Online Human Training of a Myoelectric Prosthesis Controller via Actor-Critic Reinforcement Learning

Patrick M. Pilarski, Michael R. Dawson, Thomas Degris, Farbod Fahimi, Jason P. Carey, and Richard S. Sutton

***Abstract*— As a contribution toward the goal of adaptable, intelligent artificial limbs, this work introduces a continuous actor-critic reinforcement learning method for optimizing the control of multi-function myoelectric devices. Using a simulated upper-arm robotic prosthesis, we demonstrate how it is possible to derive successful limb controllers from myoelectric data using only a sparse human-delivered training signal, without requiring detailed knowledge about the task domain. This reinforcement-based machine learning framework is well suited for use by both patients and clinical staff, and may be easily adapted to different application domains and the needs of individual amputees. To our knowledge, this is the first myoelectric control approach that facilitates the online learning of new amputee-specific motions based only on a one-dimensional (scalar) feedback signal provided by the user of the prosthesis.**

I. INTRODUCTION

Multi-function myoelectric prostheses are devices that monitor electrical signals produced by muscle tissue in limb-deficient patients and use these signals to control the movement of a multiple-actuator robotic appendage [1]. Myoelectric control can be challenging for new amputees, and the transition to a powered prosthesis often requires repeated calibration of the artificial limb by patients and physiotherapists. This difficulty is in part due to the complex and variable nature of the electromyograph (EMG) signals used for limb control, and in part due to the lack of flexibility inherent in commercial control methods.

Work has been done to address these problems, using enhancements to traditional control schemes and more advanced methods based on machine learning techniques. Specific research into machine learning techniques for myoelectric control of upper limb prostheses has been an active area since the 1970s [1]. Notable examples include the use of linear discriminant analysis [2], EMG-based low-dimensional embeddings [3], artificial neural networks [4], and support vector machines [5]. While the majority of this prior work focused on offline supervised learning techniques, additional research has demonstrated unsupervised techniques [6], and semi-supervised techniques that are trained online (e.g., the work of Nishikawa et al. using human feedback in learning to classify a repertoire of prosthetic hand gestures [7]). Reviews by Parker et al. [1], and Oskoei and Hu [8], provide in-depth coverage of the current state of myoelectric control research.

P.M. Pilarski, T. Degris, and R.S. Sutton are with the Department of Computing Science, University of Alberta, Edmonton, AB T6G 2E8, Canada Please direct correspondence to: pilarski@ualberta.ca

M.R. Dawson and J.P. Carey are with the Department of Mechanical Engineering, University of Alberta, Edmonton, AB T6G 2G8, Canada

F. Fahimi is with the Department of Mechanical and Aerospace Engineering, University of Alabama in Huntsville, Huntsville, AL 35899, USA

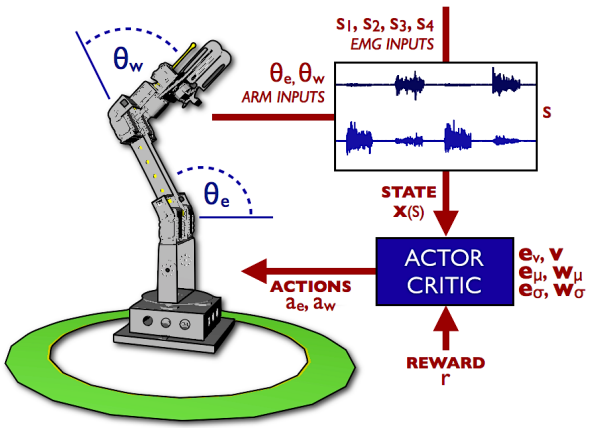

Fig. 1. A schematic diagram of actor-critic reinforcement learning applied to a two-joint robotic arm. At each time step, a state approximation $\mathbf{x}(s)$ and a scalar user-provided reward signal r are given to the learning system. Based on this, the system updates its weights and generates two continuous joint velocity actions, a_e and a_w, that are given as input to the arm.

One known limitation of most methods is that controllers do not adapt over time to changes in the patient, the patient's intent, or the patient's usage patterns [6]. Clinical and technical intervention is typically required for an amputee to improve or update the control policy of their prosthesis. Even for existing adaptive methods (as summarized by Sensinger et al. [6]), initializing a controller and making amputee-specific changes requires detailed expert knowledge regarding the physiology of an individual patient and the hardware of the patient's prosthesis. There are also cases where an amputee has a desired movement goal (e.g., "zip up my new backpack") but even in the clinic it may be difficult to frame their objective in terms of control parameters or existing device gestures; it is currently impossible for most amputees to improve their limb controllers independently, outside the clinic.

The following questions therefore remain challenging open problems for research: 1) how best to automatically translate multiple, overlapping muscle signals into usable control commands for a multi-function mechanical limb; 2) how to automatically tailor the control system to the needs and specific physical conditions of individual patients, without constant manual intervention, retraining, and periods of frustration or reduced function for the patient; and, 3) how to continuously improve control based on patient feedback.

Proceedings of the 2011 IEEE International Conference on Rehabilitation Robotics, Zurich, Switzerland, June 2011, pp. 134–140.

Fig. 2. Image of the AX-12 Smart Arm robot workspace and EMG setup.

