Training an Actor-Critic Reinforcement Learning Controller for Arm Movement Using Human-Generated Rewards

Kathleen M. Jagodnik, *Member, IEEE*, Philip S. Thomas, Antonie J. van den Bogert,
Michael S. Branicky, *Fellow, IEEE*, and Robert F. Kirsch, *Member, IEEE*

***Abstract*—Functional Electrical Stimulation (FES) employs neuroprostheses to apply electrical current to the nerves and muscles of individuals paralyzed by spinal cord injury to restore voluntary movement. Neuroprosthesis controllers calculate stimulation patterns to produce desired actions. To date, no existing controller is able to efficiently adapt its control strategy to the wide range of possible physiological arm characteristics, reaching movements, and user preferences that vary over time. Reinforcement learning (RL) is a control strategy that can incorporate human reward signals as inputs to allow human users to shape controller behavior. In this paper, ten neurologically intact human participants assigned subjective numerical rewards to train RL controllers, evaluating animations of goal-oriented reaching tasks performed using a planar musculoskeletal human arm simulation. The RL controller learning achieved using human trainers was compared with learning accomplished using human-like rewards generated by an algorithm; metrics included success at reaching the specified target; time required to reach the target; and target overshoot. Both sets of controllers learned efficiently and with minimal differences, significantly outperforming standard controllers. Reward positivity and consistency were found to be unrelated to learning success. These results suggest that human rewards can be used effectively to train RL-based FES controllers.**

***Index Terms*—Artificial intelligence, human-machine teaming, Functional Electrical Stimulation, rehabilitation, reinforcement learning.**

Manuscript received September 6, 2016; revised February 16, 2017; accepted April 3, 2017. Date of publication May 2, 2017; date of current version October 23, 2017. This work was supported by the National Institutes of Health Fellowship under Grant TRN030167, in part by the Veterans Administration Rehabilitation Research and Development Predoctoral Fellowship—Reinforcement Learning Control for an Upper-Extremity Neuroprosthesis, and in part by the Ardiem Medical Arm Control Device under Grant W81XWH0720044. *(Corresponding author: Kathleen M. Jagodnik.)*

K. M. Jagodnik is with the Icahn School of Medicine at Mount Sinai, New York, NY 10029 USA (e-mail: kathleen.jagodnik@mssm.edu).

P. S. Thomas is with Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: pthomascs@gmail.com).

A. J. van den Bogert is with Cleveland State University, Cleveland, OH 44115 USA (e-mail: a.vandenbogert@csuohio.edu).

M. S. Branicky is with the University of Kansas, Lawrence, KS 66045 USA (e-mail: msb@ku.edu).

R. F. Kirsch is with Case Western Reserve University, Cleveland, OH 44016 USA (e-mail: rfk3@case.edu).

Digital Object Identifier 10.1109/TNSRE.2017.2700395

I. Introduction

FUNCTIONAL Electrical Stimulation (FES) involves the application of electrical current to nerves and muscles to restore voluntary movement to individuals paralyzed by spinal cord injury (SCI) [1], [2]. Our work aims to restore upper-extremity function to those affected by high-level SCI, who are paralyzed below the neck. Neuroprostheses consist of the stimulating hardware and control software that generate movement. FES control algorithms calculate muscular stimulation patterns required to achieve desired movements. Effective FES control algorithms are challenging to develop, due to a wide range of pathological physiological characteristics of individuals with SCI, including spasticity [3] and joint contractures [4].

The arm is particularly difficult to control because a large variety of reaching movements must be restored. Unlike the lower extremity, which typically requires cyclical (e.g. walking) or stereotyped (e.g. sit-to-stand) stimulation patterns, the upper extremity must be controlled using a wide range of uniquely specified stimulation patterns. Controllers have been developed for a range of upper-extremity functions including elbow extension [5], wrist stabilization [6], and hand grasp [7]. The feedback-controlled hybrid neuromuscular electrical stimulation (NMES)-exoskeleton of Klauer et al. [29] shows robust adaptation to new users and use conditions; it employs lockable joints so that only one joint is controlled at a particular time. In contrast, the present work aims to control two joints simultaneously to achieve the specified reaching movements, and does not employ an exoskeleton; this method is more similar to normal physiological function and is likely to produce more natural-looking movements. Iterative learning control (ILC) [30]–[32] achieves accurate performance for FES-related control of the triceps and anterior deltoid muscles when tested on a set of unimpaired test subjects, although this method requires repetitive learning to track specific, pre-defined trajectories and does not perform well unless tasks are very similar to those used for training [33], [34]. In contrast, for the present work that aims to restore a diversity of arm movements for FES applications, it will be important to train on a wide range of tasks to ensure that the resulting controller is generalizable. ILC has been shown to work well to control planar arm trajectories via triceps stimulation in hemiparetic stroke patients [34]–[36]; however, this method involved 18 hours of training per subject, and efficiency in controller training is an important consideration when aiming to develop an effective and practical control method. Despite the progress that has been made, no upper-extremity FES controller has been able to effectively and efficiently produce natural-looking reaching movements for a multiple-joint FES arm that includes the shoulder. Control techniques that are flexible and able to adapt to time-variant and user-specific physiological properties and user preferences will be needed to achieve this goal.

Reinforcement learning (RL) describes a class of control algorithms that learn by trial-and-error search to maximize a numerical reward signal by mapping situations to actions [8]. The RL controller explores a range of actions and, based upon the rewards that result, selects future actions according to these experiences. Interaction with a dynamic environment is fundamental to the RL controller training process [9]. Micera et al. [37] developed a fuzzy logic controller that uses RL to tune parameters for control of a simulated elbow-like system. Sigmoidal and sinusoidal trajectories were successfully tracked in the sagittal plane using this controller. An action-space constraint and relaxation technique was employed by Izawa et al. [38] to accelerate the learning of their simulated planar arm controller, and to facilitate stable learning. However, between 2,500 and 8,000 training trials, depending on arm stiffness, were required to achieve significant learning. In human FES systems, this large number of trials to achieve good controller behavior would be unacceptably high. Thomas [21] and Thomas et al. [20], [22] applied RL control to a planar arm simulation, achieving good performance within a few hundred episodes of training.

One feature of RL control is that it can incorporate human-generated reward signals to shape controller learning. Human-generated rewards have the potential to tailor RL controller performance to the preferences of each individual user, which may change over time. Rewards provided by human users have also been shown to increase RL controller learning speed for some domains [10]. RL controller training may benefit from human-generated reward signals because human trainers are able to perceive high-level performance characteristics that may be difficult for a computer program to recognize [11], such as planning long-term strategy or judging the natural appearance of simulated arm movements.

RL control shaped by human rewards has been explored for a number of systems with continuous state and action spaces. Vien and Ertel [39] and Vien et al. [40] demonstrated successful learning on two computer games with continuous state and action spaces, via human feedback signals using the ACTAMER (Actor-Critic Training an Agent Manually via Evaluative Reinforcement) framework, which employs function approximation of this signal. Continuous actor-critic RL with a sparse, human-generated training signal was used by Pilarski et al. [41] to successfully complete a 2-joint velocity control task in a simulated robotic arm. Most recently, Mathewson and Pilarski [42] compared the use of environmentally-derived rewards, human-generated rewards, and the combination of both to control a humanoid robot. They found that rewards assigned by humans augmented performance beyond rewards strictly derived from the environment.

Although the use of human rewards has the potential to improve RL controller learning, they also represent a challenging addition to RL systems. While computers can update state information and generate rewards on a millisecond timescale, humans typically have a reaction time of 0.3 to 0.8+ s to press a button (as when assigning a reward) when responding to a visual stimulus [49]. Such delays in rewards necessitate that the RL controller assess to how much of the preceding action the reward should be applied. This temporal credit assignment problem [8] is a challenge of RL control that is amplified when human-generated rewards, with their significant delays, are used.

Furthermore, human attention is finite, and humans are not able to train the RL controller by a constant sequence of viewing controller action and immediately assigning a reward; such an all-consuming training protocol would not permit useful integration into the daily life of the FES user. Instead, RL controllers should be designed to use sparse rewards from human trainers to effectively learn useful policies. While decreasing the frequency of rewards has been shown to slow learning speed [21], [43], techniques such as increasing the actor's learning rate [21] have been found to compensate for the slowed learning that results from sparse rewards.

To our knowledge, the use of human-generated rewards to train RL controllers for FES human arm control has not yet been explored. This control problem is distinct from previous work on RL control with human rewards for continuous-state, continuous action systems in that the planar arm model we employ in the present work requires 6 nonlinear, redundant muscles to be independently controlled. This control challenge is significantly more complex than in other systems involving the control of robots, the properties of which are more predictable.

In our previous work [12], [27], we used sparse, delayed, computer-generated pseudo-human rewards to shape RL control and demonstrated that these rewards could result in significant RL controller learning of goal-oriented reaching tasks for a simulated planar human arm. However, whether rewards generated by humans, which will be less consistent [24] and may vary subjectively over time [28], will train the controllers as effectively as computer-generated rewards remains an open question. Extending that work, in this study, ten neurologically intact human subjects generate reward signals to train an actor-critic RL architecture to control a planar arm.

These experiments aim to determine, for the control of goal-oriented reaching movements using a planar simulated human arm system, whether any fundamental similarities or differences exist when RL controllers trained using a population of human subjects are compared against controllers trained using a pseudo-human reward-generation algorithm [12]. The relative advantages of each training method will be determined, and based upon our findings, we will make recommendations for training actor-critic RL controllers for the planar arm system.

