Plug-and-Play Gesture Control Using Muscle and Motion Sensors

Joseph DelPreto
Daniela Rus
delpreto@csail.mit.edu
rus@csail.mit.edu
MIT Distributed Robotics Lab
Cambridge, Massachusetts

ABSTRACT

As the capacity for machines to extend human capabilities continues to grow, the communication channels used must also expand. Allowing machines to interpret nonverbal commands such as gestures can help make interactions more similar to interactions with another person. Yet to be pervasive and effective in realistic scenarios, such interfaces should not require significant sensing infrastructure or per-user setup time. The presented work takes a step towards these goals by using wearable muscle and motion sensors to detect gestures without dedicated calibration or training procedures. An algorithm is presented for clustering unlabeled streaming data in real time, and it is applied to adaptively thresholding muscle and motion signals acquired via electromyography (EMG) and an inertial measurement unit (IMU). This enables plug-and-play online detection of arm stiffening, fist clenching, rotation gestures, and forearm activation. It also augments a neural network pipeline, trained only on strategically chosen training data from previous users, to detect left, right, up, and down gestures. Together, these pipelines offer a plug-and-play gesture vocabulary suitable for remotely controlling a robot. Experiments with 6 subjects evaluate classifier performance and interface efficacy. Classifiers correctly identified 97.6% of 1,200 cued gestures, and a drone correctly responded to 81.6% of 1,535 unstructured gestures as subjects remotely controlled it through target hoops during 119 minutes of total flight time.

CCS CONCEPTS

• **Human-centered computing** → *Human computer interaction (HCI)*; **Gestural input**; • **Computer systems organization** → **Robotics**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Gestures, EMG, IMU, Plug-and-Play, Machine Learning, Robotics

ACM Reference Format:
Joseph DelPreto and Daniela Rus. 2020. Plug-and-Play Gesture Control Using Muscle and Motion Sensors. In *Proceedings of the 2020 ACM/IEEE International Conference on Human-Robot Interaction (HRI '20), March 23–26, 2020, Cambridge, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3319502.3374823

HRI '20, March 23–26, 2020, Cambridge, United Kingdom

ACM ISBN 978-1-4503-6746-2/20/03.
https://doi.org/10.1145/3319502.3374823

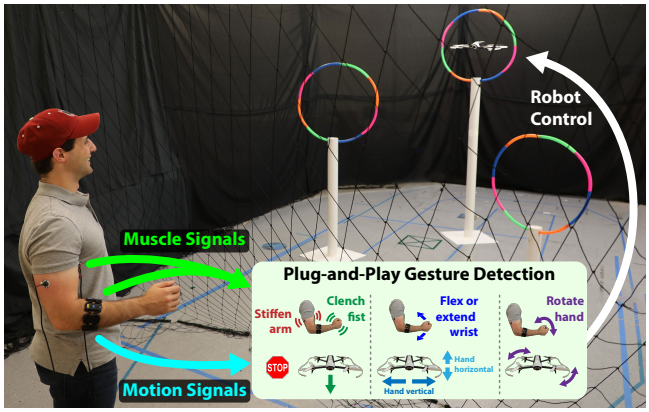

Figure 1: A suite of plug-and-play gesture detectors enables remote robot control without calibration sessions or per-user training data. Eight gestures are classified via wearable muscle and motion sensors on the upper and lower arm.[1]

1 INTRODUCTION

From personal assistants to remote exploration, pervasive robots could extend the capabilities and productivity of casual users. Yet a core question is how to enable effective communication; rather than adapting to the robot, a person should convey intentions as if they were interacting with another person. A variety of techniques have worked towards this goal including speech detection, natural language processing, computer vision, haptics, and gestures.

Gestures in particular could help robots interpret nonverbal cues that people naturally use while communicating, to supplement other modalities or to create a stand-alone interface. However, deploying gestural interfaces for everyday applications with casual users yields a variety of challenges. Interfaces should be plug-and-play, not requiring users to provide extensive training data or perform dedicated calibration routines, and should minimize sensing infrastructure. Despite limited training data, which is generally costly and time-consuming to collect, classifiers must accommodate variations in gestures between users and over time.

The current work takes a step towards addressing these challenges by implementing a plug-and-play gesture vocabulary for robot control using wearable muscle and motion sensors as illustrated in Figure 1. It features detection of arm stiffening to stop, fist clenching to move forward, clockwise/counterclockwise rotation, left/right motion, and up/down motion. Together, these enable real-time remote control of a mobile robot. Using wearable surface

[1] Videos are available at https://people.csail.mit.edu/delpreto/hri2020

electromyography (EMG) sensors and an inertial measurement unit (IMU) eliminates the need for external sensing infrastructure. Muscle signals enable detection of states such as arm stiffness, and can detect hand movements without placing sensors on the hand itself.

This suite of gesture detection pipelines leverages an algorithm for real-time clustering of streaming data that only stores a small number of online observations. This building block can process EMG signals directly, act on synthesized signals to detect more complex gestures, or augment alternative classification methods. It is applied to detect arm stiffening, fist clenching, and forearm activation from EMG signals as well as rotation gestures from processed motion signals. Forearm activation augments a neural network pipeline that detects wrist flexion and extension from EMG signals, reducing false positives due to unrelated user motion. The neural network is trained on selected data from previous subjects, and its classifications are interpreted as left, right, up, or down gestures based on hand orientation detected via an IMU.

In particular, this work contributes the following:

- a vocabulary of 8 gestures for robot control using wearable sensors and not requiring training or calibration sessions;
- an online clustering algorithm, using Gaussian Mixture Models fit in real time to a limited history of observations;
- use of this algorithm to detect arm stiffening, fist clenching, and forearm activation from muscle signals and rotation gestures from motion signals;
- augmentation of a neural network pipeline for detecting wrist flexion and extension, by using forearm activation to reduce false positives and using IMU processing to interpret the indicated gesture direction;
- experiments with 6 subjects that evaluate the plug-and-play gesture classifiers during structured open-loop sessions and during real-time teleoperation of a drone.

2 RELATED WORK

The presented system builds on investigations of EMG for robotic control, gesture detection interfaces, and online clustering.

2.1 EMG for human-robot interaction

Using muscle signals as the basis for human-robot interfaces is promising since they can provide information about a person's motions and physical intentions. Work has been done on modeling these signals and associated muscle dynamics to aid controller development [8, 28, 40, 43, 47, 60]. Prominent applications of EMG include assistive robots [23] such as exoskeletons [1, 19, 24, 33, 36, 45, 49, 59] and prostheses [11, 52], augmented reality interfaces [57], and continuous trajectory estimation [3, 4, 39].

Such studies demonstrate that EMG can yield effective human-robot interfaces, but also reveal challenges such as noise, variance between users, and complex muscle dynamics. Approaches to address these include redundant switching models [39] or leveraging a human within the control loop during physical interaction [18, 36, 46]. Exoskeletons have also used parameterized models [49], neuro-fuzzy systems [33], or impedance controllers [24].

The current work uses an online clustering algorithm to adapt to EMG variations and detect gestures without offline calibration or training. For gestures that are hard to characterize by an adaptive threshold, it leverages a neural network trained on strategically selected data from past subjects. Gestures are used instead of continuous motion to help increase robustness and limit the number of required electrodes. This increases deployability and reduces model complexity, at the expense of restricting interactions to predefined motions. This is acceptable for many robot applications though, and the gesture vocabulary could be extended as needed.