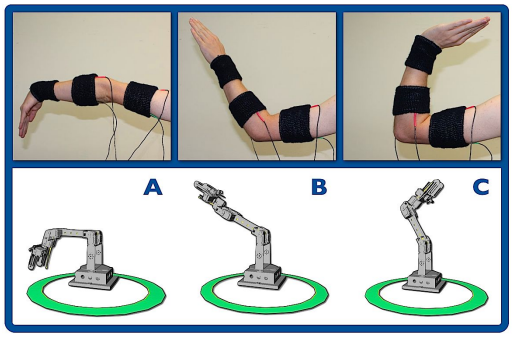

Fig. 3. *Top:* electrode locations on an able-bodied subject's arm, and limb positions for *reach* (A), *relax* (B), and *retract* (C) activity types. *Bottom:* the corresponding target joint angles $\langle \theta_e, \theta_w \rangle$ for the simulated appendage.

With these questions in mind, we propose an adaptive algorithmic approach to generalized myoelectric control that pairs *online human training* of controllers with a statistical machine learning algorithm called *continuous actor-critic reinforcement learning* (ACRL) [9], [10]. Reinforcement learning (RL) is an approach for solving optimal control problems in which a control policy is learned through repeated trial and error interactions between a learning system and its environment [11]. In RL, a learning system aims to maximize the expected sum of a scalar feedback signal, termed *reward*. It does so even in cases where an *a priori* model of the problem environment is not available. A schematic of this approach, as applied to limb control, is shown in Figure 1.

RL methods, and particularly ACRL, have several advantages that make them well-suited to a general prosthetic learning task. First, ACRL methods are parameter based, allowing incremental (linear) computation of variables and weight vectors. Learning updates are therefore fast to compute, with millisecond-scale updates possible even for very large problems (e.g., those with millions of features). In addition, past samples of experience are not stored in memory. As such, memory requirements remain constant throughout learning. This facilitates learning over very long time frames and when using embedded hardware with limited resources. Second, through the use of temporally extended credit assignment, it is possible to learn quickly even when teaching signals are sparse [11], [12]. Finally, when supplemented with function approximation schemes (e.g., tile coding [11]), ACRL methods scale well to continuous-space real-world tasks, learning to predict and control a complex environment using a stream of sensorimotor data [13]. RL also provides an opportunity for intuitive human-computer interaction—user feedback can take the form of a scalar, goal-directed signal of approval or disapproval (e.g., the reinforcement-based training system of Knox and Stone [14]).

Not surprisingly, RL techniques have found use in a number of robotic and biomedical problem domains. Peters and Schaal have demonstrated ACRL methods as applied to learning complex movement systems and motor primitives for humanoid robotics [10], [15], and Izawa et al. showed how RL can be used in a reaching task to move a simulated biological two-joint arm to a goal region using an explicit reward signal (though with no human or EMG input) [16]. Tamei and Shibata demonstrated a policy-gradient RL system that used EMG data from an able-bodied subject to control a robotic arm in a collaborative lifting task [17]. Of note in the context of rehabilitation, Thomas et al. presented the application of an actor-critic architecture to the functional electrical stimulation control of a biological human arm [13].

Despite this promising body of research, the use of ACRL algorithms specifically for the myoelectric control of powered prostheses has been largely unexplored. In addition, most of the ACRL approaches above rely on predefined, *a priori* reward signals based on expert knowledge and/or the absolute spatial or kinematic error in the workspace. This makes it impossible for an amputee to adapt a controller online based on his or her current needs and goals. As a contribution toward the goal of adaptable, intelligent artificial limbs, this paper demonstrates the use of intuitive human feedback and ACRL to enable the user-specific initialization and optimization of multi-function myoelectric prostheses.

II. METHODS

A. Robotic Arm and EMG Data Acquisition

The AX-12 Smart Arm (Crustcrawler, Inc.) was chosen as a cost-effective physical test platform to mimic the functionality of commercial myoelectric prostheses such as the Utah Arm 3 (http://www.utaharm.com/ua3.php). The Smart Arm, as seen in Figure 2, includes five degrees of freedom: hand open/close, wrist flexion/extension, wrist rotation, elbow flexion/extension, and shoulder rotation. The size of the robotic arm is about half that of a human arm. For this initial study, a Smart Arm simulator was used that shares the dimensions and kinematic model of the physical device. On each time step, angular velocity commands are sent to the five joints (simulated servos) as integers in the range $[-1023, 1023]$. Commands outside this range are cropped. Possible feedback from the simulated arm includes the angle (in radians) and angular velocity (in radians per second) of each joint, and the Cartesian position of the end effector.

