Interpreting Volitional Movement Intent from Biological Signals

Henrique Dantas, *Member IEEE*, Taylor C. Hansen, *Student Member, IEEE*, David J. Warren, *Senior Member, IEEE*, and V John Mathews, *Fellow, IEEE*

Abstract

This paper reviews technologies and algorithms for decoding volitional movement intent using bioelectrical signals recorded from the human body. Such signals include electromyograms, electroencephalograms, electrocorticograms, intracortical recordings and electroneurograms. After reviewing signal features commonly used for interpreting movement intent, this paper describes traditional movement decoders based on Kalman filters and machine learning. A number of deficiencies of the current state of the art in this field are described, and three approaches that mitigate some of these deficiencies are reviewed. They include data aggregation-based training to improve decoder performance when only limited amounts of training data are available, a shared controller that incorporates estimates of movement goals, and an adaptive decoder designed to compensate for time-variations in the relationships between the human body and the prosthesis. Also included are experimental results that illustrate some of the concepts discussed in the paper.

I. Introduction

More than one million limb amputations occur worldwide each year. The loss of a limb profoundly changes one's ability to perform activities of daily living. However, individuals with limb deficiency often retain the underlying neural circuitry and much of the ability to control movements of their deficient limb. Those with upper-limb amputation lack an end effector (*i.e.*, a hand) to produce the desired movements that are still being transmitted along

H. Dantas was with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331 USA. He is now with Microsoft Research, Bellevue, WA, USA (e-mail: hedantas@microsoft.com).

V J. Mathews is with the School of Electrical Engineering and Computer Science, Oregon State University, Corvallis, OR 97331 USA. (e-mail: mathews@oregonstate.edu).

T. C. Hansen and D. J. Warren are with the Department of Biomedical Engineering, University of Utah, Salt Lake City, UT 84112 USA (e-mail: taylor.c.hansen@utah.edu; david.warren@utah.edu).

the peripheral nervous system. Neuroprostheses aim to restore function to the deficient limb by utilizing intact neural circuitry [1].

Prosthetic limbs have progressed from cable-driven systems with only a few degrees of freedom (DoFs) that are controlled by shoulder movements to highly dexterous systems that simultaneously control multiple DoFs with native biological signals [1]. Prostheses have also been sensorized to evoke meaningful artificial percepts for closed-loop control of their movements [2]. Naturalistic control of prostheses is facilitated by interpreting (decoding) movement intent from signals generated by the neuro-muscular system. This paper reviews modern approaches to movement intent decoding.

A. Physiology of Movement Generation

The nervous system consists of the central nervous system (CNS) composed of the brain and spinal cord, and the peripheral nervous system (PNS) composed of nerves and ganglia. The neuron is the basic unit of the nervous system and contains a cell body, dendrites, and an axon. Neurons are interconnected, with dendrites receiving information from upstream neurons and axons relaying information to downstream neurons, muscles, and organs. Neurons propagate information via electrochemical processes called action potentials, which, in mammals, last for approximately 1 ms and result in an approximately 100 mV change in the electrical potential across the neuron's cell membrane [3]. The PNS innervates all skeletal musculature involved in volitional movement. An action potential propagated by a motoneuron (a neuron of the motor system) will cause a twitch in a downstream muscle fiber. Musculoskeletal movements are evoked by multiple twitches resulting from action potentials transmitted from the CNS to the PNS motorneurons.

B. Natural Control Signals for Movement Decoding

Electrodes placed near neurons can record individual action potentials, and those located further away record *population signals* representing the aggregate activity of many neurons. Table I lists commonly-used neuromuscular signals for movement intent decoding [3].

In the CNS, population activity can be measured at the surface of the skull via electroencephalography (EEG). Although EEG is distorted by the skull and underlying tissue and requires complex signal enhancement algorithms, it has received renewed interest in recent

TABLE I
COMMONLY USED BIOELECTRICAL SIGNALS FOR MOVEMENT INTENT DECODING

Name	Signal Source	Amplitude	Frequency Range (Hz)
Electroencephalography (EEG)	Neuron Population Over the Cranium	5-300 μV	dc-150
Electrocorticography (ECoG)	Intracranial Neuron Population	10-5000 μV	dc-150
Intra-cortical Recordings	Single Neurons in Cortex	50–800 μV	100-20,000
Electroneurography (ENG)	Single Axons in Peripheral Nerves	20-800 μV	500-7,000
Electromyography (EMG)	Population of Muscle Fibers	0.1-5 mV	dc-500

years for brain-computer interfaces because it is noninvasive. Electrocorticography (ECoG), which measures population activity under the skull on the surface of cerebral cortex provides improved signal quality at the expense of invasiveness. The highest resolution is provided by intra-cortical recordings using electrodes that penetrate the cortical region of interest.

In the PNS, single unit spikes can be detected using penetrating electrodes and recorded via electroneurography (ENG). Motoneuron activity is amplified by muscle fibers and recorded via electromyography (EMG) with electrodes embedded in muscles or placed on skin. EMG signals are currently the most utilized for movement decoding and are a robust signal source for providing simultaneous, multi-DOF control [1], [4], [5].

C. Overview of the Paper

A typical pipeline for decoding movement intent from bioelectrical signals is shown in Fig. 1. Features derived from different bioelectrical signals and used as input to the decoders are reviewed in Section II along with signal enhancement techniques. Section III discusses early work in this field, the most commonly used decoder algorithms, and briefly discusses post-processing algorithms. Section IV describes a number of challenges that have prevented wider adoption of high-end, high-DoF prosthetic systems, and reviews three recent algorithmic approaches for mitigating some of these challenges. This section also contains illustrative experimental results obtained from amputee and non-amputee subjects. The results provided herein were acquired with human subjects under University of Utah IRB Protocols 55621 and 98851 with expiration dates of 27-Apr-2021 and 10-Feb-2022, respectively. The concluding remarks are provided in Section V.

