided by the exploration-centered workflow (Section 3.2), guiding them throughout the creation process, while hiding technical complexities in EMBody's base version. Extensive documentation, including electrode setup and best practices is provided[34]. We first informally verified this procedure during the workshops by guiding the students, confirming its viability. We later successfully instantiated the workflow in our sample applications and confirmed in the interviews that it was comprehensible and easy to follow. Moreover, the workflow closely draws from standard workflows when working with physiological sensing. Hence, experts felt immediately at home and quickly identified extensions points suitable for customization.

6.3 Towards More Accessible EMG Input for HCI

Our toolkit effectively lowers entry barriers for researchers and designers to begin exploring EMG-based input. Our evaluation of EMBody highlighted several challenges for future EMG input systems to further support developing interactive systems.

EMG measurements require placing electrodes on muscles and connecting electrode measurements to measurement units. We observed that researchers using EMBody were eager to experiment with multiple muscles and, consequently, measurement channels. This offers the opportunity of sensing complex movements, effectively increasing the fidelity of the motions that an EMG system can detect. However, a high number of channels results in a high number of cables to be connected, which may be cumbersome. Thus, future EMG toolkits should include advanced cable management. Despite cable-based solutions posing certain problems, cables are still the technology of choice for

[33] https://www.olimex.com/Products/Duino/Shields/SHIELD-EKG-EMG/open-source-hardware
[34] Available at https://github.com/HCUM/embody.

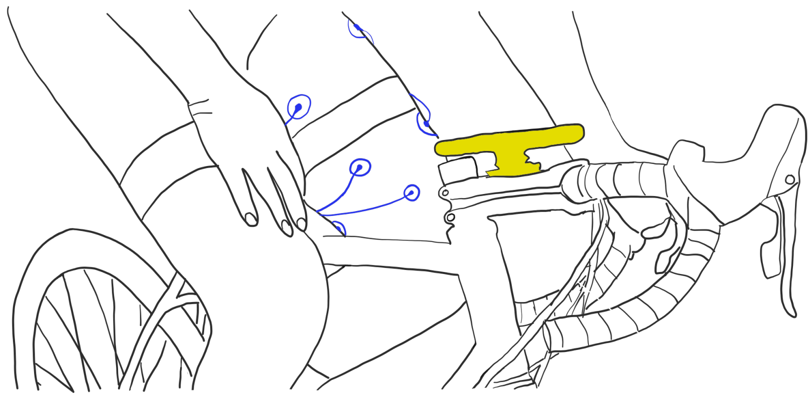

Fig. 16. A speculative future usage scenario for EMBody. A cyclist monitors their quadriceps activity (using electrodes, marked in blue) during a bike ride on their smartphone (yellow). We envision that EMBody will foster experimentation with EMG.

HCI prototyping. While textile wearable electrodes are being researched, they are often muscle-specific and may not support diverse users, e.g. [32]. Wireless electrodes would require individual power source which would increase their mass.

Another finding from the evaluation of our toolkit is the fact that EMG measurements are highly person-dependent. As a consequence, EMG systems require individual calibration and detection is based on values specific to the user. EMBody includes calibration routines, but the need for explicit calibration does increase the complexity of interacting with prototypes which use EMG input. We envision that future EMG tools for HCI researchers should explore if implicit calibration methods can be used. This could be achieved by integrating calibration in tasks. There is a need for developing methods similar to ad-hoc calibration in eye tracking, e.g. [20].

Finally, developing and evaluating EMBody enabled us to observe how designing EMG input was part of an interaction design process. One of the overarching ideas behind our toolkit was enabling designers to focus on the nature of the interaction technique they were designing and emphasize the limitations of the sensing modality. While we did observe that EMBody eliminated initial barriers to using EMG, EMG input still produces additional constraints in the design process. Limiting input to the muscles monitored or the need to place electrodes are likely to have a significant impact on how a design team develops an interactive artefact which uses EMG. Future work should address this challenge and study how designers can consider EMG as a input modality and be implicitly aware of the EMG design space without investing time in extensive EMG prototyping.

7 CONCLUSION AND FUTURE WORK

In this paper, we introduced EMBody—a data-centric toolkit for rapid prototyping and experimentation with EMG. We provided details of the design of the toolkit and information on how to access the open-source resources needed to build it. We also illustrated the utility of EMBody by reporting on workshops with students, presenting two systems that make use of the toolkit and a final evaluation through expert interviews. We concluded that the EMBody toolkit can help practitioners focus on designing the interface and feedback, reducing the need to troubleshoot data acquisition and interpretation. Additionally, experts appreciated its modular structure and data processing pipeline, confirming that EMBody successfully tailored to the needs of a wider audience.