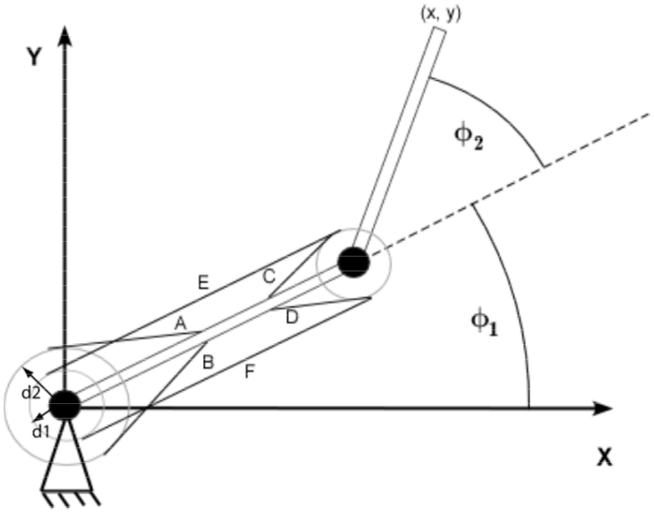


Fig. 1. Top view of the biomechanical arm model. The Y-axis is anterior. Movements occur in the sagittal plane with no gravity, as if sliding across a frictionless tabletop. Antagonistic muscle pairs are listed as (flexor, extensor): monoarticular shoulder muscles: (A: anterior deltoid, B: posterior deltoid); monoarticular elbow muscles: (C: brachialis, D: triceps brachii (short head)); biarticular muscles: (E: biceps brachii; F: triceps brachii (long head)). φ_1 and φ_2 are shoulder and elbow joint angles, respectively. Adapted from [13] and [26]. Moment arm values: $d_1 = 30$ cm, $d_2 = 50$ cm [12].

II. Methods

A. Experimental Setup

A planar biomechanical human arm and shoulder model implemented in the C language was used for all experiments [13], [27]. This model is named the Dynamic Arm Simulator 1 (DAS1) (Fig. 1). The model includes two segments (upper arm and forearm), two joints (shoulder and elbow), and 6 muscles (4 monoarticular and 2 biarticular). Muscles are modeled according to the Hill convention [14], [15], and are represented by two first-order ordinary differential equations [16]. Refer to [13] and [27] for additional model details.

The three flexor muscles of the arm model were weakened by 50% to simulate muscular atrophy [17], [18]. This produced a compromised level of baseline performance upon which the RL controllers had to improve.

The continuous actor-critic RL controller [19] (Algorithm 1) using artificial neural networks (ANNs) to represent the actor and critic was implemented for this arm model using C++ [20]–[22]. Fully connected feed-forward ANNs were used; the actor consists of 22 neurons (6 input, 10 hidden, 6 output), and the critic consists of 17 neurons (6 input, 10 hidden, 1 output), for a total of 39 neurons.

The ANN actor and critic weight vectors are initialized to the PD controller's actor policy, followed by initialization of the critic's weight vector by training with the actor's learning rate set to 0. Subsequently, eligibility traces, which record on a short-term basis learning-related events including visited states and performed actions, are initialized to 0. For each episode, the state of the system is initialized, muscle stimulation values are calculated and applied to the arm model, 20 ms is allowed to elapse, and the next state and reward are calculated (Fig. 2). TD error is calculated, the eligibility traces are updated, and updates are made to the actor and to the critic weights.

Algorithm 1 The Continuous Actor-Critic Reinforcement Learning Algorithm [19], [21]

```
Initialize ANN actor and ANN critic weight vectors $\vec{w}_a$ and $\vec{w}_c$ via error backpropagation supervised learning to PD controller's actor policy. Then, train with actor's learning rate = 0 to initialize critic's weight vector.
Initialize eligibility values to zero: $\vec{e}_c = 0$
Repeat for each episode (reaching task)
    s ← initial state of the system
    Repeat for each 20 ms time step within episode
        Calculate muscle stimulation levels a: a ← $\pi$(s) + n(t)
        Apply muscle stimulations to arm model
        Allow 20 ms to elapse
        Calculate next state s' and reward: $r_{Total} = r_{Automated} + r_{Human}$
        Compute TD error:
            $\delta(t) = r(t) + \frac{1}{\Delta t}\left[(1 - \frac{\Delta t}{\tau})V(t) - V(t - \Delta t)\right]$
        Update critic eligibility traces:
            $\kappa ei(t) = -e_i(t) + \frac{\delta V(s(t); w)}{\delta w_i}$
        Update actor:
            $\dot{w_i}^A = \eta_A \delta(t) n(t) \frac{\delta A(s(t); w^A)}{\delta w_i^A}$
        Update critic weights: $wi = \eta_C \delta(t) e_i(t)$
    Until maximum episode length reached
Until maximum run length reached
```

Symbols are defined as follows: π(s): actor's policy; n(t): explorational noise (defined in Equation 3); r(t): reward at timestep t;
V(t): evaluation of the value function at timestep t;
κ : constant to scale eligibility traces over time; η_A: actor learning rate; η_C: critic learning rate.

The actor generates actions according to

$$u(t) = S\left(A(s(t); w^A) + \sigma n(t)\right) \quad (1)$$

where u(t) is a set of six continuous muscle stimulation values ranging from 0.00 to 1.00 (indicated as Action(6) in Fig. 2), $A(\ )$ is the action-selection function, w^A is the vector of actor parameters encoding the policy, σ is a constant that scales exploration, $n(t)$ defines explorational noise:

$$\tau_n \dot{n}(t) = -n(t) + N(t) \quad (2)$$

whereτ_n is a time constant, and $N(t)$ is normal Gaussian noise having the same dimension as the action space. S is the monotonically increasing logistic function, defined as:

$$S(x) = \frac{1}{1 + e^{-x}} \quad (3)$$

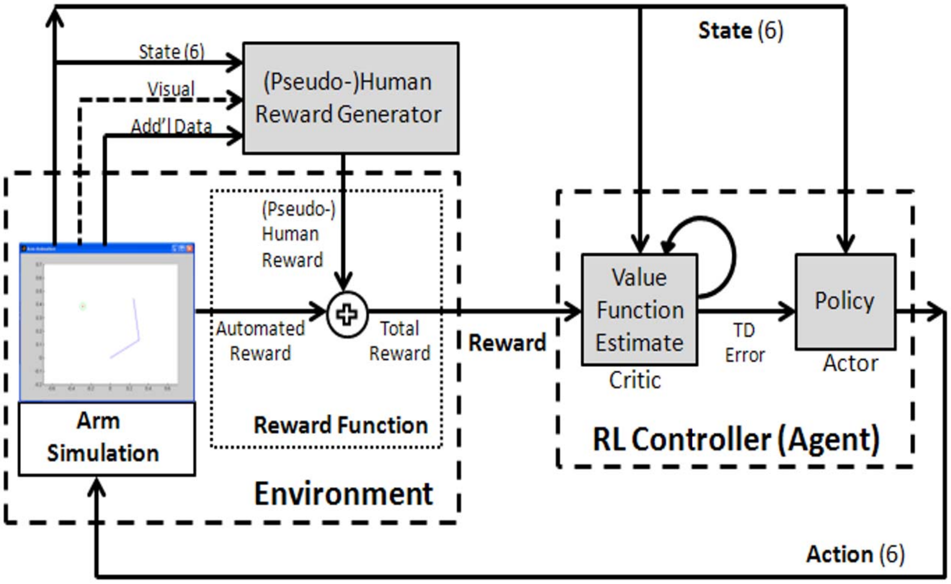


Fig. 2. Block diagram of the actor-critic reinforcement learning (RL) controller [19] with human-generated and pseudo-human rewards, to control a simulated planar human arm. TD error is temporal difference error. State variables consist of 2 current-value joint angles, 2 target-value joint angles, and 2 angular velocities.

The critic generates Temporal Difference (TD) error after the actor applies actions to the environment; this TD error is used to update the actor and the critic (Algorithm 1).

The actor's weights were initialized to match the policy conforming to a proportional-derivative (PD) controller optimized via the simulated annealing algorithm [44] for this 2-joint arm system as described in [13]. The critic's weights were initialized so that they were consistent with the initial actor. Neural networks [21] were used to implement these initializations. For additional details of the implementation of this controller, refer to [12].

A graphical user interface (GUI) (Fig. 9) that permitted human rewards to be integrated with the RL controller was designed using Graphical User Interface Development Environment (GUIDE) software (The MathWorks, Inc. Natick, Massachusetts), and supporting functionality was provided by MATLAB and Simulink (The MathWorks, Inc.). Our design of the arm animation GUI (Fig. 9) was intentionally simple to avoid complications that might have arisen from a more detailed representation of the arm and hand; for example, had a detailed hand with fingers been illustrated, subjects might have become confused about which part(s) of the hand were required to be inside or near the target in order to score the reaching movement positively. With our simple representation of the hand as a dot of the same size as the target dot, we avoided this issue.

Fig. 2 shows the block diagram of the system. The agent calculates the set of 6 continuous muscle stimulation values (ranging from 0.0 to 1.0) to be applied to the arm model at the current 20 ms timestep, and after this action has been applied, the arm model updates its states (joint angles, angular velocities, and target joint angles; target angular velocities are specified to be constantly 0.0). State information is used to update the actor and critic components of the RL controller, as well as the arm animation viewed by human subjects. At the conclusion of each 2-s reaching movement episode, the human subject rated the quality of the viewed reaching movement using the GUI, specifying reward values by manually clicking a computer mouse.

B. Experimental Protocol

This study used 10 adult human subjects (7 males, 3 females) under the age of 40 years who had no neurological or visual impairments that limited hand movement or visual information processing; all had normal or corrected-to-normal vision. Human experimentation was approved under the MetroHealth Medical Center IRB protocol #IRB10-00126.