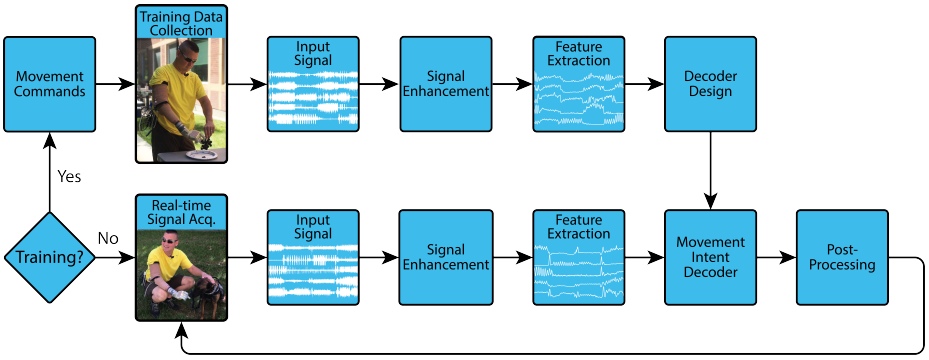

Fig. 1. A generic block diagram of the components of a movement intent decoder in a neuroprosthesis system. In general, the decoder must be trained before the prosthesis can be used for normal activities. The training process is usually supervised, where the prosthesis users are instructed to move their limb to follow specific trajectories, and the evoked bioelectrical signals are recorded along with the desired trajectories. These signals are processed to remove interference and other sources of noise. The decoders are trained using features extracted from the signals and the intended trajectories. The movement intent decoder designed in this manner is used to interpret the movements during normal activity. The control signal for the prosthesis is a processed version of the movement trajectory estimated by the decoder. The goal of the post processing system is to remove jitter and other estimation artifacts from the decoder output.

II. Signal Features used for Movement Interpretation

The success of biological signals as sources for movement intent decoders is dependent on the extraction of signal features that contain information about the intended movement. In general, the decoder algorithms work in similar ways once the features are computed. Although most of the experimental results presented herein use EMG-derived features, nearly any other feature source could have been employed in the illustrative examples.

A. Signal Features

Electroencephalogram: EEG is often split into features based on frequency bands including the delta (<4 Hz), theta (4-7 Hz), alpha (8-15 Hz), and beta (>15 Hz) bands. For movement intent decoders, the most useful of these is the alpha band, where signal power decreases during movement execution [6]. Coefficients of an autoregressive model that fits the signal in the alpha band have also found use as features for movement intent decoders [6].

Electrocorticogram: Two features of ECoGs are most used for movement decoding [6]. The first is low-frequency amplitude modulations which are generally correlated with the direction of limb movements, but not to the degree seen with intra-cortical recordings. The second major feature is changes in signal energy above 75 Hz, contrasting with the low-frequency features useful in EEG decoding. Recording beneath the skull for ECoG lessens signal distortion, thus facilitating the value of these high-frequency features.

Intraneural Recordings: Intraneural recordings in the CNS and PNS (e.g. intra-cortical recording and ENG, respectively) have analogous features to one another based on action potential spike detection. Detection of these spikes is generally accomplished with a voltage threshold. Once detected, spikes can be sorted in order to infer single-unit activity, under the presumption that each unit provides unique information about the intended movement [6]. However, the potential value of single-unit activity versus the computational cost of extracting single-units remains an open question. The resulting features include firing rate and power and phase information in discrete frequency bands [1]. Besides their application in movement intent decoding, such techniques have also been explored to restore communication for those with assistive needs beyond motor control [7].

Electromyogram: More than three dozen EMG features have been described in the literature for movement intent decoding [8]. Redundancies exist among them, and only the most commonly-used features are discussed here. The most common time-domain features used as input to movement decoders include mean absolute value (MAV), zero crossing rate, rate of slope sign changes, and waveform length [9]. Time-frequency domain features as well as signal energy in different frequency bands have also found use as EMG features [10].

B. Signal and Feature Enhancement

Biological signal quality is greatly influenced by circuit design and choices regarding amplifier architecture, preprocessing algorithms, and sampling rate, among others. These nuances are beyond the scope of this paper, and we recommend Northrop [11] as a good source of information on such issues. Regardless of the design choice, the resulting signals for movement intent decoding can be distorted due to various factors. Noise and redundant information between recording sites motivate the need for signal enhancement in tandem with feature extraction. Interference from external sources such as line voltage are reduced using differential amplification and appropriate filtering. Artifacts in the sensor signal caused by body movements and other activities of the human body such as breathing may affect all the sensors in the same way. The common average reference (CAR) method, in which a virtual reference obtained by averaging the sensor signals across all electrodes is subtracted from each sensor signal, is useful to remove such artifacts [1]. Further, redundant feature

information is typically managed with well-known dimensionality reduction techniques such as principal component analysis and linear discriminant analysis. These methods are also useful in mitigating the effects of broadband noise in the signals [4]. Finally, the best choices of signal features may be different for decoding intended position, intended velocity, or intended force [12], [13]

III. A Review of Classical Decoder Algorithms

A variety of algorithms have been employed to infer motor intent from bioelectrical signals, with most studies initially being performed in animal models followed by first-in-human demonstrations. Here, we provide an overview of the research in this field, and summarize the most commonly encountered *mature* algorithms. Generally, the algorithms can be separated into continuous controllers that can estimate the movement intent continuously, and classifier-based controllers that estimate a specific movement goal over a finite set of possibilities. To achieve the same goal, say a pencil grip, the continuous controller must individually and simultaneously control the thumb, index, and middle fingers to the correct positions to hold a pencil whereas the classifier only must decide that the intent for a pencil grip is more probable than any other movement in the set. Although classification approaches potentially reduce the cognitive load needed to perform a movement, their value to the user is greatly reduced due to their inability to go beyond the limited set of movements they were trained for. Consequently, this paper focusses primarily on continuous controllers. For a more comprehensive list of algorithms, see [1], [13].

