Improving the Learning Rate, Accuracy, and Workspace of Reinforcement Learning Controllers for a Musculoskeletal Model of the Human Arm

Douglas C. Crowder, Jessica Abreu, Robert F. Kirsch [Member, IEEE]
Department of Biomedical Engineering, Case Western Reserve University, Cleveland, OH 44106 USA, and also with the Functional Electrical Stimulation Center, Louis Stokes Cleveland VAMC, Cleveland, OH 44106 USA.

Abstract

Cervical spinal cord injuries frequently cause paralysis of all four limbs - a medical condition known as tetraplegia. Functional electrical stimulation (FES), when combined with an appropriate controller, can be used to restore motor function by electrically stimulating the neuromuscular system. Previous works have demonstrated that reinforcement learning can be used to successfully train FES controllers. Here, we demonstrate that transfer learning and curriculum learning can be used to improve the learning rates, accuracies, and workspaces of FES controllers that are trained using reinforcement learning.

Keywords

Reinforcement learning; transfer learning; curriculum learning; musculoskeletal model; functional electrical stimulation

I. Introduction

HIGH-LEVEL cervical spinal cord injuries (SCIs) often cause paralysis of all four limbs, resulting in decreased patient independence and quality of life. Functional electrical stimulation (FES) can be used to restore motor function to people with paralysis by electrically stimulating the neuromuscular system [1]. When paired with an appropriate command source, such as a brain computer interface [1], and a controller to coordinate electrical stimulation [2]-[14], FES systems can help people with paralysis regain some independence and quality of life [15].

Many FES controllers have been proposed in the literature [2]-[14], including controllers based on reinforcement learning (RL) [16]. Controllers based on RL are attractive because 1) they require little involvement from trained personnel to train, 2) they can accept subjective user feedback as an objective function [8], 3) they can work with partial state information [7]-[9], [12]-[14], and 4) they can reach to arbitrary target locations within a
continuous region when given information about the current state and goal state of the arm [9].

Recently, an FES controller was trained using RL to move a horizontal-planar model of the human arm with 2 segments (upper and lower arm) and 6 redundant muscles to targets with radii of 7.5 cm that spawned randomly within the joint angle range $[20^\circ, 90^\circ]$ for both the shoulder and elbow [9]. The controller was able to acquire nearly 100% of targets with radii of 5.0 cm with a median time to target of approximately 0.3 seconds. Training took 15-30 minutes of simulated training time. The results of [9] clearly demonstrated the feasibility of using RL for training FES controllers for a two-dimensional musculoskeletal model of the arm.

In this study, we demonstrate that machine learning techniques, including curriculum learning and transfer learning, can be used to improve the learning rates, accuracies, and workspaces of RL controllers for a model of the FES-actuated human arm. In particular, we demonstrate methods that can increase the workspace of the controllers by a factor of 3, improve the fraction of small targets (1.0 cm radii) that are acquired by 50%, and decrease the time it takes to train controllers by as much as 100%.

II. Methods

A. Musculoskeletal Model

We evaluated the controllers by using an existing multi-input, multi-output musculoskeletal model of the human arm, as described previously [4], [7]-[9]. The model had 2 mechanical segments (forearm and upper arm) and 2 joints (shoulder and elbow) that were modeled as pin joints, each with 1 degree of freedom. The arm was actuated by 6 Hill-type muscles that approximated the anterior deltidoid, posterior deltidoid, biceps, brachialis, long head of the triceps, and short head of the triceps. The biceps and long head of the triceps crossed both joints, allowing the controller to be evaluated on a system with redundant actuators. A full illustration of the model is provided in Figure 1a. Euler approximation was used to update the state of the model every 20 ms.

B. Musculoskeletal Model Parameters

For most simulations, musculoskeletal model parameters were chosen to represent an adult male with approximately average height (177 cm) and body mass (80 kg) [9]. All simulations were performed using this average adult male model unless otherwise stated. For some simulations (Section II-F3), we used a model of an adult female with 5th percentile height (149 cm) and mass (41 kg), as well as a model of an adult male with 95th percentile height (190 cm) and mass (99 kg) [17]. Arm segment parameters were scaled according to [18]. Muscle parameters were derived from [19] and [20], and muscle forces were scaled linearly with body mass. To explore how subject-specific differences could influence results, we also optionally increased muscle forces by 30%, which approximated changes in muscle forces associated with strength training in adult males [21]. Alternatively, we decreased muscle forces by 30% (which could represent muscle atrophy due to chronic disuse) to test the robustness of controller training. By changing the strengths of the muscles, we were able
to explore how subject-specific anatomical differences affect controller performance without the confound of perfectly-scaled anatomical parameters.