In contrast to off-the-shelf products, EMBody provides a full exploration-centered workflow from data acquisition through calibration and model training for live predictions. We envision that further iterations of our toolkit can be driven by the community, enabling access to electromyography for less technology-proficient practitioners, while allowing experts to benefit from improved algorithms. This would enable end-users to experiment with EMG for understanding their own bodies (see Figure 16 for an example scenario). In combination with recent advances in sensing technologies [13, 30], we hope that our work can help establish EMG as a key modality for future embodied interaction.

8 ACKNOWLEDGMENTS

This research is financially supported by the European Union's Horizon 2020 Programme under ERCEA grant no. 683008 AMPLIFY. Part of this research was supported by the German Research Foundation (DFG) within the project C02 of SFB/Transregio 161.

REFERENCES

[1] Ann Blandford, Dominic Furniss, and Stephann Makri. 2016. Qualitative HCI research: Going behind the scenes. *Synthesis lectures on human-centered informatics* 9, 1 (2016), 1–115.

[2] Marta G. Carcedo, Soon Hau Chua, Simon Perrault, Paweł Wozniak, Raj Joshi, Mohammad Obaid, Morten Fjeld, and Shengdong Zhao. 2016. HaptiColor: Interpolating Color Information as Haptic Feedback to Assist the Colorblind. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 3572–3583. https://doi.org/10.1145/2858036.2858220

[3] Pei-Yu (Peggy) Chi and Yang Li. 2015. Weave: Scripting Cross-Device Wearable Interaction. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 3923–3932. https://doi.org/10.1145/2702123.2702451

[4] Enrico Costanza, Samuel A. Inverso, Rebecca Allen, and Pattie Maes. 2007. Intimate Interfaces in Action: Assessing the Usability and Subtlety of Emg-Based Motionless Gestures. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. ACM, New York, NY, USA, 819–828. https://doi.org/10.1145/1240624.1240747

[5] Paul Dourish. 1999. Embodied interaction: Exploring the foundations of a new approach to HCI. *Unpublished paper, on-line: http://www. ics. uci. edu/˜ jpd/publications/misc/embodied. pdf* (1999).

[6] Stephen H. Fairclough. 2009. Fundamentals of physiological computing. *Interact Comput* 21, 1-2 (Jan. 2009), 133–145. https://doi.org/10.1016/j.intcom.2008.10.011

[7] Tong Gao, Jessica R. Hullman, Eytan Adar, Brent Hecht, and Nicholas Diakopoulos. 2014. NewsViews: An Automated Pipeline for Creating Custom Geovisualizations for News. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 3005–3014. https://doi.org/10.1145/2556288.2557228

[8] Saul Greenberg and Chester Fitchett. 2001. Phidgets: Easy Development of Physical Interfaces through Physical Widgets. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST '01)*. Association for Computing Machinery, Orlando, Florida, 209–218. https://doi.org/10.1145/502348.502388

[9] Tobias Grosse-Puppendahl, Yannick Berghoefer, Andreas Braun, Raphael Wimmer, and Arjan Kuijper. 2013. OpenCapSense: A Rapid Prototyping Toolkit for Pervasive Interaction Using Capacitive Sensing. In *2013 IEEE International Conference on Pervasive Computing and Communications (PerCom)*. 152–159. https://doi.org/10.1109/PerCom.2013.6526726

[10] Juan Haladjian. 2019. The Wearables Development Toolkit: An Integrated Development Environment for Activity Recognition Applications. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 3, 4 (Dec. 2019), 134:1–134:26. https://doi.org/10.1145/3369813

[11] Juan Haladjian, Katharina Bredies, and Bernd Brügge. 2016. Interactex: An Integrated Development Environment for Smart Textiles. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers* (Heidelberg, Germany) *(ISWC '16)*. Association for Computing Machinery, New York, NY, USA, 8–15. https://doi.org/10.1145/2971763.2971776

[12] Björn Hartmann, Leith Abdulla, Manas Mittal, and Scott R. Klemmer. 2007. Authoring Sensor-Based Interactions by Demonstration with Direct Manipulation and Pattern Recognition. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07)*. Association for Computing Machinery, San Jose, California, USA, 145–154. https://doi.org/10.1145/1240624.1240646

[13] Cedric Honnet, Hannah Perner-Wilson, Marc Teyssier, Bruno Fruchard, Jürgen Steimle, Ana C. Baptista, and Paul Strohmeier. 2020. PolySense: Augmenting Textiles with Electrical Functionality Using In-Situ Polymerization. In

Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20). Association for Computing Machinery, Honolulu, HI, USA, 1–13. https://doi.org/10.1145/3313831.3376841

[14] Steven Houben, Connie Golsteijn, Sarah Gallacher, Rose Johnson, Saskia Bakker, Nicolai Marquardt, Licia Capra, and Yvonne Rogers. 2016. Physikit: Data Engagement Through Physical Ambient Visualizations in the Home. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 1608–1619. https://doi.org/10.1145/2858036.2858059

[15] Steven Houben and Nicolai Marquardt. 2015. WatchConnect: A Toolkit for Prototyping Smartwatch-Centric Cross-Device Applications. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 1247–1256. https://doi.org/10.1145/2702123.2702215

[16] Scott E. Hudson and Jennifer Mankoff. 2006. Rapid Construction of Functioning Physical Interfaces from Cardboard, Thumbtacks, Tin Foil and Masking Tape. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology (UIST '06)*. Association for Computing Machinery, Montreux, Switzerland, 289–298. https://doi.org/10.1145/1166253.1166299

[17] Martin Kaltenbrunner and Ross Bencina. 2007. ReacTIVision: A Computer-Vision Framework for Table-Based Tangible Interaction. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction* (Baton Rouge, Louisiana) *(TEI '07)*. Association for Computing Machinery, New York, NY, USA, 69–74. https://doi.org/10.1145/1226969.1226983

[18] Jakob Karolus, Annika Kilian, Thomas Kosch, Albrecht Schmidt, and Paweł W. Wozniak. 2020. Hit the Thumb Jack! Using Electromyography to Augment the Piano Keyboard. In *Proceedings of the 2020 ACM Designing Interactive Systems Conference* (Eindhoven, Netherlands) *(DIS '20)*. Association for Computing Machinery, New York, NY, USA, 429–440. https://doi.org/10.1145/3357236.3395500

[19] Jakob Karolus, Hendrik Schuff, Thomas Kosch, PawełW. Wozniak, and Albrecht Schmidt. 2018. EMGuitar: Assisting Guitar Playing with Electromyography. In *Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18)*. ACM, New York, NY, USA, 651–655. https://doi.org/10.1145/3196709.3196803 event-place: Hong Kong, China.

[20] Mohamed Khamis, Ozan Saltuk, Alina Hang, Katharina Stolz, Andreas Bulling, and Florian Alt. 2016. TextPursuits: Using Text for Pursuits-Based Interaction and Calibration on Public Displays. In *Proc. ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*. 274–285. https://doi.org/10.1145/2971648.2971679

[21] David Ledo, Fraser Anderson, Ryan Schmidt, Lora Oehlberg, Saul Greenberg, and Tovi Grossman. 2017. Pineal: Bringing Passive Objects to Life with Embedded Mobile Devices. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 2583–2593. https://doi.org/10.1145/3025453.3025652

[22] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. Association for Computing Machinery, Montreal QC, Canada, 1–17. https://doi.org/10.1145/3173574.3173610

[23] Johnny C. Lee, Daniel Avrahami, Scott E. Hudson, Jodi Forlizzi, Paul H. Dietz, and Darren Leigh. 2004. The Calder Toolkit: Wired and Wireless Components for Rapidly Prototyping Interactive Devices. In *Proceedings of the 5th Conference on Designing Interactive Systems: Processes, Practices, Methods, and Techniques (DIS '04)*. Association for Computing Machinery, Cambridge, MA, USA, 167–175. https://doi.org/10.1145/1013115.1013139

[24] Javier Marco, Eva Cerezo, and Sandra Baldassarri. 2012. ToyVision: A Toolkit for Prototyping Tabletop Tangible Games. In *Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems* (Copenhagen, Denmark) *(EICS '12)*. Association for Computing Machinery, New York, NY, USA, 71–80. https://doi.org/10.1145/2305484.2305498

[25] Nicolai Marquardt, Robert Diaz-Marino, Sebastian Boring, and Saul Greenberg. 2011. The Proximity Toolkit: Prototyping Proxemic Interactions in Ubiquitous Computing Ecologies. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. Association for Computing Machinery, Santa Barbara, California, USA, 315–326. https://doi.org/10.1145/2047196.2047238

[26] Nicolai Marquardt and Saul Greenberg. 2007. Distributed Physical Interfaces with Shared Phidgets. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction (TEI '07)*. Association for Computing Machinery, Baton Rouge, Louisiana, 13–20. https://doi.org/10.1145/1226969.1226973

[27] Roberto Merletti and Dario Farina. 2016. *Surface Electromyography: Physiology, Engineering and Applications*. John Wiley & Sons.