Each subject participated in 5 RL controller training sessions. Each session involved training an RL controller over 500 episodes of simulated goal-oriented reaching movements; each episode consisted of rating one animated reaching movement. Each 500-episode training session required approximately 1 hour to perform. The 500-task training task set consisted of a set of 50 unique randomly-generated tasks repeated 10 times; each set of 50 tasks was always performed in the same order. Each movement allowed both joints to range from [20.0° , 90.0°]. The mean linear distance between the initial and target hand positions over the 50-task set was 32.04 ± 15.97 cm. For each episode, the human subject viewed on a computer monitor the simulated reaching task performed by the animated planar arm, and after the task had completed, the subject was instructed to assign a reward based on his or her subjective assessment of the quality of the movement that had been performed. Subjects were allotted 4 s between tasks to permit adequate time to assign rewards. Permissible reward values were integers in the range of [−2, +2]. Preliminary experiments [21] had applied

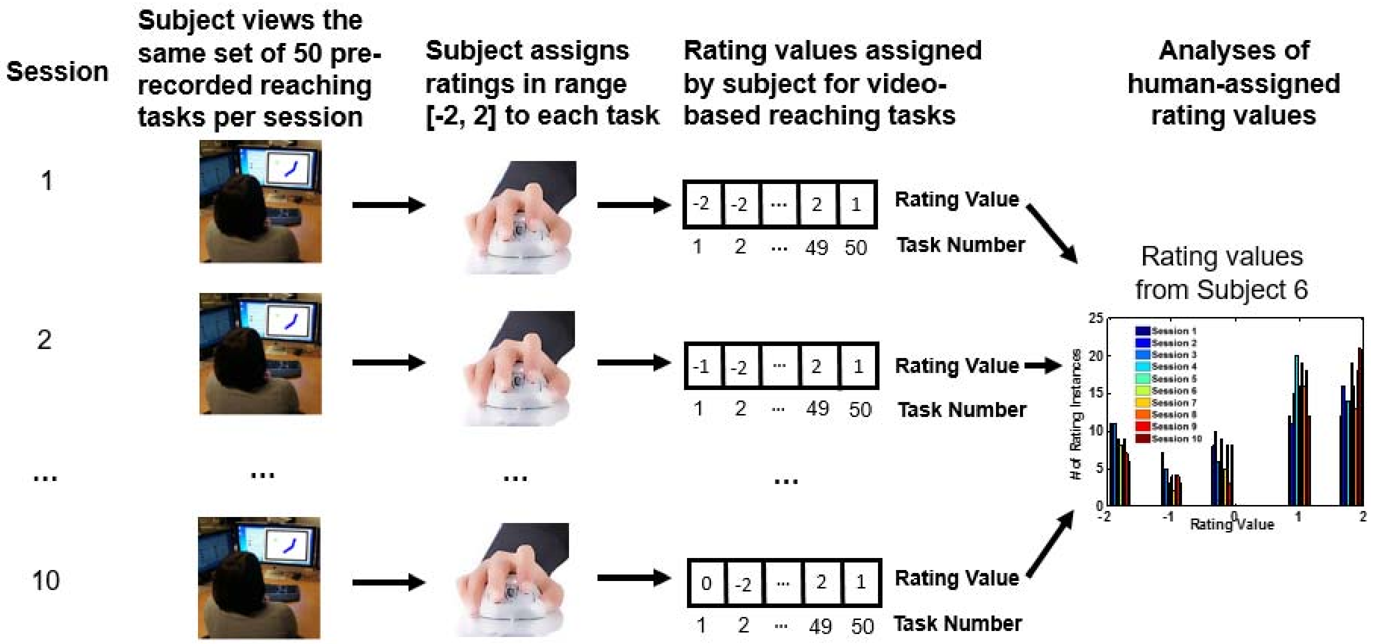


Fig. 3. Design of 50-task video animation rating experiment. The design displayed was used for each of 10 human subjects; sample data and results from Subject 6 (representing typical reward assignment consistency behavior) are shown.

3, 5, or 7 discrete reward levels to determine which yielded the most useful learning of the actor-critic RL controller. The controller's learning measurably improved when 5 discrete reward levels were used vs. only 3, but performance between the 5- and 7-level rewards systems was so similar that it was decided that using 5 reward levels would be sufficient. Additionally, we took into account the consideration of how many levels of discrete reward would be feasible for human subjects to effectively assign. Using 5 discrete reward levels corresponds to intuitive interpretations of Very Poor, Poor, OK, Good, and Very Good ratings, whereas using a higher number of levels might introduce a delay in the provision of human-generated feedback rewards, should subjects become confused about the interpretation of finer gradations of reward level. For each subject, the trained controller resulting from one session was used as the initial controller for the subsequent session.

Before their first session, human subjects were advised of a list of criteria on which they might consider rating tasks. However, subjects were free to select their own rating systems based on their subjective assessments of each individual task performed.

In order to assess the consistency of human reward-giving for a set of dynamic tasks that was invariant (i.e., did not change across sessions due to RL controller learning), a video recording of 50 unique, randomly-generated movements controlled by an actor-critic RL controller was created. These recorded reaching tasks varied significantly with respect to the movement being performed as well as the qualitative movement characteristics. During each of 5 data collection sessions, each human subject viewed and rated the 50-task video (Fig. 3) once at the beginning of the session, and again at the end of the session, with the 500-episode RL controller training run occurring between these two video rating runs.

C. Automated Rewards

To allow comparison with the collected human-rewards data, two additional sets of data were collected using computer-generated rewards to train RL controllers. Both of these data sets involved collecting 10 runs of 5 sequential 500-episode sessions per condition. The first data set used automated rewards only:

$$r_{Automated}(t) = W \sum_{i} u_i^2 - d((x, y), (x_{Goal}, y_{Goal})) \quad (4)$$

where $W = -0.016$ [21] was selected to match the reward function employed in the optimized PD controller [13] used for the initial policy, u is a vector of 6 muscle stimulations, $d(\,)$ is Euclidean distance, and (x,y) is the current hand position, calculated according to:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} L_1 \cos(\theta_{sh}) + L_2 \cos(\theta_{sh} + \theta_{elb}) \\ L_1 \sin(\theta_{sh}) + L_2 \sin(\theta_{sh} + \theta_{elb}) \end{bmatrix} \quad (5)$$

where θ_{sh} is shoulder angle, θ_{elb} is elbow angle, L_1 is length of the upper arm, L_2 is length of the forearm, and both segments were assumed to have identical lengths. The target hand position is denoted (x_{Goal}, y_{Goal}), and is calculated from (5), using the target shoulder and elbow joint angles.

D. Human-Generated and Pseudo-Human Rewards

The second form of computer-generated rewards added pseudo-human rewards, which were generated by Algorithm 2, and assigned once per episode, at the final timestep. For this pseudo-human rewards case, 5 discrete reward levels were used, analogous to the human data collected. In this algorithm, final-timestep pseudo-human rewards are assigned using a combination of success at remaining within the target zone and extent of target overshoot. A detailed description is included below this algorithm.

Algorithm 2 Assign Pseudo-Human Rewards

```
Uppeer OvershootThreshold = 0.1
LowerOvershootThrehold = 0.2
At final timestep of each movement:
if AtTarget and InDwellState
and (MaxOvershoot < UpperOvershootThreshold) then
    FinalTimestepReward = 2
else if AtTarget and InDwellState then
    FinalTimestepReward = 1
else if NotAtTarget and ReachedTargetButExited
and (maxOvershoot < lowerOvershootThresho1d) then
    FinalTimestepReward = −1
else if (MaxOvershoot > LowerOvershootThreshold)
then
    FinalTimestepReward = −2
else FinalTimestepReward = 0
end if
```

Final TimestepReward is the variable being assigned. For each reaching movement, maxOvershoot is the maximum Euclidean overshoot of the target position produced by the controller. UpperOvershootThreshold is a constant value selected by preliminary experiments that specified the maximum amount of target overshoot that was permitted to occur during a 2-second reaching episode and still allow the reaching task to be assigned a high pseudo-human reward value. LowerOvershootThreshold is a constant value selected by preliminary experiments that specified the maximum amount of target overshoot that was permitted to occur during a 2-second reaching episode and still allow the reaching task to be assigned a particular, non-maximal level of pseudo-human reward value. AtTarget is a binary value defined based on whether the hand was within a distance of 0.075 distance units from the target, determined experimentally. This target zone is shown as a ring around the target dot in Fig. 9. NotAtTarget is a binary-valued state indicating whether the hand was within an experimentally-selected value of 0.075 distance units from the target. InDwellState is a binary value indicating whether the hand had been located within the target zone for at least 100 ms (5 timesteps) consecutively. MaxOvershoot is the maximum Euclidean overshoot of the target position produced by the controller. Final TimestepReward is the reward assigned at the final timestep of each episode. ReachedTargetButExited is a binary-valued state indicating whether the hand passed through the target zone and subsequently exited this zone.

In the system block diagram (Fig. 2), the (Pseudo-)Human Reward Generator block is the source of the reward that is added to the automated reward [12], according to:

$$r_{Total} = r_{Automated} + \nu r_{(Pseudo-)Human} \tag{6}$$

where r_{Total} is the total reward, $r_{Automated}$ is the reward calculated from the arm model, $r_{(Pseudo-)Human}$ is the reward generated from either human or pseudo-human sources at the final timestep of each episode (i.e. reaching movement), and ν is a constant weighting factor. The value of ν was selected to be 20.0 as a result of preliminary experiments so that human-generated or pseudo-human rewards would have a substantial impact on learning while still allowing the automated rewards to serve as a baseline component of the reward that permits moderate levels of learning that are able to be improved.