The EMG signals used in learning were recorded with BL-AE-N pre-amplified surface electrodes and acquired via a National Instruments PCI-6259 data acquisition card. After acquisition the signals were post processed digitally using a notch (60Hz) and high pass (10Hz cutoff) filter to reduce power line and motion artifact noise respectively. Batch data

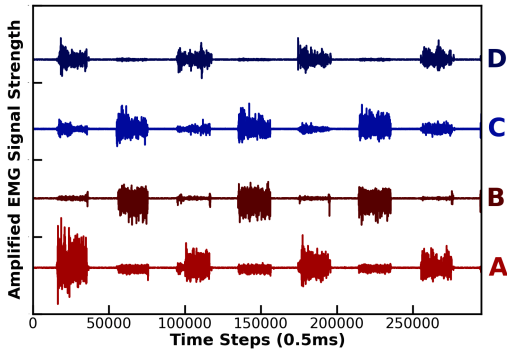

Fig. 4. Time varying amplitude profile for the four input EMG signals: (A) wrist extensors, (B) wrist flexors, (C) triceps, and (D) biceps. Traces stacked vertically to show temporal overlap; $\sim$ 10V between vertical ticks.

were sampled from four input electrodes and a reference electrode that were affixed to the left arm of an able-bodied human subject (Figure 3). Electrode locations were selected for maximal EMG signal strength on each of the following muscle groups: *biceps* (BI), *triceps* (TRI), *wrist flexors* (WF), and *wrist extensors* (WE). The reference sensor was affixed to the bony part of the wrist. The subject was directed to perform a repeated reaching task: at ten second intervals they were signalled to reach, relax, or retract their arm, strongly flexing the BI/WE and TRI/WF muscle group pairings in an alternating pattern. These three activity patterns are depicted in Figure 3. EMG voltages were recorded every 0.5ms, along with the target arm position. Two datasets—one for testing and one for training—were collected from a single subject on different days, each lasting approximately 5 minutes. An example of this raw EMG data is shown in Fig. 4.

B. The Continuous Actor-Critic Algorithm

For this work, we consider the standard RL framework described by Sutton and Barto [11], with a discrete-time Markov Decision Process (MDP). This MDP contains an infinite set $\mathcal{S}$ of states $s \in \Re^{n_S}$ and actions $a \in \Re$, where n_s is the number of dimensions of $\mathcal{S}$. We denote s_t, a_t and r_t as the state, action, and reward respectively at time t; action a_t may affect the reward received on future time steps $(r_{t+i}, i > 0)$, but not the current reward (r_t).

In this paper, we focus on actor-critic algorithms, a subclass of policy gradient algorithms. Here the *control policy* of a system, denoted $\pi(a|s)$, is a function that defines the probability with which the system will select action a in state s. We assume that π is characterized by a vector of parameters $\mathbf{w} \in \Re^n$, and that for any state-action pair, $\pi(a|s)$ is continuously differentiable in $\mathbf{w}$. The goal of policy gradient methods is to find a policy π that maximizes a long-term reward objective, for this work defined as:

$$J(\pi) = \lim_{T\to\infty} \mathbb{E}_\pi \left[\sum_{t=0}^{T-1} \gamma^t r_{t+1} \right],$$

with a discounting factor $0 \leq \gamma < 1$. Policy gradient methods update the policy parameter vector $\mathbf{w}$ in the direction of the gradient of the return $J(\pi)$ according to $\mathbf{\Delta_w} \propto \nabla_{\mathbf{w}} J(\pi)$,

Algorithm 1 Continuous Actor-Critic Algorithm

1: **initialize:** $\mathbf{w}_\mu, \mathbf{w}_\sigma, \mathbf{v}, \mathbf{e}_\mu, \mathbf{e}_\sigma, \mathbf{e}_\mathbf{v}, s$
2: **repeat:**
3: $\quad \mu \leftarrow \mathbf{w}_\mu^T \mathbf{x}(s)$
4: $\quad \sigma \leftarrow \exp[\mathbf{w}_\sigma^T \mathbf{x}(s) + \log(\sigma_c)]$
5: $\quad a \leftarrow \mathcal{N}(\mu, \sigma^2)$
6: $\quad$ **take action** a, **observe** r, s'
7: $\quad \delta \leftarrow r + \gamma \mathbf{v}^T \mathbf{x}(s') - \mathbf{v}^T \mathbf{x}(s)$
8: $\quad \mathbf{e_v} \leftarrow \lambda \mathbf{e_v} + \mathbf{x}(s)$
9: $\quad \mathbf{v} \leftarrow \mathbf{v} + \alpha_\mathbf{v} \delta \mathbf{e_v}$
10: $\quad \mathbf{e}_\mu \leftarrow \lambda \mathbf{e}_\mu + (a - \mu)\mathbf{x}(s)$
11: $\quad \mathbf{w}_\mu \leftarrow \mathbf{w}_\mu + \alpha_\mathbf{w} \delta \mathbf{e}_\mu$
12: $\quad \mathbf{e}_\sigma \leftarrow \lambda \mathbf{e}_\sigma + [(a - \mu)^2/\sigma^2 - 1]\mathbf{x}(s)$
13: $\quad \mathbf{w}_\sigma \leftarrow \mathbf{w}_\sigma + \alpha_\mathbf{w} \delta \mathbf{e}_\sigma$
14: $\quad s \leftarrow s'$

where $\mathbf{\Delta_w}$ is the change in $\mathbf{w}$ and $\nabla_\mathbf{w} J(\pi)$ is the gradient of $J(\pi)$ w.r.t. to $\mathbf{w}$ [18]. This gradient can be estimated from samples $\langle s_t, a_t, s_{t+1}, r_{t+1} \rangle$ of interaction between the learning system and its environment [19]. However, the estimate of the gradient can have a high variance, thus requiring a large numbers of samples to converge.