A. Early Work

Perhaps the earliest demonstration of the feasibility of neural decoding of motor intent resulted from Fetz's research in non-human primates in the late 1960s and 1970s [14]. Fetz demonstrated that animals could be trained to modulate the activity of individual neurons in motor cortices in response to a behavioral task and that these neurons could be used to control an external device. Their decoder was a simple linear classifier operating on the firing rates of multiple neurons recorded using an electrode implanted in the animal's brain.

Georgopoulos showed in the 1980s and 1990s [15] that some neurons in motor cortices increase their activity (firing rate) when the intended movement is in a particular direction

(the neuron's preferred direction) and decrease their activity when moving in the opposite direction. On average, the relationship (tuning curve) between the movement direction and the neuron's activity formed a cosine shape with the maximum activity occurring at the preferred direction and the minimum occurring when moving in the opposite direction. Further, they found that they could reliably estimate the direction of the intended movement with a *population vector*, defined for a set of monitored neurons that exhibited a wide range of preferred directions. The population vector is the sum of the measured activity vectors associated with all the neurons in the set. A neuron's activity vector is oriented along its preferred direction and has a magnitude equal to its normalized firing rate. Kennedy [16] reported in 2000 that a human chronically implanted with a cone electrode could use a population vector to move a two-dimensional computer cursor.

In the late 1990s, both the electrode technology and the associated recording technology advanced to allow simultaneous recording of hundreds of neurons. This made it possible to monitor correlated neuronal activity, which led to the use of a number of linear regression and multivariate linear regression approaches for decoding. These approaches have a very simple formulation of $\hat{\mathbf{x}}_k = \boldsymbol{\beta}\mathbf{Z}_k$, where $\hat{\mathbf{x}}_k$ is the estimated kinematic state vector of the limb (joint angle positions, velocities, etc.) at time k, $\mathbf{Z}_k = [\mathbf{z}_k, \mathbf{z}_{k-1}, \cdots, \mathbf{z}_{k-m}]$ is a matrix containing the most recent m feature data, m represents the memory in the system, and $\boldsymbol{\beta}$ is a matrix of regression parameters found by training. These methods were inherently non-recursive and often resulted in jittery estimates. Typically, linear filters with long memory (up to 1 second) were used to smooth the estimates but this led to lags in the evoked movement.

B. Kalman Filter-Based Movement Decoders

Lack of smoothness (*i.e.*, jitter) is a major drawback of the non-recursive regression algorithms discussed above. Recursive algorithms can mitigate this problem to some extent, and the Kalman filter (KF), has become popular in neuroprosthesis research. The classic linear discrete-time Kalman filter [12] is performed by the recursive operations shown in Fig. 2. It assumes a linear generative model that describes the relationship between the movement intent vector (states in the KF, $\mathbf{x}$) and the features (observations in the KF, $\mathbf{z}$). In each iteration, the state and observation vectors are first predicted through the assumed model and then

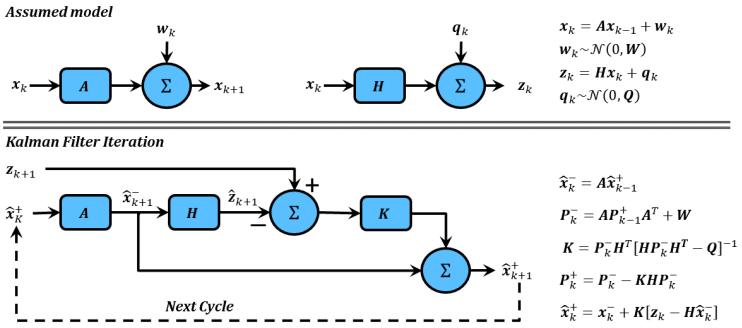

Fig. 2. Assumptions and simplified block diagram of the Kalman filter. The Kalman filter assumes a discrete time linear model of the plant, and that both the states of the plant (intent kinematic variables) and the observations of the plant (features) have zero-mean, normally distributed additive noise. For each iteration, the Kalman filter predicts the new state and observation vector, and then adjusts the state vector based upon the difference between the predicted observations and the measured observations. This adjustment is done by use of the Kalman Gain matrix, K, that is calculated to reduce the state estimation error, in a least square sense.

the error between the actual and predicted observations are used to adjust the predicted state vector via the Kalman gain matrix (**K**). The value of the Kalman gain matrix is set in each iteration to minimize the residual error in the estimate of the state vector.

Many variations of the Kalman decoder are available. The *steady-state Kalman decoder* uses a fixed Kalman gain matrix and avoids the need to perform matrix inversions in real time [17]. Assuming a stationary operating environment, this fixed gain matrix is computed during training by iterating the equations for the Kalman gain until they converge. Alternative formulations include adding latent variables that affect the performance of the decoder (*e.g.*, attentiveness to the task) or movement variables such as the end goal position to the KF's plant model [18]. Adaptation of the plant model to address the nonstationarity of the neuronal data has been explored [19]. Methods that incorporate nonlinear models of the plant in the KF have also been studied, including the *extended Kalman filter* that linearizes the plant during each iteration, and the *unscented Kalman filter* and *particle filter*-based methods that characterize the effects of the nonlinearities using sampling approaches.

C. Machine Learning-Based Decoding Methods

Concurrent with the ability to simultaneously record from large numbers of electrodes and the initial investigation of regression-based decoders, researchers also investigated how to approximate the relationship between kinematic states and the feature states using nonlinear functions. It is common to model the next kinematic states as a function of the current and past kinematic states and the current and past feature vector as $\hat{\mathbf{x}}_{k+1} = f_\theta(\mathbf{Z}_k, \mathbf{x}_k)$, where

f is a nonlinear function fully described by the parameters θ. Artificial neural networks (ANNs) have been used to learn this relationship. Perhaps the most common example is that of the multilayer perceptron (MLP) network. In almost all implementations of ANN-based movement intent decoders, the parameters of the network are determined by training using a machine learning (ML) algorithm, on a large data set containing features and movement trajectories that represent activities the prosthesis will normally encounter in everyday use. The ML algorithms are designed almost always to minimize some non-negative cost function, *e.g.*, the least-squares error (used with some form of regularization to reduce over-fitting), between the desired movement trajectory and the estimated trajectory.