2.2 Gesture detection

There has been significant work on recognizing gestures [44] using modalities such as motion capture or 3D vision systems [27, 30, 37, 54, 56, 58], wearable motion sensors [5, 7, 26, 29, 31, 61, 62], linguistic cues [21], sensorized gloves [10, 20, 32, 63], and capacitive sensors [13]. Building on these investigations, the presented framework focuses on using wearable EMG and motion sensors; this avoids the need for external sensing infrastructure such as cameras or motion capture devices, and is not susceptible to environmental interference such as occlusions or ambient noise. Using muscle signals enables detection of states such as joint stiffness that can be hard to observe via visual sensing modalities. It also enables hand gesture detection via sensors on the forearm rather than the hand itself, reducing inhibitions to dexterity or task performance.

Past work has explored gesture detection via muscle activity [16, 29, 34, 41, 48, 50, 51, 61]. Classification techniques include machine learning pipelines often with manually defined features [12, 13, 29, 34, 41, 51], Hidden Markov Models (HMMs) [31, 50, 61, 62], and dynamic time warping [26]. The current work focuses on detecting gestures designed for robot control from a minimal number of wearable sensors, without requiring per-user training or calibration sessions. It uses an online clustering technique for dynamic thresholding, and augments the neural network pipeline of [16] for increased robustness and additional robot commands.

2.3 Clustering streaming data

There are many techniques for clustering and classifying data [6, 42] including k-means clustering [38], multi-class Support Vector Machines (SVMs), or Gaussian Mixture Models (GMMs). Various algorithms have been presented for incrementally updating GMMs based on streaming data [2, 9, 15, 22, 53], often buffering enough data to fit a new model that can be merged with the existing one or adapting model parameters based on new observations. Key considerations include limiting data storage requirements and ensuring good model fit. Dirichlet process GMMs have also been used offline to threshold sensor data [35]. The presented system builds on these concepts, fitting GMMs in an online fashion to dynamically threshold streaming data. Instead of updating an existing GMM or merging components, the presented approach maintains a limited subset of informative observations such that a new GMM reflecting the desired states can be periodically fit from scratch in real time.

3 GESTURE VOCABULARY AND SYSTEM

A suite of gestures is defined for remote robot control. Gestures are detected via wearable muscle and motion sensors without requiring offline calibration sessions for new users, thus increasing portability and deployability. This gesture vocabulary could be extended in the future to more continuous or user-defined gestures.

3.1 Gesture vocabulary

The presented gesture vocabulary consists of discrete actions, each comprising a brief motion or stiffening and a return to neutral position. The vocabulary enables indication of planar directions, rotation, halting, and selection. While these could potentially be used in a variety of scenarios including navigating menus on an electronic device or supervising an autonomous robot, the current work focuses on remotely controlling a robot. Gestures are mapped to robot actions as described below and illustrated in Figure 1. The detection methodology is described in Section 4.

Arm stiffening: stop. Stiffening the upper arm can be conceptually associated with a cringe response that may be appropriate when noticing an impending robot error or collision. As such, this gesture is interpreted as a stop command. It is detected by estimating arm stiffness from biceps and triceps muscle activity, then applying a clustering-based pipeline that learns an adaptive threshold from the unlabeled streaming data.

Fist clenching: move forward. Squeezing a fist is similar to a grasping motion, which may be interpreted as a selection. In the context of navigation, this gesture selects the current heading and moves the robot forward. It is detected by estimating overall forearm muscle contraction and applying a clustering-based adaptive threshold.

Hand rotation: turn clockwise/counterclockwise. Rotating the hand counterclockwise or clockwise causes the robot to turn in place in the specified direction. These gestures are detected via a gyroscope on the forearm. A priori knowledge of the expected motion is leveraged to synthesize a signal reflecting whether a gesture is present in the streaming data, then the pipeline for learning an adaptive threshold via clustering from few online examples is applied.

Wrist flexion and extension: move left/right/up/down. Wrist flexions or extensions can indicate directional motions; they wave the hand left or right when the hand is held vertically, and down or up when the hand is held horizontally. A neural network classifies inner and outer forearm muscle activity to detect these wrist gestures, which are interpreted as an appropriate direction according to the orientation detected via a forearm accelerometer. To reduce false positives, the online clustering algorithm detects when the forearm is sufficiently activated to plausibly contain a gestural motion.

3.2 System overview

An end-to-end system was implemented as shown in Figure 2 to detect gestures and enable experimental evaluation. It includes wearable muscle and motion sensors, signal processing and classification, an interface to cue gestures if needed, and robot control.

3.2.1 Muscle and motion signal acquisition. Muscle signals are acquired from the user's right upper arm using differential pairs of surface EMG electrodes. Disposable adhesive 24 mm Covidien electrodes with pre-applied conductive gel are placed over the biceps brachii short head and triceps brachii long head, with reference to [14]. Wearable MyoWare Muscle Sensors differentially amplify each pair. An NI USB-6216 data acquisition (DAQ) device samples these two signals at 1 kHz and sends data to Simulink R2018b via USB as buffers of 50 samples every 50 ms. Software-defined signal processing then includes a bandpass filter for 5-400 Hz, which retains

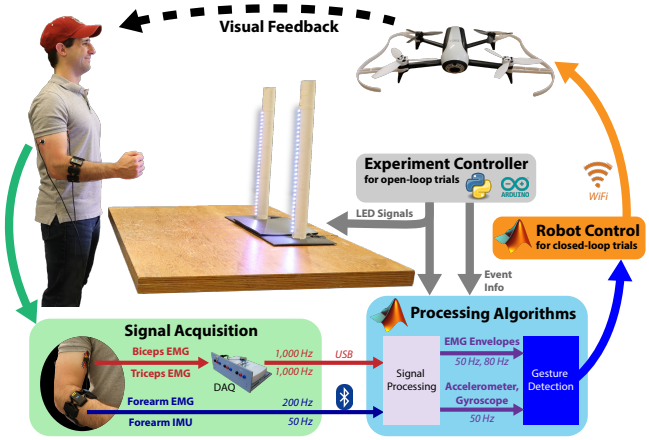

Figure 2: The closed-loop system consists of EMG and IMU acquisition, classification pipelines, and robot control. LEDs cue gestures during open-loop blocks.

usable EMG frequencies [14], and envelope detection (full-wave rectification followed by a 5 Hz lowpass filter).

Muscle and motion signals are acquired from the user's right forearm via the Myo gesture control armband from Thalmic Labs. Eight pairs of dry EMG electrodes surround the user's upper forearm. The device differentially amplifies and normalizes each signal pair, sampling muscle activity at 200 Hz. The device also contains an IMU with a 3-axis accelerometer and a 3-axis gyroscope, which is sampled at 50 Hz. Sensor data is transmitted via Bluetooth, and streamed to background logs within Matlab R2018b using the infrastructure provided in [55]. Custom Simulink wrappers collate these logs into streamable buffers every 200 ms, containing 40 samples of each EMG channel and 10 samples of each IMU channel. Software-defined filtering and envelope detection is then applied as above to each EMG channel, except with a bandpass range of 5-100 Hz due to the lower sampling frequency.

3.2.2 Experiment controller. To facilitate controlled evaluation of the gesture classifiers, a user interface cues gestures at specific times when the robot is not being used. Two pillars with LED strips are placed on a table in front of the subject, and a Python program controls them via an Arduino Mega 2560 to implement the experimental paradigm. The program also sends status codes describing the state of the experiment, such as the cued gesture and trial timing. The Arduino reflects these codes on an 8-bit parallel port, which is sampled by the DAQ along with the LED control signals and the upper-arm EMG data. This aligns experimental information with biosignals to facilitate offline performance evaluation.

3.2.3 Robot control. To evaluate the gesture vocabulary and detection pipelines for closed-loop robot control, users piloted a Parrot Bebop 2 drone by making gestures at any time. The drone is approximately 35 cm x 43 cm, weighs 500 g, and can fly for approximately 20 min using a 3.1 Ah battery. The drone is controlled via WiFi using the Matlab R2019a Hardware Support Package for Parrot Drones, slightly modified to enable non-blocking commands.