E. Performance Metrics

The recorded performance metrics were defined as follows:

1) Dwell-At-Target Success: At the final timestep of each reaching movement episode, the arm model's hand position was evaluated relative to the target position. If the hand fell within the specified target zone, and had continuously remained within this target zone for at least 100 ms, the episode was scored as a success. Otherwise, the episode was counted as a failure. Because this metric is a binary value, success was measured over groups of episodes: success percentages were recorded over the set of 500 episodes performed per session, as well as over each 100-episode subset.

2) Target Overshoot: One overshoot value was recorded for each reaching movement episode as the largest Euclidean distance from the target position traversed by the hand, subsequent to the hand entering the target zone. Mean overshoots were calculated over each set of episodes. Episodes in which the target zone was not reached were necessarily excluded from reported overshoot values.

3) Mean Rewards: For the human-generated and pseudo-human reward conditions, each reaching movement episode involved the assignment of an integer reward value ranging from −2 to +2. The mean reward was calculated as the average reward value over each 500-episode session, as well as over 100-episode subsets of this data set.

4) Positive:Negative Rewards Ratio: The final-timestep reward, generated by human subjects or the pseudo-human rewards-generation algorithm (Algorithm 2), was recorded for each training episode. Positive rewards were defined as the sum of the counts of +1 and +2 rewards, and negative rewards constituted the set of all −1 and −2 rewards. Ratios of the sum of positive reward instances divided by the sum of negative reward instances were calculated for selected subsets of the collected data.

F. Trained Controller Testing

After the RL controllers had been trained, each was tested on a set of 500 unique randomly-generated tasks ranging from [20.0° , 90.0°] for both joint angles. None of the testing tasks had previously been used to train the controllers. The mean linear distance between the initial and target hand positions over the 500-task testing set was 28.38 ± 16.88 cm. The arm model continued to have its three flexor muscles weakened by 50% of their maximum force. RL controller learning was turned off during this testing stage, so that each controller could perform a single set of the 500 tasks in a deterministic

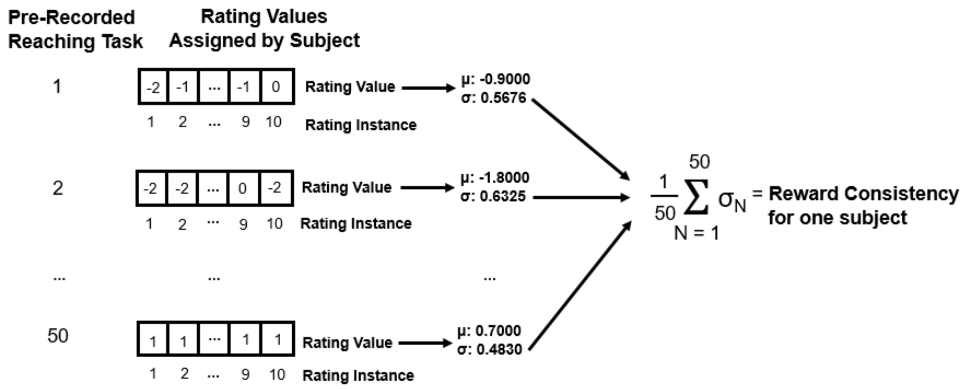


Fig. 4. Calculation of reward consistency value for a single subject; data for Subject 6, representing typical reward assignment consistency among the 10 subjects, is shown. μ is mean, σ is standard deviation.

process. An optimized PD controller [13] was also applied to this set of 500 tasks.

G. Data Analysis

Data were inspected for adherence to standard statistical assumptions (e.g., normality, linearity, homoscedasticity) and alternative analyses were conducted when assumptions were violated. The relative performance of RL controllers trained using human-generated rewards against controllers trained using pseudo-human computer-generated rewards and automated rewards was assessed via pairwise t-tests that were corrected for multiple comparisons using false discovery rate (FDR) [48]. These tests were performed on the dwell-at-target success metric of the controller training data over the final 100 episodes. To compare the rewards assigned by human subjects against those assigned by the pseudo-human algorithm, Welch's t-test was used. The ratios of positive:negative rewards for both conditions over Session 1 and Session 5 were calculated. To test whether human reward signals significantly improve accuracy in arm movement, linear mixed modeling was used, and the slope of the human-rewards condition was tested to determine whether it differed significantly from zero.

To determine the relationship between human reward-assignment consistency and dwell-at-target success of the RL controllers trained using human rewards, it was necessary to calculate this consistency value (Fig. 4); this quantity was calculated in the following way. For the video rating experiment, each subject had rated each of 50 tasks ten times. For each of the 50 tasks, the mean and standard deviation over the subject's 10 rating values for that task were calculated. The 50 standard deviation values corresponding to each subject's 50 rated tasks were averaged, and this mean standard deviation value was used as the Reward Consistency value for each subject. For the dwell-at-target success data set, each human-trained controller's success percentage over the final 100 episodes of Session 5 was used. Spearman's ρ was calculated to compare the human reward consistency and dwell-at-target success data sets for each human subject.

For analysis of trained controller testing, Kolmogorov-Smirnov analysis was applied to compare RL controllers trained using human and pseudo-human rewards for the metrics of dwell-at-target success, time to achieve the dwell state, and target overshoot.

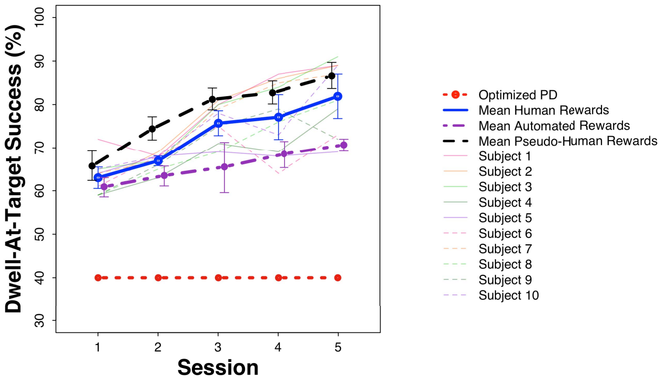


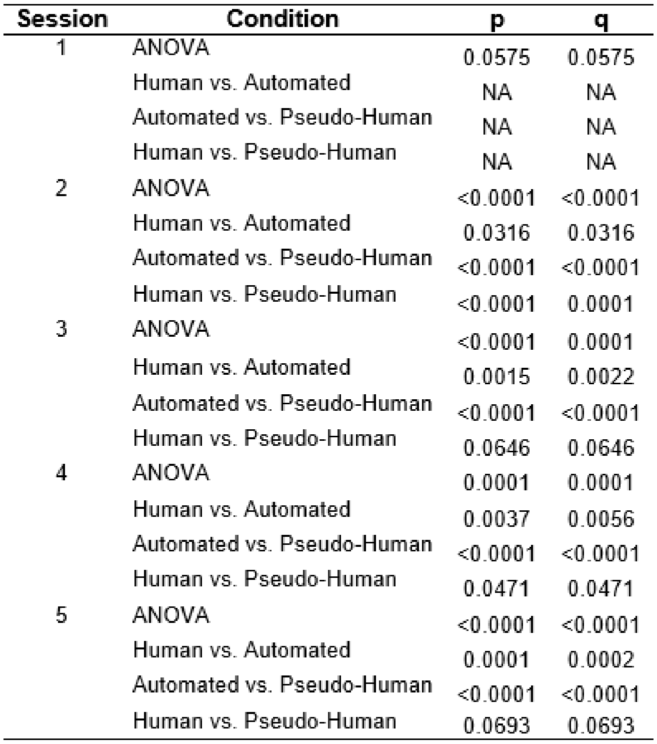
Fig. 5. Dwell-at-target success over the final 100 episodes of each of five 500-episode reinforcement learning (RL) controller training sessions. Learning resulting from 10 individual human subjects are shown as thin trendlines, and the mean human dwell-at-target success is shown as the thick solid blue trendline. All standard error bars represent 95% confidence intervals with averages over 10 runs of the same controller training condition in the center; the human data set (thick solid blue trendline) is averaged over all 10 human subjects. Means and confidence intervals have been offset on the x-axis between the conditions for visual clarity. Controllers trained using human rewards significantly outperformed those trained using automated rewards starting at Session 2 ($p = 0.03$, FDR-adjusted $q = 0.03$), and maintained this advantage over the remaining sessions. In Session 5, pairwise t-tests showed significant differences between automated rewards and both the human-generated ($p = 0.0001$, FDR-adjusted $q = 0.0002$) and pseudo-human rewards training conditions ($p < 0.0001$, FDR-adjusted $q < 0.0001$). No significant difference ($p = 0.07$, FDR-adjusted $q = 0.07$) was observed between the human-generated and pseudo-human rewards conditions for the final session's dwell-at-target success values.

III. Results

In this section, we compare the performance of RL controllers trained using human-generated rewards with the performance of RL controllers trained using computer-generated pseudo-human rewards and computer-generated automated rewards, for dynamic goal-oriented reaching movements using our arm model. A benchmark optimized PD controller [13] is also used for comparison.