C. Region Definitions

In order to evaluate controller performance, we defined the following regions. All regions are illustrated in Figure 1b.

1) **Range of Motion**: Range of motion (ROM, Figure 1b, blue region) referred to the passive range of motion of the arm. The passive range of motion a limb was defined as the region of space where the limb could move with assistance from external forces, such as forces applied by the contralateral limb or by another person. In the musculoskeletal model, the (passive) ROM was defined by specifying the minimum and maximum joint angles: \([-20^\circ, 130^\circ]\) for the shoulder and \([5^\circ, 170^\circ]\) for the elbow, as used previously [9].

The active range of motion of a limb is a subregion of the (passive) ROM. The active range of motion of a limb was defined as the region of space where a limb could move when only experiencing forces applied by muscles that are directly attached to the limb. The musculoskeletal model did not include any external forces. Consequently, some regions of the (passive) ROM of the arm model were unreachable for the controllers since those regions were outside of the active range of motion. The parameters of the active range of motion of the arm were not parameters of the model and were not known a priori.

2) **Previous Workspace**: (Figure 1b, yellow region) In previous work, targets were spawned uniformly in joint-angle space between 20° and 90° of flexion for both the shoulder and the elbow [7]-[9]. For an adult male with average limb length, the area of the ROM of the arm is 4.4 times larger than the area of the Previous Workspace.

3) **Virtual Keyboard**: (Figure 1b, green region) Controllers were assessed on a “functional” task by placing a virtual keyboard (45 × 15 cm) in the workspace of the controller and averaging the performance of the controller in that smaller workspace. The position of the virtual keyboard was optimized for each simulation condition using an evolutionary algorithm. Performance in the virtual keyboard region represented the expected upper-bound on single-handed, single-finger typing for a controller that could select keys by hovering above them.

D. Task

Targets were spawned randomly (uniformly and continuously) in joint angle space within the full ROM of the arm. In order to acquire targets, controllers were required to move the endpoint of the arm to within the target region within 900 ms and remain in the target region for 100 ms, as described previously [9]. Reaches where targets were not acquired within 1 second were labeled failures. Once a target was acquired, (the dwell time exceeded 100 ms), or once a reach was failed (the target was not acquired within 1 second), a new target was spawned. The radius of the targets varied between simulation conditions, as described below, within the range of \([0.5, 7.5]\) cm. The distance between the centers of the targets was 45 ±
24 cm (mean ± standard deviation). The beginning state of each reach was the ending state of the previous reach.

E. Controller

1) Reinforcement Learning Algorithm: To train the controllers, we used RL (Figure 1c) [16]. For RL, the controller observed the environment (arm model) and chose an action (commanded relative muscle activation levels) that was applied to the environment, causing the environment to transition to a new state that was then observed by the controller. At each time step, the controller’s performance was assessed by a reward function (described below) that produced a scalar reward. The RL architecture that we chose was based on the actor-critic architecture, as used previously [9]. Here, the actor and critic were implemented as neural networks - each with 2 hidden layers and 64 nodes per layer. The RL algorithm employed twin-delayed deep deterministic policy gradients (TD3) [9], [22] to help the controllers learn faster by stabilizing parameter updates. We also used hindsight experience replay (HER) [9], [23] to allow the controllers learn faster by providing the controller with learning gradients, even when attempted reaches failed. RL controllers were implemented in Python 3.6 using the stable-baselines package version 2.10.0 [24].

2) Inputs and Output: At each time step, the controller observed a 6-dimensional vector composed of: 1) the angular joint position (2D), 2) the angular joint velocity (2D), and 3) the target angular joint position (2D) of the arm. The controller produced a relative muscle activation value (the action) in the range [0, 1] for each of the 6 muscles.

3) Reward Function: As stated above, the reward function produced a scalar reward at each time step based on the performance of the controller. The reward function that we used was composed of two terms. The first term rewarded being in the target region whereas the second term penalized the magnitude of the relative muscle activation levels:

\[ r = 1 \cdot I_{at} - 0.05 \cdot \|a\|_2 \]  \hspace{1cm} (1)

where \( r \) was the reward, \( I_{at} \) was an indicator (boolean) function which was 1 if the endpoint of the arm was “at target” (in the target region) and 0 otherwise, and \( \|a\|_2 \) was the Euclidean norm of the action vector \( a \).