Actor-critic methods reduce this variance by using two learning elements: a *critic* and an *actor*. The actor shapes the policy π and selects actions. The critic predicts the expected future differential reward while following π. In other words, the critic represents the value function $V^\pi(s)$ of the current policy π in the state s:

$$V^\pi(s) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_0 = s \right].$$

An estimate of $V^\pi(s)$ is then used as a baseline to compute $\mathbf{\Delta_w}$. Though $V^\pi(s)$ is not known, it can be estimated using temporal difference (TD) learning [11]. For more detail, comprehensive descriptions of RL and AC methods are given by Sutton and Barto [11], and Peters and Schaal [10].

Because $\mathcal{S}$ is continuous, a standard function approximation method known as *tile coding* is used to expand the state s into a high-dimensional binary feature vector $\mathbf{x}(s)$. This expansion discretizes the space, and allows generalization and non-continuity in the learned policies [11]. Tile coding defines a set of N_T exhaustive partitions, or *tilings*, of the input space s. Each tiling is a grid of resolution N_R where only one element of the partition, a *tile*, is active. Note that the number of non-zero features in $\mathbf{x}(s)$ is always equal to N_T and does not depend on s.

The continuous actor-critic algorithm used in this paper is described in Algorithm 1. In it, the actor selects a new action a from a normal distribution $\mathcal{N}(\mu, \sigma^2)$ with mean $\mu = \mathbf{w}_\mu^T \mathbf{x}(s)$ and standard deviation $\sigma = \exp[\mathbf{w}_\sigma^T \mathbf{x}(s) + \log(\sigma_c)]$ (lines 1–3). $\mathbf{w}_\mu$ and $\mathbf{w}_\sigma$ are parameter vectors for the mean and standard deviation of the actor, and σ_c is the starting value for σ. After the system executes action a and observes the new state s' and reward r (line 6), the critic computes a temporal difference error δ from r, $\mathbf{v}^T \mathbf{x}(s)$, and $\mathbf{v}^T \mathbf{x}(s')$—the current estimates of $V^\pi(s)$ and $V^\pi(s')$ respectively (line 7); $\mathbf{v}$ is the critic's parameter vector. A core problem

in RL is credit assignment—i.e., how to assign reward or punishment to past decisions that may have contributed to the current situation. *Eligibility traces* are a standard mechanism in RL to help to fill the gap between the decisions and their consequences. We use accumulating traces, as per Sutton and Barto [11]. In the critic, the trace $\mathbf{e_v}$ of the feature vector $\mathbf{x}(s)$ is updated (line 8) and then used to update the critic parameters $\mathbf{v}$ (line 9); λ is the trace decay rate. The actor maintains traces $\mathbf{e}_\mu$ and $\mathbf{e}_\sigma$, updated in line 10 and 12; these are used to update the parameters $\mathbf{w}_\mu$ and $\mathbf{w}_\sigma$ respectively for the mean and variance of its policy (line 11 and 13). $\alpha_\mathbf{v}$ and $\alpha_\mathbf{w}$ are scalar step-size (learning rate) parameters.

In cases where more than one action $(a_t^1, a_t^2, ...)$ is required, one set of actor parameter vectors $(\mathbf{w}_\mu, \mathbf{w}_\sigma, \mathbf{e}_\mu, \mathbf{e}_\sigma)$ is maintained for each action (e.g., Tamei and Shibata [17]).

III. EXPERIMENTS

Two experiments were conducted: learning a control policy using a fixed goal-based reward (Sec. III-A), and learning a control policy from a human-delivered reward signal (Sec. III-B). In both experiments, the goal of the learned control policy was to output joint velocity commands that caused the simulated arm to successfully track a temporal sequence of desired joint angles—in this case, the set of reaching, retracted, and relaxed activities described in Sec. II-A.

The learning system's action space consisted of two continuous angular velocity values, one for the wrist joint motor (a_w) and one for the elbow joint motor (a_e). Actions were computed on each timestep as shown in Algorithm 1. A continuous state space consisting of $s = \langle \theta_w, \theta_e, \bar{s}_1, \bar{s}_2 \rangle$ was used, where θ_w and θ_e indicate the current wrist and elbow joint angles, and $\bar{s}_1$ and $\bar{s}_2$ are two differential EMG signals (defined as follows). As per standard practice, each raw EMG signal s_E was rectified and averaged as $\bar{s} = (1-\tau)\bar{s} + \tau|s_E|$, with a time constant $\tau = 0.05$. Differentials were then computed for each pair of opposing muscle groups: $\bar{s}_1 = \bar{s}_{BI} - \bar{s}_{TRI}$ and $\bar{s}_2 = \bar{s}_{WF} - \bar{s}_{WE}$. While not essential to finding a successful policy, this reduction in problem dimensionality decreased memory and processing requirements, and allowed for faster policy convergence. All components in s were normalized to the range $[0, 1]$ according to their minimum and maximum possible values.

This state space was used to construct the state approximation vector $\mathbf{x}(s)$ used in learning. To capture different levels of generalization, $\mathbf{x}(s)$ was a concatenation of $N_T = 25$ incrementally offset tilings of s, each at four different resolution levels $N_R = [5, 8, 12, 20]$, along with a single active baseline unit. This resulted in a single binary feature vector consisting of 4,636,426 features. Exactly $m = 101$ features in $\mathbf{x}(s)$ were active at any given time, one for each tiling and one for the baseline feature. The learning parameters were set as follows: $\alpha_\mathbf{v} = 0.1/m, \alpha_\mathbf{w} = 0.01/m, \gamma = 0.99, \lambda = 0.3$, and $\sigma_c = 1023$. Weight vectors $\mathbf{e_v}, \mathbf{e}_\mu, \mathbf{e}_\sigma, \mathbf{v}, \mathbf{w}_\mu, \mathbf{w}_\sigma$ were initialized to 0, and σ was bounded by $\sigma \geq 1$.