Because of the limited availability of real time hardware at the time, early research in ML-based decoders involved shallow networks, with only one hidden layer, and only a small number of nodes (5 to 20) in the hidden layer. Interest in more sophisticated ML approaches for decoding has renewed in recent years. A number of modern approaches for decoding movement intent have been investigated, such as kernel learning [20], recurrent neural networks, deep learning, and reinforcement learning [9]. Although ANNs can be implemented as either continuous controllers or classifiers of movement intent, many researchers in the field have used ANNs as classifiers that identify which signals patterns are responsible for certain movements [21], [22].

D. Post-Processing Methods

In general, the decoder outputs contain jitter that adversely affect the quality and comfort of use, as well as increase the cognitive load necessary to control a prosthesis. Various *ad hoc* post-processing steps on the decoder's outputs have been proposed to reduce the effects of jitter. A common jitter-reduction approach is to set movement estimates less than a threshold to zero. Another approach is to apply smoothing methods such as a lowpass filter to the decoder output, but lowpass filters can exhibit undesirable lags in performing movements. Nonlinear smoothing filters that can smooth the data without introducing unacceptable delays are better alternatives than linear lowpass filters for jitter reduction. It is best to consider the inclusion of post-processing methods in the overall decoder design, instead of using them as methods for addressing flaws of the decoder.

IV. Recent Advances in Movement Decoding

A. Improving Learning from Limited Amount of Training Data

Although artificial neural network movement decoders employ more complex models than those used for Kalman decoders, many researchers have informally reported that machine learning-based systems do no better than the KF, particularly when performing multi-DoF movements [23]. This observation may be a result of training ANN decoders with incomplete data sets that do not contain all possible state transitions, resulting in the machine learning algorithms over-fitting the limited amount of training data available. As a result, the ANN decoders perform poorly outside the regions for which it was trained, because it has no assumed model for these regions, whereas the KF decoders perform better as they fill this space with approximations based upon generative models. Although acquiring additional training data is an obvious solution, doing such with human subjects, particularly those with reduced stamina due to a disability such as amputation or paralysis, is not always practical.

A number of solutions to address insufficient training data have been proposed in the machine learning community. Many of these create new training samples from the original data set (for example, the SMOTE algorithm [24]). Regret-based reinforcement learning algorithms [25] are capable of fully modeling the dynamics of the underlying systems, but require very large amounts of training data. Non-regret-based imitation learning algorithms [26], on the other hand, often provide adequate performance with much less actual training data.

Here we describe, as an illustrative example, a method known as data set aggregation (DAgger) [26], a non-regret algorithm, and demonstrate its ability to address limited training data using experimental data. DAgger (Fig. 3) is an iterative algorithm in which the training data are augmented during each iteration using newly decoded data from the previous iteration. Let $\mathbf{Z}$ represent the features extracted from the training data, and let $\mathbf{X}$ be the set of instructed movements that generated the features $\mathbf{Z}$. In the first iteration of DAgger, the decoder is trained with a conventional training algorithm using movement data ($\mathbf{X}_1 = \mathbf{X}$) and feature data ($\mathbf{Z}_1 = \mathbf{Z}$). The decoder then computes an estimate of the intended movements ($\hat{\mathbf{X}}_1$) from the feature set ($\mathbf{Z}_1$). The estimated movement data along with a randomly perturbed feature set are added to the existing training data set to create an augmented training set for the next iteration. This set for the rth iteration is given by $\left\{\mathbf{Z}_r;\mathbf{X}_r\right\} = \left\{\mathbf{Z}_{r-1}\bigcup\left(\mathbf{Z}_1+\Gamma_r\right);\mathbf{X}_{r-1}\bigcup\hat{\mathbf{X}}_{r-1}\right\}$,

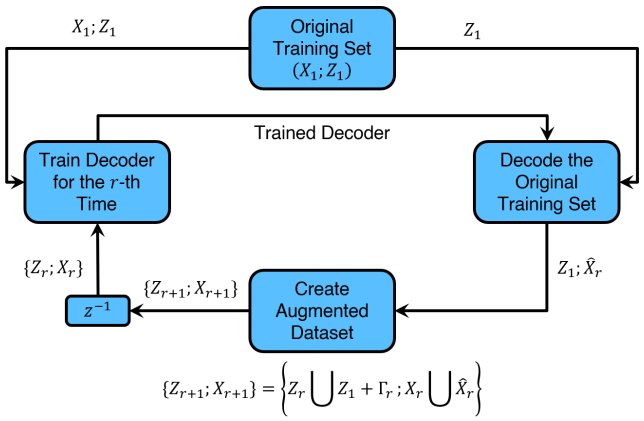

Fig. 3. In DAgger, the decoder is trained initially using the original feature data and the intended movements. Then the trained decoder is used to decode trajectories in the original training set. The predicted trajectories with a randomly perturbed version of the feature data are used to expand the training data. The decoder is retrained with the augmented dataset. This algorithm runs until a convergence criteria is met.

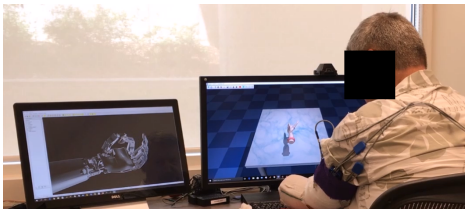

Fig. 4. Experimental setup used to obtain the results presented herein. The movement intent decoders were trained and tested in a virtual reality environment (VRE). During training, subjects were instructed to mimic the movement of a hand displayed in the VRE (left screen) with their phantom limb while EMG and/or PNS signals were recorded. The signal features calculated from the recordings were used to train the decoder. Once trained, the algorithm and its parameters were tested by having the volunteer again try to follow instructed movement presented in the VRE but in the testing phase, the volunteer additionally observed the decoded hand position in real time in the VRE (right screen). Typically, each subject participated in multiple sessions spanning several months, and the results presented are averages over a large number of trials conducted in this manner.

where Γ_r is a random perturbation added to the feature set to avoid over-fitting. The iteration is repeated until reaching a user-selected end point. DAgger effectively creates a richer training set during each iteration because the augmented data set contains a larger number of samples of the kinematic states than there were during the previous iteration.