TABLE I

P VALUES (AND CORRESPONDING FDR-CORRECTED *q* VALUES) FOR PAIR-WISE COMPARISONS FOR EACH CONTROLLER TRAINING REWARD CONDITION ACROSS EACH OF THE FIVE SESSIONS FOR THE DWELL-AT-TARGET SUCCESS PERCENTAGES FOR THE FINAL 100 EPISODES OF EACH SESSION (FIG. 5)



Session	Condition	p	q
1	ANOVA	0.0575	0.0575
	Human vs. Automated	NA	NA
	Automated vs. Pseudo-Human	NA	NA
	Human vs. Pseudo-Human	NA	NA
2	ANOVA	<0.0001	<0.0001
	Human vs. Automated	0.0316	0.0316
	Automated vs. Pseudo-Human	<0.0001	<0.0001
	Human vs. Pseudo-Human	<0.0001	0.0001
3	ANOVA	<0.0001	0.0001
	Human vs. Automated	0.0015	0.0022
	Automated vs. Pseudo-Human	<0.0001	<0.0001
	Human vs. Pseudo-Human	0.0646	0.0646
4	ANOVA	0.0001	0.0001
	Human vs. Automated	0.0037	0.0056
	Automated vs. Pseudo-Human	<0.0001	<0.0001
	Human vs. Pseudo-Human	0.0471	0.0471
5	ANOVA	<0.0001	<0.0001
	Human vs. Automated	0.0001	0.0002
	Automated vs. Pseudo-Human	<0.0001	<0.0001
	Human vs. Pseudo-Human	0.0693	0.0693

A. Dwell-at-Target Success

Dwell-at-target success percentages over the final 100 episodes of each of the five sessions of RL controller training are presented in Fig. 5. Error bars show 95% confidence intervals averaged over 10 runs; the solid blue trendline averages all 10 human subjects' data. The red dotted trendline indicates the optimized PD controller's performance on the same set of 50 unique tasks on which the RL controllers were trained. All RL controllers significantly outperformed the PD controller (which had a mean success rate of 40%), and all RL controllers improved their dwell-at-target success rates over the course of the five 500-episode sessions. In particular, the mixed model analysis showed that human rewards significantly improved accuracy from trial to trial (prediction equation: accuracy = 64.1 + 4.1 x Trial; χ^2 = 70.398, $p < 0.0001$). Phrased differently, human reward signals did improve the accuracy in arm movement beyond what would be expected by chance, and their improvement occurred at a constant rate of approximately 4.1% each trial.[1]

We analyzed dwell-at-target success (Fig. 5) of the final 100 episodes of each training session, to emphasize the performance that controllers had achieved toward the end of each session. The superior dwell-at-target success of controllers trained using pseudo-human rewards when compared with controllers trained using automated rewards became detectable from Session 2 onward ($p < 0.0001$, FDR-adjusted $q < 0.0001$; see Table I). Furthermore, controllers trained using human rewards also began to significantly outperform those trained using automated rewards over the final 100 episodes starting at Session 2 ($p = 0.03$, FDR-adjusted $q = 0.03$), and maintained this advantage over the remaining sessions. For dwell-at-target success of the final 100 episodes of Session 5, pairwise t-tests showed significant differences between the automated rewards condition and both the human-generated ($p = 0.0001$, FDR-adjusted $q = 0.0002$) and pseudo-human rewards training conditions ($p < 0.0001$, FDR-adjusted $q < 0.0001$). No significant difference ($p = 0.07$, FDR-adjusted $q = 0.07$) was observed between the human-generated and pseudo-human rewards conditions for the final 100 episodes of the final session's dwell-at-target success values.

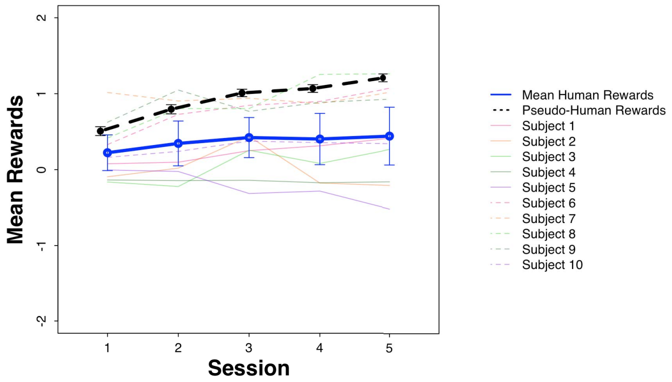


Fig. 6. Mean rewards over five 500-episode reinforcement learning (RL) controller training sessions. Rewards resulting from 10 individual human subjects are shown as thin trendlines, and the mean human reward value is shown as the thick solid blue trendline. All standard error bars represent 95% confidence intervals with averages over 10 runs of the same controller training condition in the center; the human data set (thick solid blue trendline) is averaged over all 10 human subjects. Means and confidence intervals have been offset on the x-axis between the conditions for visual clarity. Pseudo-human rewards are significantly more positive than human-generated rewards for all 5 sessions (Table II).

B. Final-Timestep Rewards

1) Reward Trends Across Sessions:

Mean rewards assigned by human subjects and by the computer-generated pseudo-human algorithm (Algorithm 2) across the five training sessions are presented in Fig. 6. The mean rewards given by each human subject appear to be fairly consistent across all episodes; individual trendlines (thin lines) do not significantly cross others or show dramatic shifts as sessions progress. Human rewards visually fall into two distinct groups: the more-positive group, consisting of Subjects 6, 7, 8, and 9; and the less-positive group, consisting of the remaining six subjects. The more-positive group tended to have a larger positive reward increase over the 5 sessions (mean difference between the mean rewards in Session 5 and in Session 1 was 0.47 ± 0.39, mean ± s.d.), compared with the less-positive group (0.05 ± 0.34). Pseudo-human computer-generated rewards (thick black dashed trendline) increased monotonically across the sessions, while the mean

[1] A squared term was also added to the model to account for possible non-linearities in the rate of improvement. This model was not significantly different than the model without the squared term ($\chi^2 = 1.799$, $p = 0.1799$).

TABLE II

PAIRWISE t, p, AND FDR-CORRECTED q VALUES FOR THE MEAN REWARDS VARIABLE (FIG. 6) COMPARED BETWEEN HUMAN-GENERATED AND PSEUDO-HUMAN-GENERATED CONDITIONS

Session	t	p	q
1	2.30	0.044	0.044
2	2.93	0.015	0.019
3	4.29	0.002	0.006
4	3.82	0.004	0.006
5	3.92	0.003	0.006

human final-timestep rewards increased only modestly and non-monotonically across sessions. The standard deviations of the pseudo-human rewards were substantially smaller than those of the human-generated rewards.

Because variances between the two groups were not equal, Welch's t-tests were performed for each session separately and their p-values adjusted for multiple comparisons using false discovery rate [48] (FDR; see Table II). The rewards assigned by the pseudo-human algorithm were found to be significantly more positive than the rewards generated by human subjects for every training session (see Table II). When rewards over all sessions were grouped into Positive (+1 and +2 values) and Negative (−1 and −2 values) categories, 2 of 10 human subjects (Subjects 4 and 5) had net-negative rewards, and the remaining 8 subjects showed net-positive rewards. Also, when the mean reward for the first (Session 1) and last (Session 5) training sessions were compared for each human subject, and the difference of the mean rewards between the final and initial sessions was calculated, 3 of the 10 subjects showed rewards that became more negative over time, while the remaining 70% of subjects showed increased positivity of rewards over time.

2) Positive:Negative Reward Ratios: Fig. 7 presents the positive:negative reward ratios for human-generated rewards and computer-generated pseudo-human rewards for the initial (Session 1: Fig. 7(a)) and final (Session 5: Fig. 7(b)) data collection sessions. The black dashed horizontal lines separate net-positive (i.e. positive:negative reward ratios > 1) and net-negative (i.e. ratios < 1) reward ratios. Human reward ratios are blue when net-positive, and red when net-negative. From Session 1 to Session 5, all 10 of the pseudo-human reward ratios became dramatically more positive. In contrast, only 3 of the 10 human subjects (Subjects 6, 8, and 9) show ratios that substantially increased from Session 1 to Session 5, and 4 subjects (Subjects 2, 4, 5, and 7) had reward ratios that decreased over this period.

C. Success as a Function of Reward Consistency

Spearman's ρ was calculated to relate RL controller learning success over the final 100 episodes of Session 5 to human-generated reward consistency over the 50-task video rating data set. No correlation was found between dwell-at-target success and rating consistency: $\rho = 0.0307$, $N = 10$, $p = 0.933$.

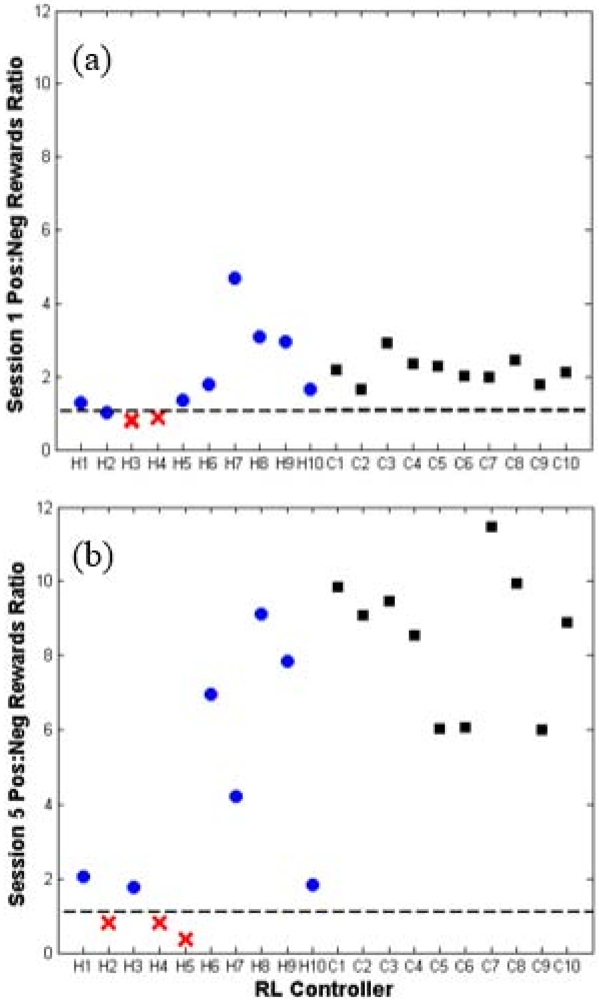


Fig. 7. Positive:negative final-timestep reward ratio for the initial (Session 1: Panel (a)) and final (Session 5: Panel (b)) data collection sessions of reinforcement learning (RL) controller training. H1 – H10 denote reward ratios from human subjects 1 – 10; C1 – C10 denote reward ratios from computer-generated pseudo-human reward algorithm (Algorithm 2). For human reward ratios, blue dots indicate net-positive reward ratios, and red x markers indicate net-negative ratios. Black dashed line shows the division between net-positive and net-negative reward ratios.