The ACRL system was trained online, with repeated cycles being made through the training EMG data described in Sec. II-A. Learning updates and action choice occurred after every 5ms of recorded data. Total training time was held constant, after which point learning was halted and the learned control policy was applied to the testing EMG data set. Performance was measured on both training and testing data in terms of the learning system's ability to achieve the desired target angles.

To improve the speed of policy convergence, the training task was divided into three sub-tasks of increasing difficulty. Initially, parameter vectors for the wrist and elbow joints were each trained in isolation for 100k time steps; the learning system was able to move one of the joints, while the other joint was actuated through the range of desired motions. This was followed by a period of combined two-joint training where the learning system was free actuate both joints. As described by Sanger, breaking down a task into regions of solvability in this way can decrease the total time needed to learn more complex composite behaviours [20]. We found this staged approach decreased policy convergence time, and also made the interactive training process easier for human users. Learning and simulation occurred on a MacBook Pro 2.53 GHz Intel Core 2 Duo laptop.

A. Learning from Goal-Based Reward Signals

In the first experiment, reward was given based on a predetermined structure independent of human feedback. This examined the ability of the system to learn the target concept when provided with only a simple "success or failure" goal-based training signal. A positive reward of $r_t = 1.0$ was delivered when θ_w and θ_e were both within 0.1 radians of their target angles. A reward of $r_t = -0.5$ was delivered in all other cases, in essence penalizing the learning system when the arm's posture differed from the target posture. Each goal-based experiment lasted 750k time steps.

B. Learning from Sparse Human Reward Signals

In the second experiment, reward was given based on a sparse human feedback signal r_h. This examined the ability of the system to learn an arbitrary user-specified behaviour in response to the input EMG signals. In this experiment, a human user was shown a 3D model of the target position of the arm, superimposed on the current (true) position of the arm. Throughout training, the user was given the option to provide—using key presses—a positive $(r_h = 0.5)$ or negative $(r_h = -0.5)$ reward to the system in response to its behaviour. In the absence of human-delivered reward, the system was provided with a static reward of $r_h = 0$. To provide a consistent training signal across millisecond timescale updates, and bridge the gap between human response speeds (seconds) and learning update time scales (ms), the reward signal on each timestep following a human delivered reward was equal to a decayed trace of the previous step's reward: $r_{t+1} = 0.01 r_t + r_h$, until the trace fell below a small threshold value; a suitable rate of decay was determined experimentally. Without this extension, we observed that δ remained high, preventing μ and σ values from converging to an accurate policy. Each experiment lasted 300k time steps, and required less than 10min of user interaction.

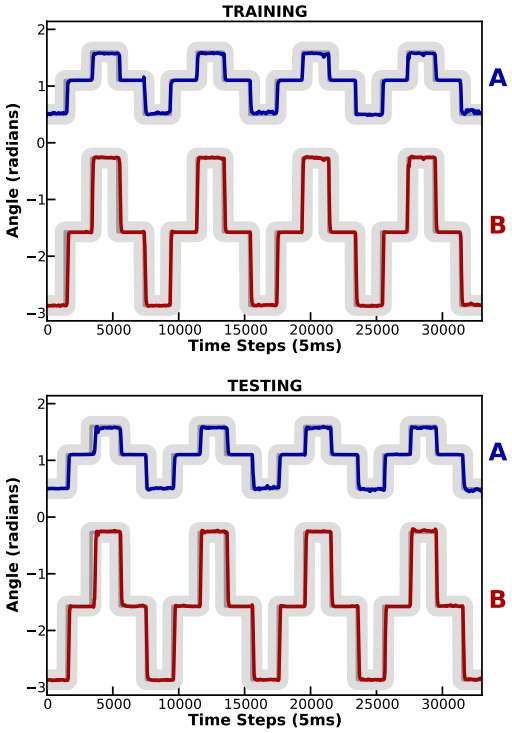

Fig. 5. Control accuracy on *training data* (top) and *testing data* (bottom) after 750k time steps of training for elbow (A) and wrist (B) joint angles. Control policy was learned from a staged task with a goal-based reward signal. Shown are the actual joint angles (red/blue), the mean target angle for each joint (dark grey), and the width of the target region (light grey).