The experimental method used to acquire training and testing data is summarized in Fig. 4 (see [9] for more details and results). The results presented here were from two volunteers with a transradial amputation, with feature data arising from 32 chronically implanted EMG electrodes and the instructed (training) or decoded (testing) movements of the volunteers' phantom limb displayed in virtual reality. The performance metric used was the normalized mean-square error (NMSE) between the estimated and instructed movements.

The results of testing both a MultiLayer Perceptron (MLP) network trained with different

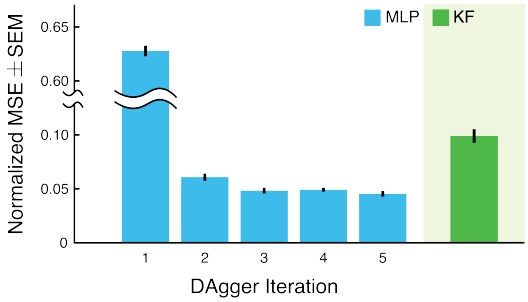

Fig. 5. The performance of an MLP network-based decoder was improved by the DAgger algorithm and reached marginal improvement after 3 iterations. See [9] for implementation details of both systems. The bars represent the mean normalized mean-square-error (MSE) and the error bars are the standard error of the mean (SEM).

numbers of DAgger iterations and a trained Kalman filter (KF) are illustrated in Fig. 5. DAgger enabled a more than 12-fold reduction in NMSE for the MLP decoder from the first iteration to the third, and quickly converged to this increased performance level in three to five iterations in this experiment. Interestingly, the KF performs substantially better than the MLP network after one iteration of DAgger (*i.e.*, with just the original training data), but the MLP network trained with multiple DAgger iterations performed better than the KF. We postulate that this poor initial performance for the MLP is due to the aforementioned inadequate training data, but it was remedied by the DAgger algorithm.

B. Interpreting Movement Goals

Nearly all the methods discussed thus far estimate the position or velocity of various DoFs of the prosthesis at each time. In general, the sensor signals used for these estimates only contain information about intended movements at the current time, and little information about the goal of the overall movement. If the prosthesis controller had information about the end point locations of the various DoFs, the controller could design the trajectory of movement of each DoF. This would reduce the cognitive effort required to perform movements as the user would only have to attend to errors in the movement, instead of having to create the entire movement. We discuss two approaches to interpreting movement goals here, one using native biological sensors and the other using external sensors.

The posterior parietal cortex (PPC) plays a critical role in planning movements. Neural signals recorded using electrode arrays implanted in the PPC can be used to infer movement goals and end points of movement trajectories. Mulliken *et al.* [18] implanted electrode arrays in the PPC of two rhesus monkeys and showed that a Kalman filter could accurately estimate

the goal as well as the trajectory of movement when the animals were controlling a joystick. Nevertheless, accessing PPC requires complex and invasive neurosurgery, and may not be appropriate for all limb deficient individuals.

A second approach to interpret movement goals is to use non-biological data acquired from external sensors such as cameras and proximity sensors to obtain a more complete understanding of the environment (scene) in which the prosthesis is operating. Movement goals may be estimated from the scene and knowledge of the motion of the prosthesis. Hotson *et al.* [27] added the movement's goal to the states of a Kalman filter, and continuously estimated the movement goal and the DoF positions from biological and external sensor signals. Their experimental results demonstrated substantial improvement in the ability of the prosthesis to track the desired movements with the inclusion of the goal estimation.

We also have seen improvement in decoding performance by sharing control between multiple decoders. In one formulation involving goal estimation [28], we find f_s, the overall controller output as $f_s = \beta f_g + (1-\beta) f_b$, where f_g is the controller output based on the goal estimate and f_b is the decoder output based on the biological signals, and $0 \leq \beta \leq 1$. Intuitively, we choose the mixing parameter β close to 1 or 0 based on whether our confidence in the goal estimation process is high or low, respectively. A formal derivation of this controller using a Markov decision process formulation is available in [28].

We performed experiments with three intact-arm subjects where the goal was to quickly reach a target location, and maintain a position within ± 0.1 of the target location for the duration of the trial. (All movement data were normalized to the ± 1 range.) The duration of time while in the target region for the 7-s trial was used as the performance metric. We used a KF as the movement intent decoder and used features derived from up to 32 channels of surface EMG signals. We simulated the goal estimate in each trial by perturbing the known end position of the virtual hand with an additive, zero-mean, and white Gaussian noise. During each trial, the subject knew the true goal, and could correct for any observed errors, but the goal estimator only knew the noisy goal. The magnitude of noise was such that 48% of the trials would have a noise-perturbed goal outside of the acceptable target region. In practice, the goal would be estimated from one or more sensors on or near the prosthesis.

The estimated thumb and index finger movements in two trials with one of the subjects

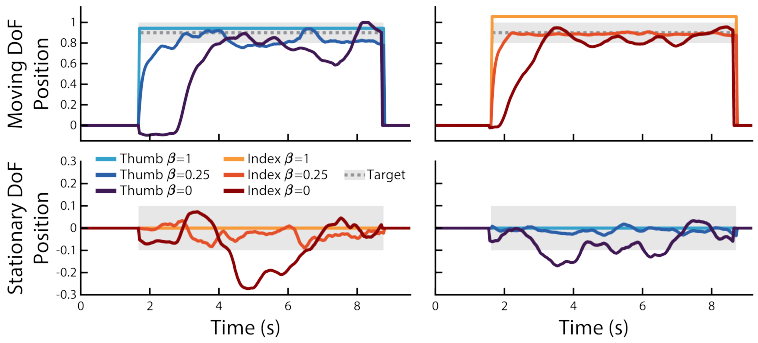

Fig. 6. Sharing control between a movement intent decoder and an estimated goal improves task performance. Shown are representative examples of the kinematic output of three different mixing parameters with an intact-arm subject. The top row depicts movements of the thumb (left panel) and index finger (right panel) when the subject was instructed to move them in succession to a specified target (target shown as dashed line and acceptable target region shown as gray rectangles). The bottom row depicts movements of the corresponding digit that was instructed to remain still. With high machine input (β=1), there is no jitter, but large errors in the goal estimate can drive the prosthesis outside the target region. No machine input (β=0) markedly increases jitter in both moving and stationary DoFs. When sharing control (β=0.25), the user can correct for errors introduced by noise in the goal and achieve better performance.