[28] Roberto Merletti and Philip A. Parker. 2004. *Electromyography: Physiology, Engineering, and Non-Invasive Applications*. John Wiley & Sons.

[29] Michael Nebeling, Theano Mintsi, Maria Husmann, and Moira Norrie. 2014. Interactive Development of Cross-Device User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2793–2802. https://doi.org/10.1145/2556288.2556980

[30] Aditya Shekhar Nittala, Arshad Khan, Klaus Kruttwig, Tobias Kraus, and Jürgen Steimle. 2020. PhysioSkin: Rapid Fabrication of Skin-Conformal Physiological Interfaces. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems (CHI '20)*. Association for Computing Machinery, Honolulu, HI, USA, 1–10. https://doi.org/10.1145/3313831.3376366

[31] Galen Panger. 2012. Kinect in the Kitchen: Testing Depth Camera Interactions in Practical Home Environments. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI EA '12)*. Association for Computing Machinery, New York, NY, USA, 1985–1990. https://doi.org/10.1145/2212776.2223740

[32] G. M. Paul, F. Cao, R. Torah, K. Yang, S. Beeby, and J. Tudor. 2014. A Smart Textile Based Facial EMG and EOG Computer Interface. *IEEE Sensors Journal* 14, 2 (2014), 393–400.

[33] Max Pfeiffer, Tim Duente, and Michael Rohs. 2016. Let Your Body Move: A Prototyping Toolkit for Wearable Force Feedback with Electrical Muscle Stimulation. In *Proceedings of the 18th International Conference on Human-Computer Interaction with Mobile Devices and Services* (Florence, Italy) *(MobileHCI '16)*. Association for Computing Machinery, New York, NY, USA, 418–427. https://doi.org/10.1145/2935334.2935348

[34] Angkoon Phinyomark, Pornchai Phukpattaranont, and Chusak Limsakul. 2012. Feature reduction and selection for EMG signal classification. *Expert Systems with Applications* 39, 8 (June 2012), 7420–7431. https://doi.org/10.1016/j.eswa.2012.01.102

[35] Raf Ramakers, Fraser Anderson, Tovi Grossman, and George Fitzmaurice. 2016. RetroFab: A Design Tool for Retrofitting Physical Interfaces Using Actuators, Sensors and 3D Printing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 409–419. https://doi.org/10.1145/2858036.2858485

[36] Raf Ramakers, Kashyap Todi, and Kris Luyten. 2015. PaperPulse: An Integrated Approach for Embedding Electronics in Paper Designs. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 2457–2466. https://doi.org/10.1145/2702123.2702487

[37] T Scott Saponas, Desney S. Tan, Dan Morris, and Ravin Balakrishnan. 2008. Demonstrating the Feasibility of Using Forearm Electromyography for Muscle-Computer Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 515–524. https://doi.org/10.1145/1357054.1357138

[38] Valkyrie Savage, Colin Chang, and Björn Hartmann. 2013. Sauron: Embedded Single-Camera Sensing of Printed Physical User Interfaces. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. Association for Computing Machinery, St. Andrews, Scotland, United Kingdom, 447–456. https://doi.org/10.1145/2501988.2501992

[39] Valkyrie Savage, Sean Follmer, Jingyi Li, and Björn Hartmann. 2015. Makers' Marks: Physical Markup for Designing and Fabricating Functional Objects. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology* (Charlotte, NC, USA) *(UIST '15)*. Association for Computing Machinery, New York, NY, USA, 103–108. https://doi.org/10.1145/2807442.2807508

[40] Valkyrie Savage, Xiaohan Zhang, and Björn Hartmann. 2012. Midas: Fabricating Custom Capacitive Touch Sensors to Prototype Interactive Objects. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology (UIST '12)*. Association for Computing Machinery, Cambridge, Massachusetts, USA, 579–588. https://doi.org/10.1145/2380116.2380189

[41] B. Schilit, N. Adams, and R. Want. 1994. Context-Aware Computing Applications. In *1994 First Workshop on Mobile Computing Systems and Applications*. 85–90. https://doi.org/10.1109/WMCSA.1994.16

[42] Albrecht Schmidt. 2016. Biosignals in human-computer interaction. *interactions* 23, 1 (Dec. 2016), 76–79. https://doi.org/10.1145/2851072