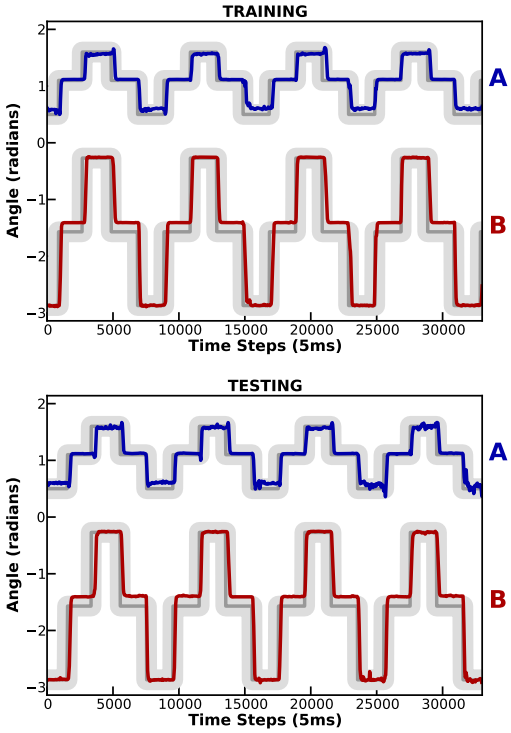

Fig. 6. Control accuracy on *training data* (top) and *testing data* (bottom) after 300k time steps of training for elbow (A) and wrist (B) joint angles. Policy was learned from a staged task with an online human-defined reward signal. Shown are the actual joint angles (red/blue), the mean target angle for each joint (dark grey), and the width of the target region (light grey).

IV. RESULTS

The first experiment examined the effectiveness of a policy learned from a staged task with an explicit, goal-based reward signal. Figure 5, top, shows joint control accuracy on the training dataset from a single able-bodied subject after 750k time steps of training, while Figure 5, bottom, shows the control accuracy of this learned policy on the independent testing dataset. Shown are the actual joint angles (red/blue), the mean target angle for each joint (dark grey), and the width of the target region where positive reward was delivered (light grey). In both testing and training cases, the joint angles remained within the target regions for the majority of the evaluation period.

The second experiment extended this approach to a policy learned from a staged task with an online human-defined reward signal. Figure 6, top, shows joint control accuracy on the training dataset after 300k time steps of training by a human user, while Figure 6, bottom, shows the control accuracy of this learned policy on the independent testing dataset. For both datasets, the learned policy tracked the angular targets for the majority of the evaluation period; we observed that additional human training further decreased the variability present for the reaching and retracted regimes, and sharpened the transitions between set points.

Figure 7 shows how the computed μ and σ values varied as learning progressed (shown here comparing starting and ending μ and σ values for one run of a staged task with online human training). While initially random, learned μ values took on a final pattern with alternating periods of increased velocity followed by periods of stability. The ACRL system learned to create periods of increased velocity that moved the arm from one stable position to the next (e.g., from relaxed to retracted). This is visible as regular positive or negative velocity spikes in Figure 7, top right.

Similar stabilization was observed for the learned σ values (Figure 7, bottom). From the initial starting value of $\sigma_c = 1023$, the magnitude of σ decreased over time in response to a decreasing TD-error δ (i.e., the learning system's predictions about its long-term reward became closer to the reward actually received.) As evident in Figure 7, bottom, σ continued to decrease with learning; by the end of the training period, σ had decreased to very small values for the relaxed position ($\sigma < 1$), and to larger values for reaching and retracted activity ($\sigma < 200$ and $\sigma < 400$, respectively). With further human training, the σ values for these targets continued to shrink, leading to more precise action choice and less jitter in arm movement.

As mentioned in Section III, the use of two differential EMG signals ($\bar{s}_1, \bar{s}_2$) in place of the four raw EMG signals allowed faster learning and significantly reduced memory and processing requirements. However, structuring of the signal domain in this way was not required for successful learning. Learning systems using the four base EMG signals also demonstrated the ability to track target joint angles.

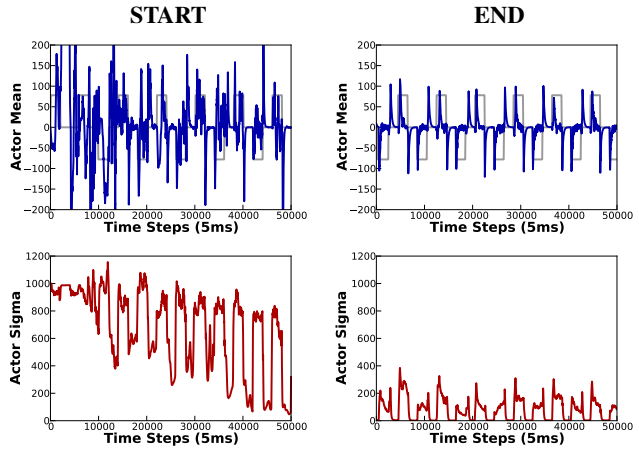

Fig. 7. Results comparing starting and ending μ (top, blue) and σ (bottom, red) values for a staged task with online human training. *Top:* While initially random, after learning μ values take on a final pattern of alternating periods of increased velocity followed by periods of stability. *Bottom:* σ values decrease over the course of learning.

V. DISCUSSION

One advantage of the ACRL-based human training approach presented in this work is that it does not assume (or require) intensive manual shaping of the input signal space or problem domain. Even without knowing the exact set of motions or motor commands that need to be performed by a robotic appendage, or how the values from any given set of EMG sensors relate to a desired motion, controllers can be trained to execute specialized movements using only a series of positive and negative rewards indicating user approval or disapproval. This approach is therefore well suited to complex patient-specific control cases such as non-physiologically mapped EMG control and targeted muscle reinnervation (TMR, where amputees have had some of their nerves attached to muscle tissue in alternate locations to facilitate prosthesis control [21]). Control objectives may be specified in terms of incremental, amputee-centric goals.