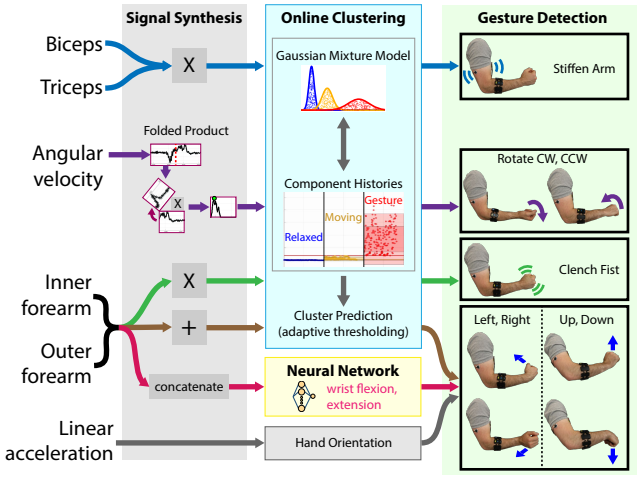

Figure 3: Four instances of the online clustering algorithm are used to detect arm stiffening and fist clenching from EMG signals, to detect rotation gestures from IMU signals, and to augment a neural network classification pipeline by thresholding forearm activation. All together, 8 gestures are detected from 4 EMG signals and 2 IMU signals.

4 GESTURE DETECTION PIPELINES

Two gesture detection methods are used, each with a focus on reducing the need for per-subject calibration sessions or training data collection. A key consideration in both pipelines is therefore strategically selecting training data from a limited pool of available examples. This is done in an online fashion for unsupervised adaptive thresholding via clustering, and in an offline fashion for training neural networks on a subset of data from previous subjects. These two approaches are applied to the chosen gesture vocabulary as illustrated in Figure 3 and described below. Figure 4 illustrates the signals on which these pipelines operate.

4.1 Online clustering for adaptive thresholding

An approach is presented to enable gesture detection via dynamic thresholding that adapts to streaming data from very few unlabeled gesture examples. At its core is a real-time clustering method outlined in Algorithm 1. Given a stream of data, it maintains a cluster model that represents key user states without storing or processing all past data. The most recent cluster model is used to classify new points and to decide which samples should be stored in a limited history. The current implementation assumes the number of states k is known ahead of time, but this could be relaxed in the future.

To create a subset of observations that still represents key user states, the algorithm maintains $k+1$ rolling buffers of duration D seconds: one for each of the k components and one for points not classified as a known component. This limits the amount of stored data to $(k+1)\,D$ seconds, allowing a model to be fit within a single system timestep. Since each buffer exclusively stores observations that were classified as a certain component, new examples of one state will not overwrite examples of another state; sporadic gestures will remain in the history until more are made or until the model readjusts enough to re-classify those points as a different state.

Algorithm 1: Online Clustering of Streaming Data

Input: n-dimensional signal sample $sample^{1\times n}$
Params: k, history length H, T_{refit}, $T_{initialize}$, $zscore\_bounds$
Output: cluster prediction $K_{predicted}$

Initialize $k+1$ rolling buffers: $histories^{(k+1)\times H\times n}$
while current time < $T_{initialize}$ **do**
  $histories[k+1] \leftarrow sample$ // store as unknown cluster
if T_{refit} seconds since last GMM fit **then**
  $allSamples^{H(k+1)\times n}$ = concatenate all $(k+1)$ $histories$
  gmm = FitModel($allSamples$, k)
  $K_{all}^{H(k+1)\times 1}$ = PredictCluster(gmm, $allSamples[i]$)$\forall i$
  $histories[c] = allSamples[K_{all} == c] \forall c = 1 \ldots k$
  $histories[k+1] = []$
$K_{predicted}^{1\times 1}$ = PredictCluster(gmm, $sample$)
$histories[K_{predicted}] \leftarrow sample$
Function PredictCluster(gmm, $sample$):
  $[posteriors^{1\times k}, zscores^{1\times k}]$ = EvalGMM(gmm, $sample$)
  $K_{predicted}$ = cluster with $max(posteriors)$
  if any of $zscores$ violate $zscore\_bounds$ for $K_{predicted}$ **then**
    $K_{predicted} = k+1$ // mark as unknown cluster

Initially, all observations are stored in the *unknown* buffer since no model has been trained. After $T_{initialize}$ seconds, the collected data is clustered into k components using a Gaussian Mixture Model (GMM). Each new streaming observation is then classified by this model and added to the appropriate rolling buffer. Every T_{refit} seconds, the model is refreshed: all rolling buffers are flushed into a common data pool, a new GMM is fit, and the new model is used to re-populate the rolling buffers by re-classifying all stored data.

To classify a point using the current GMM, the posterior probability of belonging to each of the components is computed and the most likely component is selected. Additional criteria can also be supplied to the algorithm that filters classifications based on a point's z-scores relative to each component distribution. If a point's z-scores do not satisfy provided bounds, it is labelled as *unknown*. For example, a detector may specify that predicting component $K = 3$ requires a z-score relative to component 2 that is greater than 1. This allows gesture detectors to be more restrictive when false positives are potentially dangerous for the task, or to mitigate the impact of transient spikes in the data on the fitted model.

Note that while the current implementation employs GMMs, other techniques could also be used such as those mentioned in Section 2.3. The computational complexity of the pipeline is dominated by that of the chosen cluster technique, while the limited amount of stored history limits the maximum processing time.

When domain knowledge about expected signal characteristics can be leveraged to cast a gesture detection problem as one of clustering or dynamic thresholding, the above algorithm can enable user-independent detection that essentially performs calibration online throughout the task. It can quickly tune itself to a new user's signal levels from one or two gesture examples, and adapt to signal variations due to gestural preferences, fatigue, EMG drift, or task interactions. The presented system uses this approach to detect arm stiffening, fist clenching, forearm activation, and rotation gestures.

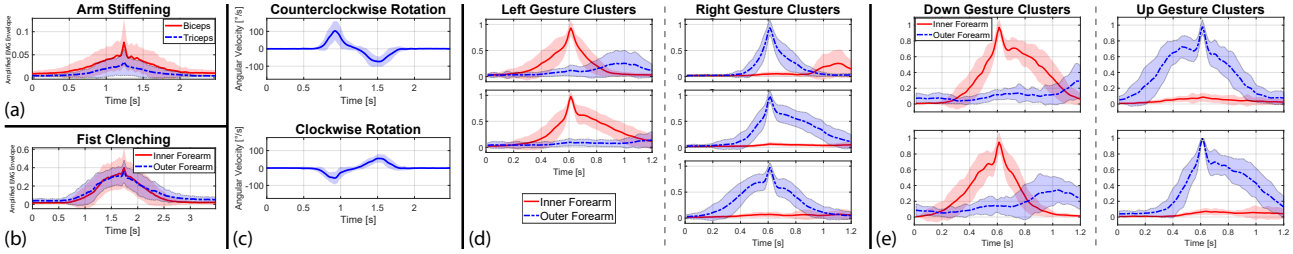

Figure 4: Signal traces from all subjects are summarized for (a) arm stiffening, (b) fist clenching, (c) rotation gestures, (d) clusters of left and right gestures, and (e) clusters of up and down gestures. Shading spans 1 standard deviation on each side of the bolded mean. Segments were centered based on their peak values. Clustered segments were independently normalized.

4.1.1 Detecting arm stiffness. Biceps and triceps EMG envelopes are multiplied to indicate arm stiffness, which is high when both antagonistic muscles are activated. This signal is downsampled to 50 Hz, and used as input to the online clustering pipeline. The pipeline is configured for $k = 3$ mixture components; in order of ascending mean value of their distributions, these are interpreted as relaxation, arm motion or task involvement, and high stiffening. To accept a classification of a point as high stiffening, it must have a z-score of at least -0.5 relative to that component's distribution. If the z-score is greater than 3, the classification is accepted but the point is excluded from the stored history. These two experimentally determined cutoffs help make the detector slightly more restrictive while eliminating large spikes from the data pool. Finally, the pipeline uses $D = 5$ s, $T_{initialize} = 10$ s, and $T_{refit} = 0.3$ s.