D. Trained Controller Testing

Fig. 8 shows the results of applying the RL controllers trained using either human-generated or pseudo-human rewards to the set of 500 randomly generated testing tasks. Boxplots of the dwell-at-target success percentages are given in Fig. 8(a). Both sets of RL controllers showed high success rates (98.26 ± 1.18% for the human-trained controllers, 99.20 ± 0.41% success for the pseudo-human-trained controllers), indicating that both forms of training allowed the hand to reach the target zone and remain there for nearly every movement tested. The human-trained controllers show slightly more variability than controllers trained using pseudo-human rewards. Optimized PD controller performance is given by the red dashed line, and is noticeably less successful (43.0% success) than either of the sets of RL controllers. Kolmogorov-Smirnov analysis revealed a significant difference between the human and pseudo-human rewards conditions: pseudo-human rewards outperformed human rewards ($D = 0.60$; $p = 0.03$).

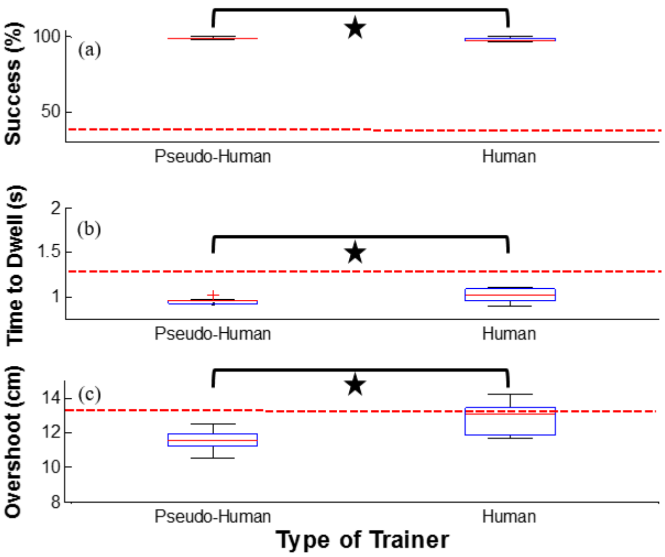


Fig. 8. Testing of reinforcement learning (RL) controllers trained using computer-generated pseudo-human and human-generated rewards. Boxplots show 10 sets of 500 tested tasks per reward condition (10 controllers trained by using pseudo-human rewards and 10 controllers trained by human subjects; 1 controller per subject). Stars indicate statistically significant differences between the two training conditions. (a) Dwell-at-target success percentages. (98.26 ± 1.18% for the human-trained controllers, 99.20 ± 0.41% for the pseudo-human-trained controllers). Kolmogorov-Smirnov analysis revealed a significant difference between the human and pseudo-human rewards conditions: pseudo-human rewards outperformed human rewards (D = 0.60; p =0.03). (b) Mean time (in seconds) required to achieve the dwell state (1.01 ± 0.07 s vs. 0.95 ± 0.02 s for human-trained vs. pseudo-human reward-trained). Kolmogorov-Smirnov analysis showed a significant difference between the two cases (D =0.60; p = 0.03), with the pseudo-human rewards condition achieving the dwell state more quickly. (c) Mean overshoot of target (in cm): 12.83 ± 0.92 cm for human-trained controllers vs. 11.58 ± 0.63 cm for pseudo-human reward-trained controllers. The Kolmogorov-Smirnov test showed a significant difference between the two reward conditions, with the pseudo-human rewards having smaller overshoot than human rewards ($D = 0.60$; $p = 0.03$). In these plots, the red central line indicates the median; upper and lower limits of the blue boxes indicate the upper and lower quartiles, respectively; and the black horizontal lines above and below each box indicate the maximum and minimum values, respectively.

Fig. 8(b) presents the mean time required to achieve the dwell state for both training conditions. Human-trained controllers had slightly larger (i.e., slower) mean times to achieve the dwell state, compared with the controllers trained using pseudo-human rewards (1.01 ± 0.07 s vs. 0.95 ± 0.02 s, respectively). Kolmogorov-Smirnov analysis showed a significant difference between the two cases ($D = 0.60$; $p = 0.03$), with the pseudo-human rewards condition achieving the dwell state more quickly. The optimized PD controller, given by the red dashed trendline (1.36 s), had a notably larger mean time to dwell value than either of the RL controller sets.

Fig. 8(c) gives the mean target overshoot of both controller training conditions. The human-trained RL controllers had a larger mean overshoot (12.83 ± 0.92 cm) than did the controllers trained with pseudo-human rewards (11.58 ± 0.63 cm), although both sets of RL controllers had smaller mean overshoot values than that of the optimized PD controller (12.88 cm) for this set of 500 tasks. The Kolmogorov-Smirnov test showed a significant difference between the two reward conditions, with the controllers trained using pseudo-human rewards having smaller overshoot than controllers trained using human rewards ($D = 0.60$; $p = 0.03$).

IV. Discussion

In the experiments described, ten neurologically intact human subjects trained actor-critic RL controllers to perform planar goal-oriented reaching movements using a biomechanical human arm model, by assigning a subjective reward to each animated arm reaching movement performed by the controller. Additionally, pseudo-human computer-generated rewards (Algorithm 2) were used to train RL controllers, as was an automated rewards training condition that used only rewards provided by the arm model (Fig. 1). A benchmark optimized PD controller was also applied to the sets of training and testing tasks used for RL controller analysis.

We found that all three forms of training rewards allowed the RL controllers to significantly outperform the optimized PD controller for the dwell-at-target success performance metric. The deterministic PD controller was not able to adapt to a new control strategy necessitated by the arm model in which the flexor muscles had been weakened significantly. In contrast, all RL controllers were able to improve their performance progressively to adapt to this weakened arm model. All three forms of RL controller training rewards accomplished measurable improvement in dwell-at-target success over five sessions of 500 episodes per session.

Both human-generated and pseudo-human rewards yielded significantly improved dwell-at-target success when compared with the performance of RL controllers trained using automated rewards. The mean dwell-at-target success values of the controllers trained using pseudo-human rewards (Fig. 5, thick black dashed trendline) were visibly somewhat larger than those of controllers trained by human subjects (Fig. 5, thick blue solid trendline) for all sessions; however, there was no statistically significant difference between the RL controllers trained using human-generated and pseudo-human rewards when the final 100 episodes of the final data collection session were evaluated, which is the most useful metric of overall learning success.

Trained RL controllers tested on tasks that they had not previously encountered showed a small but statistically significant advantage of training using pseudo-human rewards instead of human-generated rewards for the performance metrics of dwell-at-target success, time to achieve the dwell state, and target overshoot (Fig. 8). Controllers trained using human-generated rewards and controllers trained using pseudo-human rewards each outperformed an optimized PD controller on all three performance metrics (Fig. 8). While the controllers trained using pseudo-human rewards had marginally better performance than controllers trained using human-generated rewards on all three metrics, the functional difference between the performance of these two training cases was minimal, with both sets of controllers demonstrating excellent performance on all three metrics.

Given the strong similarities in performance when using pseudo-human and human-generated rewards to train the actor-critic RL architecture employed for planar arm movement

(Figs. 5 and 8), we propose the use of pseudo-human rewards to pre-train controllers in simulation to achieve a baseline level of performance. Then, when the controller is introduced to its human FES user, human rewards can be substituted to shape controller performance to the preferences of the individual user. Sequential RL controller training with computer-generated and human-generated rewards has been shown to be possible [10], although careful parameter tuning was found to be essential to the success of such a technique [10]. It will remain to be experimentally determined whether an actor-critic RL architecture will be sufficiently flexible to learn effectively from pre-training with a pseudo-human reward function followed by subsequent training using human-generated rewards, when these two forms of reward have distinctly different properties.

Pseudo-human rewards proved to be consistently and significantly more positive than human-generated rewards (Figs. 6 and 7). The deterministic pseudo-human reward generation algorithm (Algorithm 2) caused rewards to become progressively more positive as controller learning improved, and the hand achieved the dwell-at-target state with increasing success. In contrast, even though the RL controller learning resulting from human training progressed apace with that of the controllers trained using pseudo-human rewards (Fig. 5), human rewards did not consistently become more positive across sessions, and variability in these rewards was much larger than that seen for pseudo-human rewards (Fig. 6). Interestingly, the human reward trendlines visually fell into two distinct groups (Fig. 6, thin trendlines), with 4 of the 10 subjects assigning rewards that had similar positivity to the pseudo-human rewards (thick black trendline), while 6 of the 10 subjects tended to assign less positive rewards, so that the calculated mean human reward values (thick blue solid trendline) were substantially less positive than the pseudo-human rewards. In analyses comparing reward positivity to dwell-at-target success, we found no association between these two quantities. Although larger numbers of subjects would be required to draw strong conclusions, it appears that human trainers may have inherent biases that shape their reward-assignment tendencies. If the positivity of assigned rewards proves not to significantly influence the success of actor-critic RL architecture training, reward positivity may not be a factor that should be included in future RL controller design considerations.