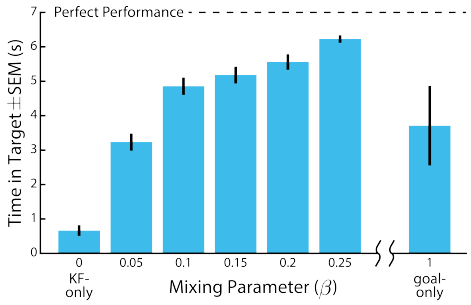

Fig. 7. Shared control improved the time in the target region for intact-arm subjects compared to the KF-only and goal-only cases. Bar height represents the aggregated mean of the performance across all datasets and the error bars represent the standard error of the mean (SEM). Using the combination of the goal and the KF-based decoder the subject was able to stay in the target longer than the two component decoders.

are shown in Fig. 6. Three cases are shown in each panel: (1) the KF-only estimate ($\beta = 0$); (2) the shared estimate of the two with $\beta = 0.25$; and (3) the goal-only estimate ($\beta = 1$). The panels on the left corresponded to the command to move the thumb (upper panel) while keeping all other fingers still (including the index finger shown in the lower panel). The panels on the right show similar cases but for movement of the index finger. For clarity of presentation, only the estimates for the thumb and index finger are shown in the figure. As can be seen in the upper panels, the KF-based decoder is relatively slow to respond, and exhibits large amounts of jitter after reaching the target zone (gray region). The error in the KF estimate is sufficient to frequently take the digits out of their target zones. The goal-based estimate showed a quick transition to a steady-state value of the noise-perturbed goal, but

this position may be outside the target zone (e.g., for the index finger in the upper right panel). The shared controller gains the advantages of each of its parts. Like the goal-based estimate, it rapidly jumps to a steady state value, but, like the KF, it allows the user to correct for any observed errors. The bottom panels displays the movements of the fingers instructed to remain still at the rest position. These results demonstrate that the shared controller exhibits lower cross talk (movement of the digits instructed to be still) than the KF alone.

Summaries of the performance averaged across all trials and the three subjects are shown in Fig. 7. These results show that even a small value of β substantially improves the performance of the KF, and that the shared controller outperforms the goal estimator for many values of β. Of the values of β tested, $\beta = 0.25$ resulted in the best performance, with the subjects able to stay in the target zone for approximately 6 s. on average out of a maximum possible 7 s.; however, the best choice of β is dependent on the quality of the goal estimates. If the goal estimates were worse, lower values of β might be preferable, and vice versa. Additional work is still needed to replace the synthetic goal estimator with a true goal estimator and evaluate shared controllers under more realistic use environments.

C. Improving Long-Term Performance Through Adaptive Decoding

In current implementations, the decoders are trained prior to deploying them, and their parameters are kept frozen during normal operation of prostheses. The performance of such systems tend to degrade over time, for reasons such as movement of the electrodes, changes in the muscle-electrode interactions, and physiological changes such as those due to fatigue and aging. Therefore, it is desirable to adapt the decoders in response to such changes.

The simplest and most common approach to tackle performance degradations over time is to periodically retrain the decoder using supervised learning. Similar to the acquisition of the initial training data, the user is directed to perform a set of pre-determined movements, and the sensor data is recorded simultaneously with the movements. The new training data is used alone or in combination with the earlier training data to retrain the system. However, frequent retraining is inconvenient for the user, and it would be ideal to design an online learning strategy that is continuously performed. Furthermore, since the limb movements during normal activities are not pre-determined, unsupervised learning algorithms are needed.

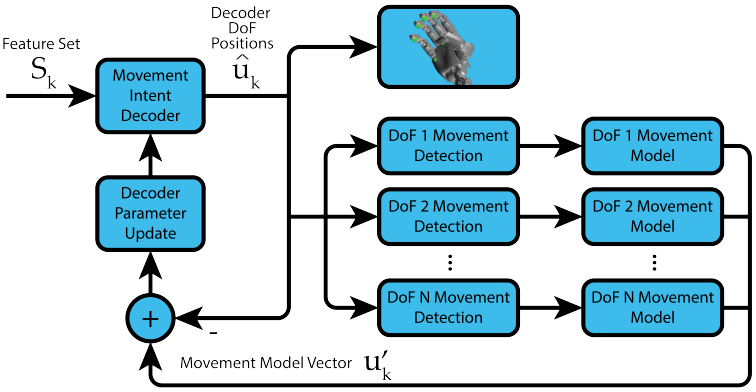

Fig. 8. Conceptual block diagram of an adaptive algorithm for movement intent decoders. At each time step, the decoder predicts the next position of the limb, and determines if a movement of one or more DoFs occurred in a small window prior to the current time. If a movement pattern is found for a DoF, the decoder output for that DoF is fit with a movement model. The difference between the decoder output and the movement model is used to update the parameters of the decoder.

Tadipatri *et al.* [29] recently developed an adaptive decoder for animal experiments involving 2-D center-out tasks. The animal was trained to use a manipulandum to move a computer cursor from a center location on a monitor to one of 8 equidistant locations. In this experiment, the straight line between the center location and the desired end point was used as the desired trajectory to develop an adaptive system. Although the straight-line model is not realistic for the movements of the digits and wrists, the concept described above may be generalized to include different movement models as shown in Fig. 8.