As shown by the pose tracking results in Figures 5 and 6, our ACRL framework was able to develop an accurate velocity control policy for responding to a heterogeneous input space made up of normalized EMG and joint angle signals. It learned this policy using only a single scalar reward signal indicating the success or failure of its action choices. Training using a scalar reinforcement signal of approval or disapproval from a human user—as done in this study—has been shown in other domains to allow users to optimize a system in a natural, accessible way, even in conditions where the exact mechanics of the system are unknown [14]. This can be viewed as an advantage for amputees, as it suggests the ability to customize a prosthesis to personal use patterns without the need to specify a collection of device-specific mechanical targets or clinically designed calibration tasks.

The impact of user decisions on policy tuning can be observed in the Figure 6. Systematic offsets (i.e., a consistent shift up or down for a given activity type) were largely due to human choices during training—with human-delivered reward, the user must decide based on visual cues and personal preference how well they feel the arm is performing. Their reward allotment (and thus the learned control policy) reflects these decisions. This further reduces the need for biological and mechanical domain knowledge when optimizing a new device, and also minimizes the computational background required of clinical staff. Such an approach is in contrast to the majority of prior related work, in which state spaces are typically crafted to match each particular application domain, and feedback (or training) signals are continuous functions of physical—e.g., spatial or kinematic—error in the workspace.

The learning system's ability to flexibly adapt to an arbitrary input space is also derived from the way information is interpreted by the system through function approximation. As described in Section II-B, tile coding function approximation is used to expand the set of joint angle and EMG input signals into a high-dimensional binary representation. This representation allows the system to generalize and form complex, non-continuous policies. From the system's perspective, the input state of the system appears in a subjective fashion—i.e., there is nothing that indicates how a bit in $\mathbf{x}(s)$ relates to any particular input signal or combination of signals. This gives further potential for "out of the box" application by end users in an online setting, allowing rapid adaptation for use with new patients, prosthesis models, and even amputation types without significant changes to the control framework.

ACRL selects actions based on normal distributions with policy parameters μ and σ (shown over the course of learning in Figure 7). One of the significant advantages to using a continuous action space of this form is that policy exploration is handled in an automatic fashion by the changing σ term [15]. This provides an effective mechanism to detect environmental change and adjust the control policy if the user's intent shifts over time. If the trainer stops providing reward for a behaviour, or provides vastly different reward from the learning system's previous experience, the weights for σ will begin to increase in response to the rising TD-error δ; this will promote more exploration of the policy space, and a corresponding shift in μ. In a rehabilitation context, this suggests that an ACRL system will be able to detect and adapt to the slight mechanical and biological changes that occur during day to day use of a prosthesis, without the need for intensive recalibration and clinical retraining.

However, as in other training environments, it is important that reward and feedback be applied consistently to an ACRL framework. If the increase in δ is due to prolonged human inconsistency, the subsequent increase in σ may degrade a stable (good) policy. While it is possible to freeze learning after periods of training (a clear delineation between training and use periods), a much better approach would be to develop methods capable of maintaining a consistent reward signal during periods of reduced human interaction. As described by Knox and Stone, distributing sparse human reward to a learning framework is a challenging problem [14]; a principled approach is the subject of ongoing investigation. While outside the scope of the present work, we expect that a detailed comparison of different reward distribution approaches will yield ways to improve the learning process.

As a first demonstration, this work explored pose tracking via joint velocity modulation. However, the reward-based problem formulation suggests that it can be applied to any type of position, velocity, or force tracking problem with little or no change to the learning architecture. One immediate example is the tracking of continuous arm movement, where reward is delivered when the speed and direction of the artificial limb matches the intent of the human user.

Recent work by DiGiovanna et al. demonstrated a co-adaption process that occurs between users and RL algorithms as they work together toward a common goal [22]. The impact of this online co-adaptation on the present approach is an interesting area for future investigation. We expect that learned control policies will improve as users become more skilled and refine their own EMG signals to provide more consistent input signals to the learning system.

The present study assessed the suitability of ACRL techniques for myoelectirc control in the context of a simulated robotic environment. Based on our preliminary experiments with the physical robotic platform, we expect this work to transfer well to real-world use. We are currently studying the effectiveness of human-trained controllers on both the simulated arm and the physical robotic platform using online EMG data, and examining the ability of ACRL to robustly deal with inter-individual differences. Future work will also explore the use of state spaces with additional EMG and force sensors, EMG features (e.g., frequency information or signal zero crossings [6]), and robot feedback signals (e.g., motor velocity, temperature, and/or current draw) to allow greater controller flexibility and precision.

VI. CONCLUSIONS

This work represents an important incremental step toward the objective of adaptable, intelligent myoelectric prostheses. It introduces the use of human-trained ACRL for myoelectric limb control, contributing a flexible, online learning framework that may be easily adapted to different application domains and the needs of individual amputees. The result is an intuitive reinforcement-based controller training approach that is well suited for use by amputees and clinical staff.

Specifically, we demonstrated that a human user could train a virtual robotic appendage and associated ACRL framework to interpret complex, overlapping EMG signals using only a sparse reward signal indicating the approval or disapproval of the human trainer. The system was able to learn a two-joint velocity control task, and demonstrated the ability to change over time according to continued user input, without the need to integrate detailed task and domain knowledge into the learning framework.