When the human rating consistency values from the animated reaching movement video rating experiment were compared against the RL controller learning achieved by each subject, no correlation was detected. This finding agrees with our previous assessment of controllers trained using human-generated vs. pseudo-human rewards: the pseudo-human rewards were, by definition, 100% self-consistent, because they were generated from Algorithm 2. This algorithm used only a few quantitative metrics to evaluate controller performance. In contrast, the human subjects were able to assess a much wider range of both quantitative and qualitative performance characteristics when selecting their reward values. As a result, human-generated rewards were much more variable. From the dwell-at-target success analysis of the data in Fig. 5 and associated statistics, we found that, even though the pseudo-human rewards were much more consistent than human-generated rewards (Fig. 6), the ultimate dwell-at-target success for both forms of training were strongly similar. This suggests that the actor-critic RL control architecture implemented for this modeled human arm domain does not appear to be sensitive to reward consistency, and that as long as the reward signals contain useful information, this controller can learn efficiently even when inconsistent, unpredictable human rewards are used.

We analyzed the effects of using both human-generated and pseudo-human rewards on the speed of RL controller learning, as measured by dwell-at-target success percentages across sessions (Fig. 5). The controllers trained using pseudo-human rewards demonstrated a significant and consistent learning speed advantage over those trained using automated rewards, starting in Session 2. The advantage of using human-generated rewards was not as pronounced, first becoming substantial in Session 3. By Session 5, the controllers trained using both human-generated and pseudo-human rewards showed a significant advantage over controllers trained using automated rewards, with the success values of the human-trained and pseudo-human controllers being statistically indistinguishable.

We conclude that training controllers using our pseudo-human reward-generation algorithm (Algorithm 2) shows a moderate advantage in learning speed over using human-generated rewards, which in turn achieves learning more quickly and efficiently than the automated-rewards condition. This finding conflicts with previous work showing an advantage in learning speed when human-generated rewards are used to train RL controllers, compared with computer-generated rewards [23], although our domain differs from that of previous work, and the superiority in learning speed of our pseudo-human rewards-generation algorithm over controllers trained using human-generated rewards was only a modest effect.

Previous experiments using humans to train RL controllers have consistently identified a strong positive bias in human reward assignment [10], [24]. Comparing these observations against our results (Figs. 6 and 7), we observe that 2 of the 10 human trainers (Subjects 4 and 5) had net-negative reward values for most of the training sessions. We posit that a combination of task domain and psychological traits of the human trainer yield observable trends in controller training behavior, and that human-generated rewards will not necessarily always have a positive bias. On the whole, however, for this study in which RL controllers learned from interactions with their human trainers, over the entire set of 2,500 training episodes completed by each subject, most (8 of 10) human trainers did assign rewards with a positive bias, to a greater or lesser degree, consistent with previous findings in the literature. Also, our experiments revealed that human-generated rewards varied in their internal consistency. This finding is in accordance with previous work showing that the human reward function is dynamic and inconsistent [24], [25].

The pseudo-human reward generation algorithm (Algorithm 2) was developed with the goal of achieving a balance among the 5 possible reward levels over a

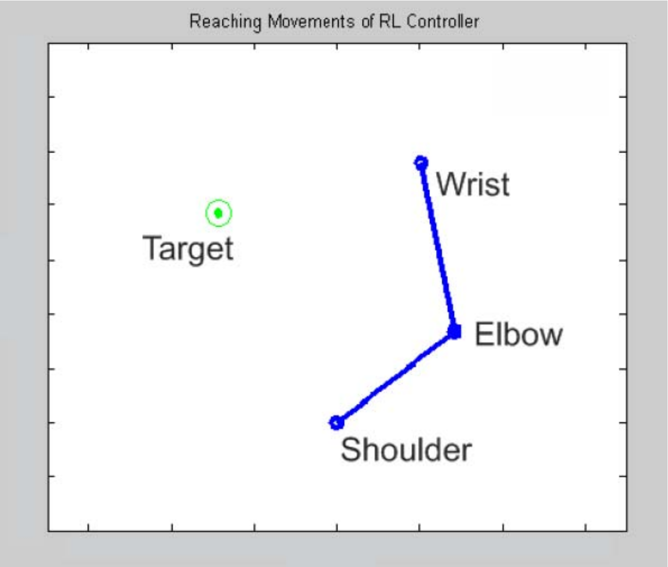


Fig. 9. Screenshot of the arm visualization GUI used in experiments. The green dot represents the target, and the green ring indicates the target zone, which was not displayed during experiments involving human-generated rewards. For each episode, the dot representing the hand (adjacent to the wrist) had the goal of reaching and remaining at the target dot, for a variety of different tasks. Note that the text labels in this figure were not present during the testing sessions.

preliminary testing set of 500 episodes. However, other criteria may instead or additionally have been used to specify this algorithm, which would have yielded different controller-training behavior. Therefore, it should be recognized that the pseudo-human reward generation described here is one specific formulation of many possible reward generation schemes, and that other algorithms would produce different controller learning properties.

Alternatives to treating feedback as numeric, such as interpreting human rewards depending on both the teaching strategy adopted by the teacher (e.g. the human deciding to withhold rewards as a sign of negative feedback) as well as the task intended to be taught, could potentially improve the rate of learning. For example, Loftin et al. [45] provided two novel Bayesian algorithms to achieve effective learning from human-generated rewards for the contextual bandit [46] problem. Exploring whether such methods will extend effectively to more complex systems such as the arm model used in the present work remains for future investigation.

Ng et al. [47] described the importance of choosing effective shaping functions for human-guided RL control; their work suggests the significance of considering how modifications to the reward functions of Markov decision processes (MDPs) affect the optimal policy. They present strategies to modify the reward function in order to preserve the optimal policy; future exploration of this concept in the context of selecting the most effective method to integrate human rewards with the actor-critic RL controller could be useful.

Despite the many challenges of using sparse, delayed rewards as a training signal, we have shown that actor-critic RL controllers can be trained in a simulated human arm domain using both human-generated and pseudo-human rewards with these properties. In future work, the pre-trained controller should be introduced to human FES users with tetraplegia, and it should be observed how successfully an actor-critic architecture is able to adapt when the shoulder and arm being controlled have properties differing from the simulated arm system on which the controller was trained.

These experiments demonstrate that it is possible for subjects to successfully train RL controllers for a simulated human arm, using subjective human-generated rewards. Pseudo-human rewards, generated from an algorithm, were also used for RL controller training, and were found to result in performance similar to that of controllers trained with human-generated rewards; pseudo-human rewards training yielded a statistically significant advantage over human-trained controllers, although the functional difference between the two forms of training was minimal. Even though the rewards used for training were delayed, sparse, and inconsistent, the actor-critic RL architecture was able to learn effectively from them. Significant learning is observable over as few as 500 training episodes, with learning progressing consistently over the five training sessions performed. We recommend that pseudo-human computer-generated rewards be used for controller pre-training in simulated environments before introduction to human Functional Electrical Stimulation (FES) systems, in which they can be trained using human-generated rewards in addition to computer-generated rewards. These results serve as a proof of concept that human rewards are a viable training signal for RL control of upper-extremity FES systems.

Acknowledgment

The authors thank Dr. S. Sidik and Dr. D. Fife for their statistical guidance, and they gratefully acknowledge the helpful comments from the three anonymous reviewers of this manuscript.

References

[1] P. H. Peckham and J. S. Knutson, “Functional electrical stimulation for neuromuscular applications,” *Annu. Rev. Biomed. Eng.*, vol. 7, pp. 327–360, Aug. 2005.

[2] K. T. Ragnarsson, “Functional electrical stimulation after spinal cord injury: Current use, therapeutic effects and future directions,” *Spinal Cord*, vol. 46, no. 4, pp. 255–274, 2008.

[3] A. McIntyre *et al.*, “Examining the effectiveness of intrathecal baclofen on spasticity in individuals with chronic spinal cord injury: A systematic review,” *J. Spinal Cord Med.*, vol. 37, no. 1, pp. 11–18, 2014.

[4] L. A. Harvey and R. D. Herbert, “Muscle stretching for treatment and prevention of contracture in people with spinal cord injury,” *Spinal Cord*, vol. 40, no. 1, pp. 1–9, Jan. 2002.

[5] W. D. Memberg, P. E. Crago, and M. W. Keith, “Restoration of elbow extension via functional electrical stimulation in individuals with tetraplegia,” *J. Rehabilitation Res. Develop.*, vol. 40, no. 6, pp. 477–486, Nov./Dec. 2003.

[6] M. A. Lemay and P. E. Crago, “Closed-loop wrist stabilization in C4 and C5 tetraplegia,” *IEEE Trans. Rehabil. Eng.*, vol. 5, no. 3, pp. 244–252, Sep. 1997.

[7] P. E. Crago, R. J. Nakai, and H. J. Chizeck, “Feedback regulation of hand grasp opening and contact force during stimulation of paralyzed muscle,” *IEEE Trans. Biomed. Eng.*, vol. 38, no. 1, pp. 17–28, Jan. 1991.

[8] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[9] L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: A survey,” *J. Artif. Intell. Res.*, vol. 4, no. 1, pp. 237–285, Jan. 1996.

[10] W. B. Knox, “Learning from human-generated reward,” Ph.D. dissertation, Dept. Electr. Comput. Eng., Univ. Texas Austin, Austin, TX, USA, 2012.

[11] W. B. Knox and P. Stone, "Reinforcement learning from simultaneous human and MDP reward," in *Proc. 11th Int. Conf. Auto. Agents Multi-agent Syst.*, vol. 1. 2012, pp. 475–482.

[12] K. M. Jagodnik, P. S. Thomas, A. J. van den Bogert, M. S. Branicky, and R. F. Kirsch, "Human-like rewards to train a reinforcement learning controller for planar arm movement," *IEEE Trans. Human-Mach. Syst.*, vol. 46, no. 5, pp. 723–733, May 2016.

[13] K. M. Jagodnik and A. J. van den Bogert, "Optimization and evaluation of a proportional derivative controller for planar arm movement," *J. Biomech.*, vol. 43, no. 6, pp. 1086–1091, 2010.