In this approach, we assume that the movement trajectory for each DoF follows a specific shape with unknown parameters. The system detects movements of each DoF separately and finds the parameters that best describe the movement trajectory from the decoded position estimate, including any adjustments provided by the user attending to the movement. Once the model trajectory parameters are computed, the system uses the model as the supervisory signal (*i. e.*, the desired trajectory), and updates the parameters of the decoder in an effort to bring the output of the decoder closer to the modeled trajectory. An implementation of this concept is the adaptive movement intent decoder reported in [30], where movements of the digits of a hand was modeled using five piece-wise linear segments as shown in Fig. 9 for full flexion of a finger. The segments are a resting phase, a rising phase corresponding to the times when the digit is moving from rest to a desired position, a hold phase, a falling phase when the digit is moving back to the rest position, and finally another resting phase.

The algorithm continuously processes the movement decoder output to first detect move-

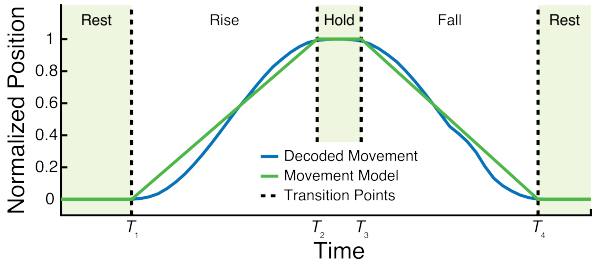

Fig. 9. Sample movement of a full flexion. Desired movement based on the movement model, in green, and the decoded position of a DoF, in blue. This movement model is composed of five sections: (1) A resting phase where the DoF is stationary. (2) A rising phase, where the DoF is moving to its target position (time is T_1 to T_2). (3) A holding phase, where the DoF stationary in the target position (time is T_2 to T_3). (4) The falling phase, where the DoF is moving back to the resting zone (time is T_3 to T_4). (5) The final resting phase.

ments by each DoF by searching for patterns that are similar to the movement model. (Movements that do not follow the movement model(s) are not used to adapt the decoder parameters.) Each detected movement is modeled by determining the transition points between the piece-wise linear segments, the slopes of the rising and falling phases, and the positions of the rest and hold phases. The decoder parameters for each DoF may be updated every time a movement of that DoF is detected and modeled, or at specific instances of time. The algorithm of [30] used a simple gradient update algorithm to reduce the mean-square difference between the movement model and the decoder output. A comparison of the long-term (150 day) performance of a fixed-parameter decoder, and the adaptive decoder was presented in [30]. The results demonstrated that both the adaptive and non-adaptive versions showed performance degradation with time, but the mean-square error performance of the adaptive decoders were approximately 25% better than that for the non-adaptive decoder, demonstrating the promise of this approach. Nevertheless, our ongoing research suggests that it is possible to substantively improve the performance of the adaptive decoder.

V. Concluding Remarks

A limb prosthesis can profoundly improve the quality of life of people with limb deficiencies, even if the prosthesis is a poor approximation of a native limb. However, limited and unreliable movement decoding has contributed to high levels of prosthesis abandonment. In this paper, we reviewed the state of the art in movement intent decoders for application in prosthetic systems. In the authors' opinion, the following suggests areas where advances in signal processing approaches could improve the functionality of prosthetic devices.

For the most part, the decoders are trained with indirect involvement of the user. That is, users are requested to attempt to move their phantom limb to follow the movements presented to them. Since the decoder has not been designed prior to training, the users are not provided any feedback (for example, visual) on the actual movements of the prosthesis at this stage. The acquired features are then related to the instructed movements to design the decoder. Methods that bring the users into the loop by providing feedback on prosthesis performance during training potentially could result in better decoder designs. There have been a few efforts in this area but, to date, most approaches have been *ad hoc*. To make decoder training with the users in the loop more effective, the problem should be formally developed and investigated using established signal processing methods.

Although out of the scope of signal processing improvements to decoding motor intent, substantive improvements in functionality of motor control can be had by providing sensory feedback to users, particularly percepts similar to those felt prior to the limb disability. Much of the dexterity of a hand is due to sensory percepts of the limb's position in space (proprioception) and how the hand is interacting with objects (tactile sensations). Sensory restoration for those with limb disabilities is an active area of research. Many researchers are investigating peripheral nerve and cerebral cortex implants that provide mechanisms to evoke sensory neural activity. Other researchers are investigating signal processing methods to provide optimal information transfer of the complex spatiotemporal patterns of neural activity that occurs during object interaction.

In Section IV-B we discussed sharing control between moment-to-moment and goal estimating decoders, and in the previous paragraph we discussed providing sensory feedback. For these ideas to become of use to the prosthesis community, environmentally aware and responsive prostheses need to be developed. Prosthetic vendors are experimenting in providing sensors that sense contact pressure and cameras to enable object recognition. There is still much work to be done in using sensor signal processing to identify and use the “best" environmental measurements to enhance the decoding of movement intent.

The ultimate goal of research in this field is the development of prostheses that act and feel like natural limbs prior to amputation or paralysis. Recent successes and ongoing research in the field suggest that the achievement of this goal is imminent.

ACKNOWLEDGMENT

This work was supported by National Science Foundation Grants 1533649, 1901492 and 1901236 and by DARPA's HAPTIX program (Contract No. N66001-15-C-4017). We thank NVIDIA Corporation for the donation of a Tesla K40 GPU used in this research, and our volunteer subjects who provided the experimental data described in this paper. DJW has licensed intellectual property that was utilized in the research described herein and he may receive financial gain from its use. He has an approved plan by the University of Utah's Individual Conflict of Interest Committee for managing potential conflicts.

REFERENCES

[1] D. J. Warren *et al.*, "Recording and decoding for neural prostheses," *Proc. IEEE*, vol. 104, no. 2, pp. 374–391, Feb. 2016.