To help smooth predictions, a rolling buffer of 12 classifications (0.24 s) is maintained. A stiffening gesture is declared if at least 5 of them are for the component with the highest distribution mean.

4.1.2 Detecting fist clenching. EMG activity from two channels over the inner forearm are averaged, and activity from two channels over the outer forearm are averaged. The results are multiplied to estimate overall forearm activation, downsampled to 50 Hz, and passed to the dynamic clustering pipeline. The configuration is the same as for stiffness detection, except with $T_{refit} = 0.4$ s. This helps stagger refitting the fist GMM with refitting the stiffness GMM, thereby spreading the processing demands across multiple timesteps of the real-time system.

Fist gesture predictions are smoothed by using a rolling buffer of 25 classifications (0.5 s) and requiring at least 20 of them to be for the mixture component with the highest mean. This filtering is more restrictive than for arm stiffening, since false positives for this detector may result in potentially harmful robot motion.

4.1.3 Detecting wrist motion. To detect whether the forearm muscles are sufficiently activated to reasonably expect a wrist flexion or extension gesture, the inner and outer forearm EMG signals are added instead of multiplied since only one muscle will be dominantly activated for each motion. The thresholding algorithm is then applied as before with $T_{refit} = 0.5$ s, omitting points with z-scores greater than 2 relative to the highest mixture component since fist clenching may cause peaks that would undesirably raise the threshold. Filtering is longer than above, using a 1.2 s window and requiring 0.1 s of activation detections, to accommodate the neural network classification window discussed in Section 4.2.

4.1.4 Detecting rotation gestures. A signal is synthesized that selects for the expected angular velocity profile of rotation gestures. Since a gesture comprises a brief hand rotation followed by a return to neutral, the angular velocity around the axis parallel to the forearm should exhibit two successive peaks of opposite sign. The detection pipeline highlights this shape by computing a "folded product." A rolling buffer with 0.8 s of angular velocity measurements around the forearm axis is split in half, one half is negated and flipped, and then the two halves are multiplied element-wise. Conceptually, this folds the velocity window on itself and multiplies it. If there are two peaks of opposite sign on either side of the buffer's center, this will yield a large positive product. The maximum value of this product is selected from each rolling buffer, creating a signal that is large when the desired angular velocity profile exists. The adaptive clustering pipeline then identifies rolling buffers with a sufficiently high folded product to declare a rotation gesture. The same parameters are used as described for arm stiffening, except with $T_{refit} = 0.6$ s. The sign of the angular velocity at the time of the detected gesture indicates which rotation direction is desired.

Filtering is implemented using a rolling buffer of 5 classifications (0.1 s) and requiring at least 2 of them to be for the cluster component with the highest mean. Additionally, predictions are discarded if the peak angular velocity around the forearm axis is less than $45°/s$, if an angular velocity around the other two axes is greater than $100°/s$, or if the opposite rotation gesture has already been detected within the last 0.5 s. These thresholds, based on initial experimentation, help filter out false positives due to unrelated arm motion or to the armband briefly oscillating after a rotation gesture.

4.2 Neural network classification

For gestures where a priori knowledge about the expected signal shape is unavailable or harder to formulate as a clustering-based detection, a supervised learning approach may be employed. The presented system uses a neural network to detect wrist flexion and extension; while a dominant muscle is expected for each motion, the signals may vary between subjects due to aspects such as amount of antagonistic muscle activation or motion preferences.

The signal processing and classification pipeline described in [16] for left and right gestures is implemented as the starting point for this system. Inner and outer forearm EMG signals are extracted from the Myo armband, envelope-detected, resampled to 80 Hz, and stored in rolling buffers of duration 1.2 s. Each buffer is independently shifted so its minimum is at 0, then both buffers are jointly

normalized so the maximum value is 1. These are concatenated to form a 192-element feature vector, and classified by a neural network with a single hidden layer of size 20.

The network is only trained on data from previous users, and a clustering-based algorithm presented in [17] selects specific training subjects since using all available data may not create the most generalizable classifier. It applies k-means clustering to all known left gestures and all known right gestures separately, using the silhouette method to determine the best number of clusters. It then greedily selects training subjects until each cluster has at least 25% of its gestures included in the training pool; at each iteration, the subject with the most gestures from the least-represented cluster is selected and all gestures from that subject are added to the training pool. This aims to choose subjects that exemplify certain gestural characteristics well; less consistent subjects likely have gestures spread across multiple clusters and are therefore selected for training less often. Using a fixed coverage ratio helps maintain the original gesture breakdown, so more common characteristics are still more prevalent in the downselected corpus.

The current system uses this classification pipeline and training methodology as a building block within the gesture vocabulary, and extends it in two ways: adding a dynamic threshold to reduce false positives, and merging it with IMU processing to interpret its predictions as multiple gesture directions.

While normalizing and shifting each rolling buffer allows the pipeline to be robust to EMG variations such as drift or fatigue, it can lead to false positives when making motions that only slightly contract the forearm muscles. To mitigate this effect, the adaptive thresholding via online clustering described in Section 4.1.3 gates the neural network predictions; if the forearm is not sufficiently activated, then wrist flexion or extension classifications are ignored.

Although the network is trained on examples of gesturing to the left and right, the current system investigates its generalizability to up and down gestures as well. Accelerometer measurements from the Myo armband are used to estimate the angle of the device relative to the downward gravity vector. Wrist motions made when the forearm is horizontal and the palm is facing sideways are interpreted as left/right motions. If the sensor is rotated by more than 25° from this neutral arm position, then classifications are interpreted as up/down motions. Classifications are ignored if the forearm is held vertically ($|a|$ along the forearm axis is at least $0.8g$) or if the arm is significantly moving ($|\vec{a}| >= 0.8g$ or $|\vec{\omega}| >= 40°/s$ where ω is angular velocity). While the primary muscles and the wrist motion are similar in both orientations, holding the arm in a different position and working against gravity may affect the signal characteristics; this will be investigated by inspecting the EMG data and evaluating classification accuracy.

4.3 Hierarchy of gestures for robot control

The gesture vocabulary is currently implemented as a collection of independent classifiers. This facilitates using different classification techniques for different gestures, such as the adaptive clustering or neural networks. It also enables adding new gestures to the vocabulary without needing to adjust existing pipelines or gather extensive training examples. However, this precludes the ability to inherently make gesture predictions mutually exclusive.

The current implementation therefore uses the following priority ordering of gestures when controlling the robot: rotation, stiffening, up/down or right, fist clenching, then left gestures. Gesture predictions are discarded if a gesture higher in the list has been detected within about 0.2 s. This order is chosen based on expected gesture interference, such as rotations moving the biceps muscle or fist clenching activating the forearm, as well as task-specific criteria such as stopping the robot being safety-critical. While this framework is sufficient for the current task, future extensions could investigate alternatives such as a classifier or state machine that acts on all gesture predictions and task knowledge.

5 EXPERIMENTAL PARADIGM

Experiments were designed to evaluate gesture detection during controlled sequences and during unstructured robot control.

5.1 Classifier evaluation: cued gestures

To gather a balanced corpus of examples from all desired gestures and evaluate the classifiers in a systematic manner, subjects were cued to make specific gestures at specific times. These trials were performed in an open-loop manner: the robot was not controlled and subjects did not receive feedback about the classifications.