To our knowledge, this is the first myoelectric control approach that enables the online learning of new amputee-specific motions based only on a single, scalar reward signal provided by the user of the prosthesis. In the long term, we expect these methods to improve quality of life for amputees by giving them greater control and autonomy over their powered prostheses, and by increasing their engagement and self-sufficiency when adapting to new prosthetic devices.

ACKNOWLEDGMENTS

This work was supported by the Alberta Ingenuity Centre for Machine Learning, the Natural Sciences and Engineering Research Council, Alberta Innovates—Technology Futures, and the Glenrose Rehabilitation Hospital. The authors thank A. Alloway, J. Duncan, K. Hensel, R. Miller, and J. Tam for their work on the AX-12 Smart Arm simulator.

REFERENCES

[1] P. Parker, K. B. Englehart, and B. Hudgins, "Myoelectric signal processing for control of powered limb prostheses," *J. Electromyography Kinesiol.*, vol. 16, no. 6, pp. 541–548, 2006.

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

[3] P. K. Artemiadis and K. J. Kyriakopoulos, "EMG-based control of a robot arm using low-dimensional embeddings," *IEEE Trans. Rob.*, vol. 26, no. 2, pp. 393–398, 2010.

[4] B. Hudgins, P. Parker, and R. N. Scott, "A new strategy for multifunction myoelectric control," *IEEE Trans. Biomed. Eng.*, vol. 40, no. 1, pp. 82–94, 1993.

[5] M. A. Oskoei and H. Hu, "Support vector machine-based classification scheme for myoelectric control applied to upper limb," *IEEE Trans. Biomed. Eng.*, vol. 55, no. 8, pp. 1956–1965, 2008.

[6] J. Sensinger, B. Lock, and T. Kuiken, "Adaptive pattern recognition of myoelectric signals: exploration of conceptual framework and practical algorithms," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 17, no. 3, pp. 270–278, 2009.

[7] D. Nishikawa, W. Yu, H. Yokoi, and Y. Kakazu, "On-line learning method for EMG prosthetic hand control," *Electronics and Communications in Japan, Part III: Fundamental Electronic Science*, vol. 84, no. 10, pp. 35–46, 2001.

[8] M. A. Oskoei and H. Hu, "Myoelectric control systems—a survey," *Biomed. Signal Process. Control*, vol. 2, no. 4, pp. 275–294, 2007.

[9] R. S. Sutton, "Temporal credit assignment in reinforcement learning," Ph.D. dissertation, University of Massachusetts, Amherst, 1984.

[10] J. Peters and S. Schaal, "Natural actor-critic," *Neurocomputing*, vol. 71, no. 7-9, pp. 1180–1190, 2008.

[11] R. S. Sutton and A. G. Barto, *Reinforcement learning: an introduction*. MIT Press Cambridge, Massachusetts; London, England, 1998.

[12] S. Singh and R. S. Sutton, "Reinforcement learning with replacing eligibility traces," *Mach. Learn.*, vol. 22, no. 1, pp. 123–158, 1996.

[13] P. Thomas, M. Branicky, A. van den Bogert, and K. Jagodnik, "Application of the actor-critic architecture to functional electrical stimulation control of a human arm," *Twenty First Innovative Applications of Artificial Intelligence Conference*, pp. 1–9, 2009.

[14] W. B. Knox and P. Stone, "Interactively shaping agents via human reinforcement: The TAMER framework," *Proceedings of the Fifth Int. Conf. on Knowledge Capture (K-CAP'09)*, pp. 9-16, 2009.

[15] J. Peters and S. Schaal, "Reinforcement learning of motor skills with policy gradients," *Neural Networks*, vol. 21, no. 4, pp. 682–697, 2008.

[16] J. Izawa, T. Kondo, and K. Ito, "Biological arm motion through reinforcement learning," *Biol. Cybern.*, vol. 91, no. 1, pp. 10–22, 2004.

[17] T. Tamei and T. Shibata, "Policy gradient learning of cooperative interaction with a robot using user's biological signals," *Proceedings of ICONIP 2008, Part II, LNCS* 5507, pp. 1029-1037, Springer, 2009.

[18] R. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3–4, pp. 229–256, 1992.

[19] R. S. Sutton, D. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," *Adv. in Neural Info. Proc. Sys. (NIPS) 12*, pp. 1057–1063, 2000.

[20] T. D. Sanger, "Neural network learning control of robot manipulators using gradually increasing task difficulty," *IEEE Trans. Robot. Auto.*, vol. 10, no. 3, pp. 323–333, 1994.

[21] T. A. Kuiken, G. Li, B. A. Lock, R. D. Lipschutz, L. A. Miller, K. A. Stubblefield, K. B. Englehart, "Targeted muscle reinnervation for real-time myoelectric control of multifunction artificial arms," *JAMA–J. Am. Med. Assoc.,* vol. 301, no. 6, pp. 619–628, 2009.

[22] J. DiGiovanna, B. Mahmoudi, J. Fortes, J. C. Principe, and J. C. Sanchez, "Coadaptive brain-machine interface via reinforcement learning," *IEEE Trans. Biomed. Eng.*, vol. 56, no. 1, pp. 54–64, 2009.