[2] J. Burck *et al.*, "Revolutionizing prosthetics: Systems engineering challenges and opportunities," *Johns Hopkins APL Technical Digest*, vol. 30, pp. 186–197, Jan. 2011.

[3] E. R. Kandel *et al.*, *Principles of Neural Science*, 4th ed. McGraw-Hill Medical, Jul. 2000.

[4] K.-S. Hong *et al.*, "Motor-commands decoding using peripheral nerve signals: a review," *Journal of Neural Engineering*, vol. 15, no. 3, p. 031004, Mar. 2018.

[5] M. Sartori *et al.*, "Robust simultaneous myoelectric control of multiple degrees of freedom in wrist-hand prostheses by real-time neuromusculoskeletal modeling," *Journal of Neural Engineering*, vol. 15, no. 6, p. 066026, 2018.

[6] W.-K. Tam *et al.*, "Human motor decoding from neural signals: a review," *BMC Biomedical Engineering*, vol. 1, no. 1, Dec. 2019.

[7] C. Pandarinath *et al.*, "High performance communication by people with paralysis using an intracortical brain-computer interface," *eLife*, vol. 6, p. e18554, feb 2017. [Online]. Available: https://doi.org/10.7554/eLife.18554

[8] A. N. Phinyomark *et al.*, "Feature reduction and selection for EMG signal classification," *Expert Systems with Applications*, vol. 39, no. 8, pp. 7420 – 7431, Jun. 2012.

[9] H. Dantas *et al.*, "Deep learning movement intent decoders trained with dataset aggregation for prosthetic limb control," *IEEE Transactions on Biomedical Engineering*, vol. 66, no. 11, pp. 3192–3203, Jan. 2019.

[10] K. Englehart and B. Hudgins, "A robust, real-time control scheme for multifunction myoelectric control," *IEEE Trans. on Biomedical Engineering*, vol. 50, no. 7, pp. 848–854, Jul. 2003.

[11] R. Northrop, *Analysis and Application of Analog Electronic Circuits to Biomedical Instrumentation (Biomedical Engineering)*, 2nd ed. CRC Press, Mar. 2012.

[12] W. Wu *et al.*, "Bayesian population decoding of motor cortical activity using a Kalman filter," *Neural Computation*, vol. 18, no. 1, pp. 80–118, Mar. 2006.

[13] A. Bashashati *et al.*, "A survey of signal processing algorithms in brain–computer interfaces based on electrical brain signals," *Journal of Neural Engineering*, vol. 4, no. 2, pp. R32–R57, Mar. 2007.

[14] E. E. Fetz and D. V. Finocchio, "Operant conditioning of specific patterns of neural and muscular activity," *Science*, vol. 174, no. 4007, pp. 431–5, Oct. 1971.

[15] A. P. Georgopoulos, A. B. Schwartz, and R. E. Kettner, "Neuronal population coding of movement direction," *Science*, vol. 233, no. 4771, pp. 1416–9, Sep. 1986.

[16] P. R. Kennedy *et al.*, "Direct control of a computer from the human central nervous system," *IEEE Trans. Rehabilitation Engineering*, vol. 8, pp. 198–202, Jun. 2000.

[17] W. Q. Malik *et al.*, "Efficient decoding with steady-state Kalman filter in neural interface systems," *IEEE Trans. Neural Syst. Rehabil. Eng*, vol. 19, no. 1, pp. 25–34, Feb. 2011.

[18] G. H. Mulliken *et al.*, "Decoding trajectories from posterior parietal cortex ensembles," *The Journal of Neuroscience*, vol. 28, no. 48, pp. 12 913–12 926, Nov. 2008.

[19] G. J. Gage *et al.*, "Naive coadaptive cortical control," *J Neural Eng*, vol. 2, no. 2, pp. 52–63, Jun. 2005.

[20] A. Paiva *et al.*, "A reproducing kernel Hilbert space framework for spike train signal processing," *Neural Comput.*, vol. 21, no. 2, pp. 424–449, Feb. 2009.

[21] H. Dantas, V. J. Mathews, D. J. Warren, and T. Hansen, "Shared prosthetic control based on multiple movement intent decoders," *IEEE Transactions on Biomedical Engineering*, pp. 1–1, 2020.

[22] G. W. Favieiro *et al.*, "Decoding arm movements by myoeletric signals and artificial neural networks," in *ISSNIP Biosignals and Biorobotics Conference*, Vitoria, Brazil, Jan. 2011, pp. 1–6.

[23] M. D. Paskett *et al.*, "Activities of daily living with bionic arm improved by combination training and latching filter in prosthesis control comparison," *medRxiv*, 2020. [Online]. Available: https://www.medrxiv.org/content/early/2020/12/23/2020.10.21.20217026

[24] L. Torgo *et al.*, "SMOTE for regression," in *Progress in Artificial Intelligence*, L. Correia, L. P. Reis, and J. Cascalho, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, Sep. 2013, pp. 378–389.

[25] S. Levine *et al.*, "Learning contact-rich manipulation skills with guided policy search," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, Seattle, WA, May 2015, pp. 156–163.

[26] S. Ross *et al.*, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the Fourteenth Int. Conf. on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, vol. 15, Fort Lauderdale, FL, USA, Apr. 2011, pp. 627–635.

[27] G. Hotson *et al.*, "High precision neural decoding of complex movement trajectories using recursive bayesian estimation with dynamic movement primitives," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 676–683, Jul. 2016.

[28] H. Dantas *et al.*, "Shared human-machine control for self-aware prostheses," in *IEEE Inter. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Calgary, Canada, Apr. 2018, pp. 6593–6597.

[29] V. A. Tadipatri *et al.*, "Overcoming long-term variability in local field potentials using an adaptive decoder," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 2, pp. 319–328, Feb. 2017.

[30] H. Dantas *et al.*, "Semi-supervised adaptive learning for decoding movement intent from electromyograms," in *European Signal Processing Conference (EUSIPCO)*, Madrid, Spain, Sep. 2019, pp. 1–5.