Two pillars of LEDs were placed on a table in front of the subjects as shown in Figure 2. All LEDs initially blinked three times to indicate the start of a block of trials. Each trial then consisted of specific LEDs illuminating for 1.25 s followed by a 3.0 s delay. Left and counterclockwise gestures were cued by the left pillar of LEDs, while right and clockwise gestures were cued by the right pillar of LEDs. Up gestures, stiffening, and fist clenching were cued by the top two LEDs, and the down gesture was cued by the bottom two LEDs. Subjects were instructed to make their gesture while the LED cue was illuminated, encouraging consistent gesture timing.

For each of the five gesture categories, subjects performed two blocks of 20 trials each. During blocks of paired gestures, the cues alternated between clockwise/counterclockwise, left/right, or up/down gestures. During blocks of stiffening and fist clenching, LED illumination cued tensing and the delay cued resting. Subjects were told to tense a comfortable amount rather than maximum contraction. A soft foam cylinder was provided to hold if desired.

Although each block focused on a single gesture category, all classifiers operated during all blocks to evaluate possible false positives. Before the first trial of each block, subjects performed stiffening, fist clenching, left, right, clockwise, and counterclockwise gestures twice. This brief period, typically about 15 s, provided the only gesture examples that the dynamic clustering pipelines would observe for gestures other than the category being cued in the upcoming trials. While realistic deployments could reload learned GMMs or recent per-component histories as starting points, the current experiments started each pipeline from scratch for every block. This enabled evaluation of multiple GMM adaptations for each subject, and of the algorithm's ability to learn from only two examples.

For the neural network classification pipeline detecting wrist flexion and extension, a new network was trained before each experiment. The clustering-based subject selection algorithm of Section 4.2 operated on all left and right gestures from previous subjects in this study as well as 11 subjects that performed the

paradigm described in [16]. Note that the past study used bar electrodes rather than the Myo armband, so combining data sets helps investigate generalizability of the pipeline to alternative sensors.

Performance metrics were computed using the LED signals as ground truth. A gesture prediction with a rising edge during the LED cue or within 1.0 s after the cue turned off is matched with that cue and marked as a true or false positive according to the true and predicted labels. The grace period after the cue turns off accommodates subject reaction times and classification delays incurred by filtering or populating a classification window. Gesture predictions not within a cue window are considered false positives. A cue associated with predictions of multiple different gestures from a classifier is considered an incorrectly classified gesture.

5.2 Robot control: arbitrary gestures

To evaluate the efficacy of the gesture vocabulary and its detection pipelines for realistic robot control, subjects were allowed to make any gesture at any time to remotely control a drone. Left, right, up, or down gestures would cause the drone to move in that direction for 1.5 s. Fist clenching caused forward motion for 2.0 s, and rotation gestures caused 30° yaw rotations. Gestures could interrupt an ongoing motion, such as arm stiffening causing the drone to stop and hover. Motions were discretized due to space constraints and safety concerns, but could be made continuous in the future.

The drone was placed in a 5.5 m x 7 m area in front of the subject surrounded by safety nets, as seen in Figure 1. Subjects piloted the drone throughout the empty area during the first control session, then three hoops were placed in the space as navigational targets. Hoops were 0.8 m in diameter with their bottoms elevated approximately 0.6 m, 1.2 m, and 1.8 m above the floor.

5.3 Survey data

Subjects completed a brief questionnaire after the experiment. It included the NASA-TLX workload assessment [25], a rating of how easy the system was to learn, and a rating of how natural the interaction with the robot was once the system was learned.

5.4 Subject selection

Six subjects recruited from MIT's campus participated in the experiments (5 male, 4 right-handed). Subjects were not screened based on criteria such as muscle signals, gesture performance, or relevant previous experience including drone control. All participants provided written consent. Experiments were approved by MIT's Committee on the Use of Humans as Experimental Subjects.

6 RESULTS AND DISCUSSION

Experiments with 6 subjects investigated the accuracy of each classifier as well as overall system efficacy for real-time robot control.

6.1 Cued open-loop gestures

Figure 4 summarizes the signals for each gesture during all blocks of cued gestures. Each gesture exhibited common patterns in the dominant muscle or motion signals, but there are also noticeable variations. For example, clustering left and right gestures as described in Section 4.2 identifies prominent antagonistic muscle activity and motion onset speed as key gesture characteristics. Clusters

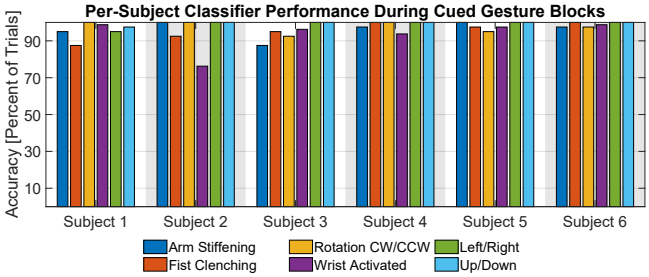

Figure 5: Classifiers successfully identified their respective gestures during cued blocks for each subject. Each bar summarizes 40 trials, except wrist activation which has 80 trials.

of up and down gestures reveal strong antagonistic activity when returning to neutral from a down gesture, but hardly any during up gestures; this reflects the motion acting against or using gravity.

These clusters were used to greedily select key training subjects, and the performance of this methodology can be compared offline to training on all previous subjects. As described in Section 5.1, left and right gesture examples are available from 17 total subjects. Each one was iteratively removed from the corpus as a hypothetical new subject, and both the clustering selection approach and training on all past subjects were performed on the remaining 16 subjects. The clustering approach chose 5 training subjects every time, based on an average of 2.2 (Standard Deviation=0.4) and 2.9 (SD=0.2) left and right clusters, respectively. It averaged 93.5% (SD=2.4%) accuracy on the holdout subjects with a minimum of 90.7%, while training on all past subjects averaged 89.0% (SD 18.6%) with a minimum of 32.7%. While the distributions were not significantly different for the current sample size, the clustering approach increased consistency while using less than a third of the available training data.

In addition to the signal shape, the signal magnitude varied between subjects. Within a single block of trials, the overall level of arm stiffness was computed as the median of the per-trial maximum stiffness. To compare this between subjects, each block's stiffness level was normalized by the highest such stiffness level found across all blocks. This normalized level averaged 31.8% (SD=31.7%) with a minimum of 6.7%. Applying the same methodology to the product and sum of the forearm signals yields normalized levels averaging 47.4% (SD 27.0%) and 73.8% (SD 14.4%), respectively. These wide distributions indicate that the adaptive thresholding pipelines needed to operate on EMG signals of varying levels for different subjects.

Signal levels also varied over time. Per-trial stiffness levels ranged from 9.3% to 718.4% of the median stiffness level in its block. The same methodology applied to the product and sum of the forearm signals yielded ranges of 35.5% to 356.1% and 29.9% to 186.6%, respectively. These results indicate that signal levels exhibited large variations even among trials from a single subject, such that finding a suitable fixed threshold may have been challenging.