[14] A. V. Hill, "The heat of shortening and the dynamic constants of muscle," *Proc. R. Soc. Lond. B, Biol. Sci.*, vol. 126, no. 843, pp. 136–195, 1938.

[15] J. M. Winters, "Hill-based muscle models: A systems engineering perspective," in *Multiple Muscle Systems: Biomechanics and Movement Organization*, J. M. Winters and S. L.-Y. Woo, Eds. New York, NY, USA: Springer, 1990, pp. 69–93.

[16] S. McLean, A. Su, and A. van den Bogert, "Development and validation of a 3-D model to predict knee joint loading during dynamic movement," *J. Biomech. Eng.*, vol. 125, no. 6, pp. 864–874, 2003.

[17] C. K. Thomas, E. Y. Zaidner, B. Calancie, J. G. Broton, and B. R. Bigland-Ritchie, "Muscle weakness, paralysis, and atrophy after human cervical spinal cord injury," *Exp. Neurol.*, vol. 148, no. 2, pp. 414–423, 1997.

[18] M. J. Castro, D. F. Apple, Jr., R. S. Staron, G. E. Campos, and G. A. Dudley, "Influence of complete spinal cord injury on skeletal muscle within 6 mo of injury," *J. Appl. Physiol.*, vol. 86, no. 1, pp. 350–358, 1999.

[19] K. Doya, "Reinforcement learning in continuous time and space," *Neural Comput.*, vol. 12, no. 1, pp. 219–245, 2000.

[20] P. S. Thomas, M. Branicky, A. van den Bogert, and K. Jagodnik, "Creating a reinforcement learning controller for functional electrical stimulation of a human arm," in *Proc. Yale Workshop Adapt. Learn. Syst.*, vol. 49326. 2008, pp. 1–6.

[21] P. S. Thomas, "A reinforcement learning controller for functional electrical stimulation of a human arm," M.S. thesis, Dept. Electr. Eng. Comput. Sci., Case Western Res. Univ., Cleveland, OH, USA, 2009.

[22] 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," in *Proc. 21st Innov. Appl. Artif. Intell. Conf.*, Pasadena, CA, USA, Jul. 2009, pp. 165–172.

[23] W. B. Knox, P. Stone, and C. Breazeal, "Learning from feedback on actions past and intended," in *Proc. 7th ACM/IEEE Int. Conf. Human-Robot Int., Late-Breaking Rep. Session*, Mar. 2012.

[24] A. L. Thomaz, "Socially guided machine learning," Ph.D. dissertation, Dept. School Comput. Sci., Univ. Massachusetts, Boston, MA, USA, 2006.

[25] W. B. Knox and P. Stone, "TAMER: Training an agent manually via evaluative reinforcement," in *Proc. IEEE 7th Int. Conf. Develop. Learn.*, Aug. 2008, pp. 292–297.

[26] N. Lan, "Analysis of an optimal control model of multi-joint arm movements," *Biol. Cybern.*, vol. 76, no. 2, pp. 107–117, 1997.

[27] K. M. Jagodnik, "Reinforcement learning and feedback control for high-level upper-extremity neuroprostheses," Ph.D. dissertation, Dept. Biomed. Eng., Case Western Res. Univ., Cleveland, OH, USA, 2014.

[28] W. B. Knox, I. Fasel, and P. Stone, "Design principles for creating human-shapable agents," in *Proc. AAAI Spring Symp. Agents Learn Human Teachers*, 2009, pp. 79–86.

[29] C. Klauer *et al.*, "Feedback control of arm movements using neuro-muscular electrical stimulation (NMES) combined with a lockable, passive exoskeleton for gravity compensation," *Front. Neurosci.*, vol. 8, pp. 44–59, Sep. 2014.

[30] C. T. Freeman, "Upper limb electrical stimulation using input-output linearization and iterative learning control," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 4, pp. 1546–1554, Jul. 2015.

[31] C. T. Freeman *et al.*, "Iterative learning control of FES applied to the upper extremity for rehabilitation," *Control Eng. Pract.*, vol. 17, no. 3, pp. 368–381, 2009.

[32] C. Freeman *et al.*, "FES based rehabilitation of the upper limb using input/output linearization and ILC," in *Proc. IEEE Amer. Control Conf. (ACC)*, Jun. 2012, pp. 4825–4830.

[33] C. T. Freeman *et al.*, "Phase-lead iterative learning control algorithms for functional electrical stimulation-based stroke rehabilitation," *Proc. Inst. Mech. Eng. I, J. Syst. Control Eng.*, vol. 225, no. 6, pp. 850–859, 2011.

[34] C. T. Freeman, E. Rogers, A.-M. Hughes, J. H. Burridge, and K. L. Meadmore, "Iterative learning control in health care: Electrical stimulation and robotic-assisted upper-limb stroke rehabilitation," *IEEE Control Syst.*, vol. 32, no. 1, pp. 18–43, Feb. 2012.

[35] A. M. Hughes *et al.*, "Feasibility of iterative learning control mediated by functional electrical stimulation for reaching after stroke," *Neurorehabilitation Neural Repair*, vol. 23, no. 6, pp. 559–568, 2009.

[36] K. L. Meadmore *et al.*, "Functional electrical stimulation mediated by iterative learning control and 3D robotics reduces motor impairment in chronic stroke," *J. Neuroeng. Rehabil.*, vol. 9, no. 1, pp. 32–42, 2012.

[37] S. Micera *et al.*, "Adaptive fuzzy control of electrically stimulated muscles for arm movements," *Med. Biol. Eng. Comput.*, vol. 37, no. 6, pp. 680–685, 1999.

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

[39] N. A. Vien and W. Ertel, "Reinforcement learning combined with human feedback in continuous state and action spaces," in *Proc. IEEE Int. Conf. Develop. Learn. Epigenetic Robot. (ICDL)*, Nov. 2012, pp. 1–6.

[40] N. A. Vien, W. Ertel, and T. C. Chung, "Learning via human feedback in continuous state and action spaces," *Appl. Intell.*, vol. 39, no. 2, pp. 267–278, 2013.

[41] P. M. Pilarski *et al.*, "Online human training of a myoelectric prosthesis controller via actor-critic reinforcement learning," in *Proc. IEEE Int. Conf. Rehab. Robot. (ICORR)*, vol. 1. Zurich, Switzerland, Jun./Jul. 2011, pp. 134–140.

[42] K. W. Mathewson and P. M. Pilarski, "Simultaneous control and human feedback in the training of a robotic agent with actor-critic reinforcement learning," in *Proc. Interact. Mach. Learn. Workshop IJCAI (IML), Connecting Humans Mach.*, New York, NY, USA, Jul. 2016. [Online]. Available: https://arxiv.org/abs/1606.06979

[43] M. J. Matarić, "Reinforcement learning in the multi-robot domain," *Auton. Robots*, vol. 4, pp. 73–83, Sep. 1997.

[44] W. L. Goffe, "Global optimization of statistical functions with simulated annealing," *J. Econometrics*, vol. 60, pp. 65–99, Apr. 1994.

[45] R. T. Loftin *et al.*, "A strategy-aware technique for learning behaviors from discrete human feedback," in *Proc. AAAI*, 2014, pp. 937–943.

[46] T. Lu *et al.*, "Contextual multi-armed bandits," in *Proc. 13th Int. Conf. Art. Intell. Statist.*, 2010, pp. 485–492.

[47] A. Y. Ng *et al.*, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Proc. ICML*, vol. 99. Jun. 1999, pp. 278–287.

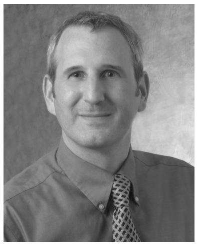
[48] Y. Y. Benjamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *J. R. Statist. Soc. B, Methodol.*, vol. 57, no. 1, pp. 289–300, 1995.

[49] C. Orosy-Fildes and R. W. Allan, "Psychology of computer use: XII. Videogame play: Human reaction time to visual stimuli," *Perceptual Motor Skills*, vol. 69, pp. 243–247, Apr. 1989.

Kathleen M. Jagodnik (M'06) received the Ph.D. degree in biomedical engineering from Case Western Reserve University, Cleveland, OH, USA, in 2014. She is currently a Post-Doctoral Fellow with the Laboratory of Dr. A. Ma'ayan, Department of Pharmacological Sciences, Icahn School of Medicine at Mount Sinai, New York, NY, USA. Her research interests include machine learning, rehabilitation engineering, rehabilitation and spaceflight biomechanics, and computational biology with a focus on drug discovery and methods to study network regulation in mammalian cells.

Philip S. Thomas received the Ph.D. degree in computer science from the University of Massachusetts, Amherst, MA, USA, in 2015. He is a Post-Doctoral Fellow with the Computer Science Department, Carnegie Mellon University, Pittsburgh, PA, USA. His research interests include machine learning, with an emphasis on policy search algorithms.

Antonie J. van den Bogert received the Ph.D. degree from the University of Utrecht. He is currently the Parker–Hannifin Endowed Chair in human motion and control with Cleveland State University. His research interests include musculoskeletal modeling and simulation, optimal control of human movement, and advanced prosthetics and orthotics.



Michael S. Branicky (M'87–SM'01–F'16) received the Sc.D. degree from the Massachusetts Institute of Technology. He is currently the Dean of Engineering and a Professor of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS, USA. His research interests include control systems, robotics, hybrid systems, intelligent control, and learning.

Robert F. Kirsch (M'82) received the Ph.D. degree from Northwestern University. He is currently a Professor and the Chair of the Department of Biomedical Engineering with Case Western Reserve University and the Executive Director of the Cleveland VA Functional Electrical Stimulation (FES) Center. His research involves restoring movement to disabled individuals using FES.