[43] Martin Schmitz, Martin Stitz, Florian Müller, Markus Funk, and Max Mühlhäuser. 2019. ../Trilaterate: A Fabrication Pipeline to Design and 3D Print Hover-, Touch-, and Force-Sensitive Objects. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems* (Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3290605.3300684

[44] Teddy Seyed, Alaa Azazi, Edwin Chan, Yuxi Wang, and Frank Maurer. 2015. SoD-Toolkit: A Toolkit for Interactively Prototyping and Developing Multi-Sensor, Multi-Device Environments. In *Proceedings of the 2015 International Conference on Interactive Tabletops & Surfaces* (Madeira, Portugal) *(ITS '15)*. Association for Computing Machinery, New York, NY, USA, 171–180. https://doi.org/10.1145/2817721.2817750

[45] C. Estelle Smith, Eduardo Nevarez, and Haiyi Zhu. 2020. Disseminating Research News in HCI: Perceived Hazards, How-To's, and Opportunities for Innovation. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–13. https://doi.org/10.1145/3313831.3376744

[46] Evan Strasnick, Sean Follmer, and Maneesh Agrawala. 2019. Pinpoint: A PCB Debugging Pipeline Using Interruptible Routing and Instrumentation. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*

(Glasgow, Scotland Uk) *(CHI '19)*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3290605.3300278

[47] Paul Strohmeier, Narjes Pourjafarian, Marion Koelle, Cedric Honnet, Bruno Fruchard, and Jürgen Steimle. 2020. Sketching On-Body Interactions Using Piezo-Resistive Kinesiology Tape. In *Proceedings of the 1st Augmented Humans International Conference (AHs '20)*. ACM, New York, NY, USA. https://doi.org/10.1145/3384657.3384774

[48] Dag Svanæs. 2013. Interaction Design for and with the Lived Body: Some Implications of Merleau-Ponty's Phenomenology. *ACM Trans. Comput.-Hum. Interact.* 20, 1, Article 8 (April 2013), 30 pages. https://doi.org/10.1145/2442106.2442114

[49] Marc Teyssier, Gilles Bailly, Catherine Pelachaud, Eric Lecolinet, Andrew Conn, and Anne Roudaut. 2019. Skin-On Interfaces: A Bio-Driven Approach for Artificial Skin Design to Cover Interactive Devices. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology (UIST '19)*. Association for Computing Machinery, New Orleans, LA, USA, 307–322. https://doi.org/10.1145/3332165.3347943

[50] Nicolas Villar, James Scott, Steve Hodges, Kerry Hammil, and Colin Miller. 2012. .NET Gadgeteer: A Platform for Custom Devices. In *Proceedings of the 10th International Conference on Pervasive Computing (Pervasive'12)*. Springer-Verlag, Newcastle, UK, 216–233. https://doi.org/10.1007/978-3-642-31205-2_14

[51] Scott R. Vrana. 1993. The Psychophysiology of Disgust: Differentiating Negative Emotional Contexts with Facial EMG. *Psychophysiology* 30, 3 (May 1993), 279–286. https://doi.org/10.1111/j.1469-8986.1993.tb03354.x

[52] Chiuan Wang, Hsuan-Ming Yeh, Bryan Wang, Te-Yen Wu, Hsin-Ruey Tsai, Rong-Hao Liang, Yi-Ping Hung, and Mike Y. Chen. 2016. CircuitStack: Supporting Rapid Prototyping and Evolution of Electronic Circuits. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology* (Tokyo, Japan) *(UIST '16)*. Association for Computing Machinery, New York, NY, USA, 687–695. https://doi.org/10.1145/2984511.2984527

[53] Michael Wessely, Theophanis Tsandilas, and Wendy E. Mackay. 2018. Shape-Aware Material: Interactive Fabrication with ShapeMe. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology (UIST '18)*. Association for Computing Machinery, Berlin, Germany, 127–139. https://doi.org/10.1145/3242587.3242619

[54] Chi-Jui Wu, Steven Houben, and Nicolai Marquardt. 2017. EagleSense: Tracking People and Devices in Interactive Spaces Using Real-Time Top-View Depth-Sensing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 3929–3942. https://doi.org/10.1145/3025453.3025562

[55] Jishuo Yang and Daniel Wigdor. 2014. Panelrama: Enabling Easy Specification of Cross-Device Web Applications. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Toronto, Ontario, Canada) *(CHI '14)*. Association for Computing Machinery, New York, NY, USA, 2783–2792. https://doi.org/10.1145/2556288.2557199

Received April 2020; revised July 2020; accepted October 2020