Despite varying signal levels, the system successfully identified cued gestures; across all subjects and gesture types, 97.6% of 1,200 gestures were correctly identified by the respective classifier. Figure 5 depicts the accuracy of each classifier for each subject, demonstrating relatively consistent performance across subjects. Although the current subject pool is relatively small, these results

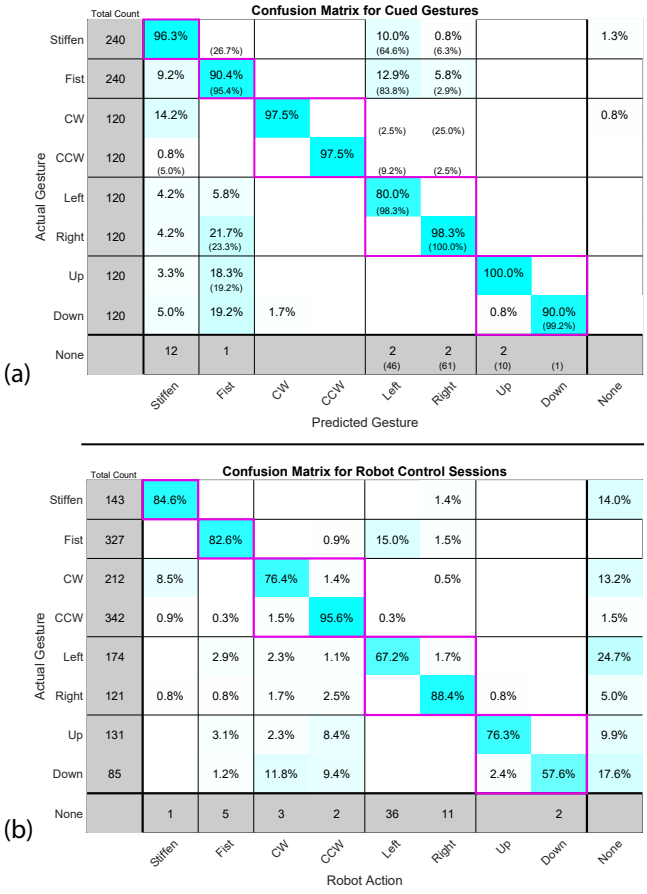

Figure 6: Confusion matrices illustrate performance during (a) open-loop gesture blocks and (b) closed-loop robot control blocks. Main numbers and coloring represent final outputs, while parenthetical values in (a) represent predictions before a gesture hierarchy is imposed and before wrist flexion and extension are gated by an adaptive threshold.

are promising and suggest that the online clustering pipelines successfully produced adaptive thresholds across subjects and over time. The neural network also identified wrist flexion and extension regardless of hand orientation, even though its training data only included left and right gestures and spanned multiple EMG devices.

As described in Section 4.3, classifier outputs can be combined in a hierarchy to help reduce false positives. Figure 6a shows the confusion matrix for cued gesture blocks both with and without this procedure. False positives while stiffening, fist clenching, and rotating decreased dramatically, as did neural network false positives when no gesture was being made. This was at the expense of a lower overall true positive rate of 93.7%, with fist clenching and wrist flexion (both left and down) decreasing the most; this may indicate that learned forearm activation thresholds were occasionally dominated by wrist extension and too high to detect wrist flexion. Yet for real-world robot control tasks, avoiding false positives can be crucial for safety and task success since unwanted robot motion is often more detrimental than repeating a missed gesture.

6.2 Closed-loop robot control

Figure 6b depicts the confusion matrix for closed-loop blocks when subjects made any gesture at any time to control the drone. Videos of the experiments were manually annotated to compare gestures with resulting robot motion. Across all sessions, 81.6% of 1,535 closed-loop gestures were correctly interpreted. Wrist flexion and extension classification decreased notably compared to open-loop trials; subjects may have performed these in rapid succession such that multiple gestures were in a single classification window, or may have used different gesture profiles such as faster motions than the neural network expected. In general there were relatively few robot motions when no gesture was intended, and missed gestures could be repeated to achieve the desired action.

The drone was flown for 25 sessions averaging 4.8 (SD=2.3) minutes each, for a total flight time of 118.7 minutes. Each subject made an average of 255.8 gestures (SD=13.8). The drone passed through a hoop 38 times, navigated around a hoop post 6 times, and hit a hoop 12 times. Note that the drone was approximately 50% as wide as the hoop diameter. Together, these results indicate successful use of the gesture vocabulary for real-time robot control.

Survey results suggest that the interface required a reasonably low workload, and was reasonably easy to learn and natural to use. Raw TLX scores averaged 3.5 (SD=0.7), the learnability score averaged 2.8 (SD=1.9), and the intuitiveness score averaged 2.8 (SD=2.0). All ratings were from 0-10, with lower numbers more desirable.

7 CONCLUSION

The presented system moves towards a deployable gesture-based interface for human-robot interaction by leveraging wearable sensors and plug-and-play detection pipelines. A lightweight real-time clustering algorithm processes streaming data while only maintaining a key subset of observations. This building block is used to adaptively threshold muscle and motion signals in a variety of contexts, detecting arm stiffening and fist clenching from EMG signals and rotation gestures from IMU signals. It also thresholds forearm activation to augment a neural network pipeline for detecting wrist flexion and extension; these enhanced predictions are fused with IMU processing to interpret left, right, up, or down gestures. Experiments demonstrate reliable EMG and IMU classification across 6 subjects, even without dedicated calibration routines or per-user training data. Subjects successfully used the interface for real-time robot control by remotely piloting a drone around obstacles.

Future extensions can evaluate performance with a larger subject pool and can expand the gesture vocabulary. Additional sensors could detect finer muscle or motion activity for richer interactions, and the clustering pipeline could process multi-dimensional signals. Task-specific or adaptive methods of combining predictions from multiple gesture detectors could also improve overall performance.

The presented approach takes a step towards improving human-robot communication in real-world scenarios. As such collaborations continue to become more accessible and pervasive, the possibilities for synergistic benefit continue to deepen.

ACKNOWLEDGMENTS

This work was funded in part by the Boeing Company, for which the authors express gratitude.

REFERENCES

[1] Di Ao, Rong Song, and JinWu Gao. 2017. Movement Performance of Human-Robot Cooperation Control Based on EMG-Driven Hill-Type and Proportional Models for an Ankle Power-Assist Exoskeleton Robot. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25, 8 (Aug 2017), 1125–1134. https://doi.org/10.1109/TNSRE.2016.2583464

[2] Ognjen Arandjelovic and Roberto Cipolla. 2005. Incremental Learning of Temporally-Coherent Gaussian Mixture Models. In *Proceedings of the British Machine Vision Conference (BMVC)*. BMVA Press, 59.1–59.10. https://doi.org/10.5244/c.19.59

[3] Panagiotis K Artemiadis and Kostas J Kyriakopoulos. 2010. EMG-Based Control of a Robot Arm Using Low-Dimensional Embeddings. *IEEE Transactions on Robotics* 26, 2 (April 2010), 393–398. https://doi.org/10.1109/TRO.2009.2039378

[4] Panagiotis K Artemiadis and Kostas J Kyriakopoulos. 2010. An EMG-Based Robot Control Scheme Robust to Time-Varying EMG Signal Features. *IEEE Transactions on Information Technology in Biomedicine* 14, 3 (2010), 582–588. https://doi.org/10.1109/TITB.2010.2040832

[5] Ari Y Benbasat and Joseph A Paradiso. 2001. An Inertial Measurement Framework for Gesture Recognition and Applications. In *International Gesture Workshop*. Springer, 9–20. https://doi.org/10.1007/3-540-47873-6_2

[6] Christopher M Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer.

[7] Sylvain Calinon and Aude Billard. 2007. Incremental Learning of Gestures by Imitation in a Humanoid Robot. In *Proceedings of the ACM/IEEE International Conference on Human-Robot Interaction (HRI)*. ACM, 255–262. https://doi.org/10.1145/1228716.1228751

[8] Ettore Cavallaro, Jacob Rosen, Joel C Perry, Stephen Burns, and Blake Hannaford. 2005. Hill-Based Model as a Myoprocessor for a Neural Controlled Powered Exoskeleton Arm - Parameters Optimization. In *2005 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 4514–4519. https://doi.org/10.1109/ROBOT.2005.1570815

[9] Moses Charikar, Chandra Chekuri, Tomás Feder, and Rajeev Motwani. 2004. Incremental Clustering and Dynamic Information Retrieval. *SIAM J. Comput.* 33, 6 (2004), 1417–1440. https://doi.org/10.1137/S0097539702418498

[10] Jean-Baptiste Chossat, Yiwei Tao, Vincent Duchaine, and Yong-Lae Park. 2015. Wearable Soft Artificial Skin for Hand Motion Detection with Embedded Microfluidic Strain Sensing. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2568–2573. https://doi.org/10.1109/ICRA.2015.7139544

[11] Jun-Uk Chu, Inhyuk Moon, Yun-Jung Lee, Shin-Ki Kim, and Mu-Seong Mun. 2007. A Supervised Feature-Projection-Based Real-Time EMG Pattern Recognition for Multifunction Myoelectric Hand Control. *IEEE/ASME Transactions on Mechatronics* 12, 3 (June 2007), 282–290. https://doi.org/10.1109/TMECH.2007.897262

[12] Beau Crawford, Kai Miller, Pradeep Shenoy, and Rajesh Rao. 2005. Real-Time Classification of Electromyographic Signals for Robotic Control. In *Proceedings of the 20th National Conference on Artificial Intelligence*. AAAI Press, 523–528. http://new.aaai.org/Papers/AAAI/2005/AAAI05-082.pdf

[13] Louis J Dankovich and Sarah Bergbreiter. 2019. Gesture Recognition via Flexible Capacitive Touch Electrodes. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 9028–9034. https://doi.org/10.1109/ICRA.2019.8794202

[14] Carlo J De Luca. 2002. Surface Electromyography: Detection and Recording. *DelSys Incorporated* (2002). https://www.semanticscholar.org/paper/992836d36c1ae4b56314e8b15f087822a1995c93

[15] Arnaud Declercq and Justus H Piater. 2008. Online Learning of Gaussian Mixture Models - A Two-Level Approach. In *Proceedings of the Third International Conference on Computer Vision Theory and Applications (VISAPP)*. 605–611. http://hdl.handle.net/2268/37204

[16] Joseph DelPreto, Andres F Salazar-Gomez, Stephanie Gil, Ramin M Hasani, Frank H Guenther, and Daniela Rus. 2018. Plug-and-Play Supervisory Control Using Muscle and Brain Signals for Real-Time Gesture and Error Detection. In *Robotics: Science and Systems (RSS)*. https://doi.org/10.15607/RSS.2018.XIV.063

[17] Joseph DelPreto, Andres F Salazar-Gomez, Stephanie Gil, Ramin M Hasani, Frank H Guenther, and Daniela Rus. 2020. Plug-and-Play Supervisory Control Using Muscle and Brain Signals for Real-Time Gesture and Error Detection. *Autonomous Robots (conditionally accepted)* (2020).

[18] Joseph DelPreto and Daniela Rus. 2019. Sharing the Load: Human-Robot Team Lifting Using Muscle Activity. In *2019 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 7906–7912. https://doi.org/10.1109/ICRA.2019.8794414

[19] Matthew DiCicco, Lenny Lucas, and Yoky Matsuoka. 2004. Comparison of Control Strategies for an EMG Controlled Orthotic Exoskeleton for the Hand. In *2004 IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 2. IEEE, 1622–1627. https://doi.org/10.1109/ROBOT.2004.1308056

[20] Laura Dipietro, Angelo M Sabatini, and Paolo Dario. 2008. A Survey of Glove-Based Systems and Their Applications. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 4 (2008), 461–482. https://doi.org/10.1109/TSMCC.2008.923862

[21] Jacob Eisenstein and Randall Davis. 2004. Visual and Linguistic Information in Gesture Classification. In *Proceedings of the 6th International Conference on Multimodal Interfaces*. ACM, 113–120. https://doi.org/10.1145/1027933.1027954

[22] Paulo Martins Engel and Milton Roberto Heinen. 2010. Incremental Learning of Multivariate Gaussian Mixture Models. In *Advances in Artificial Intelligence – SBIA 2010*. Springer, 82–91. https://doi.org/10.1007/978-3-642-16138-4_9

[23] RARC Gopura, DSV Bandara, JMP Gunasekara, and TSS Jayawardane. 2013. Recent Trends in EMG-Based Control Methods for Assistive Robots. In *Electrodiagnosis in New Frontiers of Clinical Research*. IntechOpen. https://doi.org/10.5772/56174

[24] Ranathunga Arachchilage Ruwan Chandra Gopura, Kazuo Kiguchi, and Yang Li. 2009. SUEFUL-7: A 7DOF Upper-Limb Exoskeleton Robot with Muscle-Model-Oriented EMG-Based Control. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 1126–1131. https://doi.org/10.1109/IROS.2009.5353935

[25] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In *Advances in Psychology*. Vol. 52. Elsevier, 139–183. https://doi.org/10.1016/S0166-4115(08)62386-9

[26] Bastian Hartmann and Norbert Link. 2010. Gesture Recognition with Inertial Sensors and Optimized DTW Prototypes. In *2010 IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 2102–2109. https://doi.org/10.1109/ICSMC.2010.5641703

[27] Guan-Feng He, Sun-Kyung Kang, Won-Chang Song, and Sung-Tae Jung. 2011. Real-Time Gesture Recognition using 3D Depth Camera. In *2011 IEEE 2nd International Conference on Software Engineering and Service Science*. IEEE, 187–190. https://doi.org/10.1109/ICSESS.2011.5982286

[28] Neville Hogan. 1984. Adaptive Control of Mechanical Impedance by Coactivation of Antagonist Muscles. *IEEE Trans. Automat. Control* 29, 8 (1984), 681–690. https://doi.org/10.1109/TAC.1984.1103644

[29] Yangjian Huang, Weichao Guo, Jianwei Liu, Jiayuan He, Haisheng Xia, Xinjun Sheng, Haitao Wang, Xuetao Feng, and Peter B Shull. 2015. Preliminary Testing of a Hand Gesture Recognition Wristband Based on EMG and Inertial Sensor Fusion. In *Intelligent Robotics and Applications*. Springer, 359–367. https://doi.org/10.1007/978-3-319-22879-2_33

[30] Nebojsa Jojic, Barry Brumitt, Brian Meyers, Steve Harris, and Thomas Huang. 2000. Detection and Estimation of Pointing Gestures in Dense Disparity Maps. In *Proceedings Fourth IEEE International Conference on Automatic Face and Gesture Recognition (Cat. No. PR00580)*. IEEE, 468–475. https://doi.org/10.1109/AFGR.2000.840676

[31] Holger Junker, Oliver Amft, Paul Lukowicz, and Gerhard Tröster. 2008. Gesture spotting with body-worn inertial sensors to detect user activities. *Pattern Recognition* 41, 6 (2008), 2010–2024. https://doi.org/10.1016/j.patcog.2007.11.016

[32] Holger Kenn, Friedrich Van Megen, and Robert Sugar. 2007. A glove-based gesture interface for wearable computing applications. In *4th International Forum on Applied Wearable Computing 2007*. VDE, 1–10. https://ieeexplore.ieee.org/abstract/document/5760491

[33] Kazuo Kiguchi and Yoshiaki Hayashi. 2012. An EMG-Based Control for an Upper-Limb Power-Assist Exoskeleton Robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 42, 4 (Aug 2012), 1064–1071. https://doi.org/10.1109/TSMCB.2012.2185843

[34] Jonghwa Kim, Stephan Mastnik, and Elisabeth André. 2008. EMG-Based Hand Gesture Recognition for Realtime Biosignal Interfacing. In *Proceedings of the 13th International Conference on Intelligent User Interfaces*. ACM, 30–39. https://doi.org/10.1145/1378773.1378778

[35] Przemyslaw A Lasota and Julie A Shah. 2019. Bayesian Estimator for Partial Trajectory Alignment. In *Robotics: Science and Systems (RSS)*. Freiburg im Breisgau, Germany. https://doi.org/10.15607/RSS.2019.XV.080

[36] T Lenzi, SMM De Rossi, N Vitiello, and MC Carrozza. 2011. Proportional EMG control for upper-limb powered exoskeletons. In *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 628–631. https://doi.org/10.1109/IEMBS.2011.6090139

[37] Yi Li. 2012. Hand Gesture Recognition Using Kinect. In *2012 IEEE International Conference on Computer Science and Automation Engineering*. IEEE, 196–199. https://doi.org/10.1109/ICSESS.2012.6269439

[38] Stuart Lloyd. 1982. Least Squares Quantization in PCM. *IEEE Transactions on Information Theory* 28, 2 (1982), 129–137. https://doi.org/10.1109/TIT.1982.1056489

[39] Natalia M López, Fernando di Sciascio, Carlos M Soria, and Max E Valentinuzzi. 2009. Robust EMG sensing system based on data fusion for myoelectric control of a robotic arm. *BioMedical Engineering OnLine* 8, 1 (2009), 5. https://doi.org/10.1186/1475-925X-8-5

[40] Kurt Manal and Thomas S Buchanan. 2003. A one-parameter neural activation to muscle activation model: estimating isometric joint moments from electromyograms. *Journal of Biomechanics* 36, 8 (2003), 1197–1202. https://doi.org/10.1016/S0021-9290(03)00152-0

[41] Jess McIntosh, Charlie McNeill, Mike Fraser, Frederic Kerber, Markus Löchtefeld, and Antonio Krüger. 2016. EMPress: Practical Hand Gesture Classification with Wrist-Mounted EMG and Pressure Sensing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*. ACM, 2332–2342. https://doi.org/10.1145/2858036.2858093

[42] Geoffrey McLachlan and David Peel. 2004. *Finite mixture models*. John Wiley & Sons.

[43] Samir Menon, Toki Migimatsu, and Oussama Khatib. 2016. A Parameterized Family of Anatomically Accurate Human Upper-Body Musculoskeletal Models for Dynamic Simulation & Control. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. 587–594. https://doi.org/10.1109/HUMANOIDS.2016.7803334

[44] Sushmita Mitra and Tinku Acharya. 2007. Gesture Recognition: A Survey. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 37, 3 (2007), 311–324. https://doi.org/10.1109/TSMCC.2007.893280

[45] Marcello Mulas, Michele Folgheraiter, and Giuseppina Gini. 2005. An EMG-Controlled Exoskeleton for Hand Rehabilitation. *9th International Conference on Rehabilitation Robotics (ICORR)* (2005), 371–374. https://doi.org/10.1109/ICORR.2005.1501122

[46] Luka Peternel, Nikos Tsagarakis, and Arash Ajoudani. 2017. A Human–Robot Co-Manipulation Approach Based on Human Sensorimotor Information. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 25, 7 (July 2017), 811–822. https://doi.org/10.1109/TNSRE.2017.2694553

[47] Anahita Qashqai, Hossein Ehsani, and Mostafa Rostami. 2015. A Hill-based EMG-driven model to estimate elbow torque during flexion and extention. In *2015 22nd Iranian Conference on Biomedical Engineering (ICBME)*. IEEE, 166–171. https://doi.org/10.1109/ICBME.2015.7404136

[48] Jinxian Qi, Guozhang Jiang, Gongfa Li, Ying Sun, and Bo Tao. 2019. Intelligent Human-Computer Interaction Based on Surface EMG Gesture Recognition. *IEEE Access* 7 (2019), 61378–61387. https://doi.org/10.1109/ACCESS.2019.2914728

[49] João Luiz AS Ramos and Marco A Meggiolaro. 2014. Use of Surface Electromyography for Human Amplification Using an Exoskeleton Driven by Artificial Pneumatic Muscles. In *5th IEEE RAS/EMBS International Conference on Biomedical Robotics and Biomechatronics*. IEEE, 585–590. https://doi.org/10.1109/BIOROB.2014.6913841

[50] Ali-Akbar Samadani and Dana Kulic. 2014. Hand Gesture Recognition Based on Surface Electromyography. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, 4196–4199. https://doi.org/10.1109/EMBC.2014.6944549

[51] 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*. ACM, 515–524. https://doi.org/10.1145/1357054.1357138

[52] Pradeep Shenoy, Kai J Miller, Beau Crawford, and Rajesh PN Rao. 2008. Online Electromyographic Control of a Robotic Prosthesis. *IEEE Transactions on Biomedical Engineering* 55, 3 (2008), 1128–1135. https://doi.org/10.1109/TBME.2007.909536

[53] Mingzhou Song and Hongbin Wang. 2005. Highly efficient incremental estimation of Gaussian mixture models for online data stream clustering. In *Intelligent Computing: Theory and Applications III*, Vol. 5803. International Society for Optics and Photonics, SPIE, 174–183. https://doi.org/10.1117/12.601724

[54] Yale Song, David Demirdjian, and Randall Davis. 2012. Continuous Body and Hand Gesture Recognition for Natural Human-Computer Interaction. *ACM Transactions on Interactive Intelligent Systems (TiiS)* 2, 1 (2012), 5. https://doi.org/10.1145/2133366.2133371

[55] Mark Tomaszewski. 2017. Myo SDK MATLAB MEX Wrapper. https://github.com/mark-toma/MyoMex

[56] Michael Van den Bergh and Luc Van Gool. 2011. Combining RGB and ToF Cameras for Real-Time 3D Hand Gesture Interaction. In *2011 IEEE Workshop on Applications of Computer Vision (WACV)*. IEEE, 66–72. https://doi.org/10.1109/WACV.2011.5711485

[57] Jonathan Weisz, Peter K Allen, Alexander G Barszap, and Sanjay S Joshi. 2017. Assistive grasping with an augmented reality user interface. *The International Journal of Robotics Research* 36, 5-7 (2017), 543–562. https://doi.org/10.1177/0278364917707024

[58] Ying Yin and Randall Davis. 2014. Real-Time Continuous Gesture Recognition for Natural Human-Computer Interaction. In *2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. IEEE, 113–120. https://doi.org/10.1109/VLHCC.2014.6883032

[59] Yue H Yin, Yuan J Fan, and Li D Xu. 2012. EMG and EPP-Integrated Human-Machine Interface Between the Paralyzed and Rehabilitation Exoskeleton. *IEEE Transactions on Information Technology in Biomedicine* 16, 4 (July 2012), 542–549. https://doi.org/10.1109/TITB.2011.2178034

[60] Felix E Zajac. 1989. Muscle and Tendon: Properties, Models, Scaling, and Application to Biomechanics and Motor Control. *Critical Reviews in Biomedical Engineering* 17, 4 (1989), 359–411. http://e.guigon.free.fr/rsc/article/Zajac89.pdf

[61] Xu Zhang, Xiang Chen, Yun Li, Vuokko Lantz, Kongqiao Wang, and Jihai Yang. 2011. A Framework for Hand Gesture Recognition Based on Accelerometer and EMG Sensors. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 41, 6 (2011), 1064–1076. https://doi.org/10.1109/TSMCA.2011.2116004

[62] Shengli Zhou, Qing Shan, Fei Fei, Wen J Li, Chung Ping Kwong, Patrick CK Wu, Bojun Meng, Christina KH Chan, and Jay YJ Liou. 2009. Gesture Recognition for Interactive Controllers Using MEMS Motion Sensors. In *2009 4th IEEE International Conference on Nano/Micro Engineered and Molecular Systems*. IEEE, 935–940. https://doi.org/10.1109/NEMS.2009.5068728

[63] Thomas G Zimmerman, Jaron Lanier, Chuck Blanchard, Steve Bryson, and Young Harvill. 1986. A Hand Gesture Interface Device. In *Proceedings of the SIGCHI/GI Conference on Human Factors in Computing Systems and Graphics Interface*. ACM, 189–192. https://doi.org/10.1145/29933.275628