4) Training: Training was performed for 120 minutes of simulated time (360,000, 20 ms time steps). At the beginning of training, the exploration parameter (\( \epsilon \)) was set to 0.3. This meant that 30% of the time, a random action was drawn from a uniform distribution (range [0, 1]) for each of the 6 muscle activations and applied to the environment. Exploration promoted learning by ensuring that the action space was explored [16]. The exploration parameter was adjusted during curriculum learning, as described below.

F. Training Enhancements

In order to improve controller performance and learning speed, we employed the following training enhancements:
1) **Curriculum Learning:** Curriculum learning (Figure 1d) [25], [26] refers to initially teaching a controller an easy task and then teaching it progressively harder tasks. Here, curriculum learning was applied by decreasing target size over time. In the beginning of training, the controller reached to targets with radii of 7.5 cm. Controller performance was assessed using a sliding window over 100 reaches. After the controller successfully acquired 50% of targets, the target size was decreased by 20%. A timeout period of 100 reaches was then enforced to prevent decreases in target radius immediately after target radius was decreased. This process was repeated until the minimum target radius (0.5 cm) was reached, or until the time allotted for controller training expired.

Every time that target size was decreased, the exploration parameter, $\epsilon$, was decreased by 90%. For simulation conditions that did not use curriculum learning, the exploration parameter was set to 0 once 50% of targets were acquired.

2) **Pretraining:** Pretraining (Figure 1e) was inspired by a machine learning technique called demonstration learning [27], where a controller learns to control a system by observing an expert as the expert demonstrates how to control the system. We used the term “pretraining” to emphasize that the data that we used was not generated by an expert trying to perform the same task, but rather, the data was generated during a task that has some similarity to the task of interest. Such data may be generated during electrode profiling sessions within the operating room or post-operatively.

We modeled electrode profiling by activating combinations of muscles with similar actions (i.e. joint flexors or joint extensors) at varying activation levels. We set a maximum relative muscle activation value of 0.25, 0.5, or 1.0 and then smoothly increased the activation value from 0 to the maximum activation value in steps of 5% of the maximum relative muscle activation value. The arm was allowed to reach equilibrium for 2 seconds before changing the activation value. This data set modeled a profiling session where the equilibrium position of the arm was determined as a function of electrode (or muscle) activation values. Then, we simulated an additional data set where, instead of smoothly increasing the activation levels, we instantaneously increased the activation values from 0 to the maximum muscle activation value (0.25, 0.5, or 1.0) or suddenly decreased the activation value from the maximum activation value to 0. This process was repeated for every possible pair of muscles. This data set modeled a profiling session where the effects of stimulation on joint angle velocity were assessed. We calculated rewards for these data sets using Equation 1. The pretraining data simulated 7.75 hours of real-world data collection. Preliminary studies indicated that subsets of the data as short as 52 minutes could produce pretraining effects. However, we chose to use the entire pretraining data set to maximize the pretraining effects.

3) **Transfer Learning:** Transfer learning [28] is a technique where a machine learning model is trained for one application and then used as the starting point of training for another machine learning model used for a different application. Here, we applied transfer learning by training a controller to control a musculoskeletal model of an arm and then using that controller to control another musculoskeletal model of an arm with a different set of parameters (Section II-B). Controller performance was assessed with and without model-specific retraining. During retraining, curriculum learning was employed. However,
pretraining was not used in conjunction with transfer learning since pretraining would overwrite the parameters of the transferred controller, which would limit interpretation of the transfer learning results. Controllers were only transferred if they performed better than the median controller performance minus 1.5 interquartile ranges. After removing poor-performing controllers from transfer learning studies, 27-32 controllers were used per simulation condition.

G. Evaluation

Every 30 minutes of simulated training time, controllers were evaluated for 2,000 reaches for each of the following target radii: \{7.5, 5.0, 2.5, 2.0, 1.5, 1.0, 0.5\} cm. The target of the reach, as well as the success/failure status of the reach was recorded. 32 controllers were trained for each simulation condition, and results from all 32 controllers were pooled to determine the average success rate as a function of simulation condition, target location, target size, and training time. During evaluation, the exploration parameter, $\epsilon$, was always set to zero to make the actions deterministic.

III. Results

A. Increased Workspace

Figure 2 shows the “baseline” average performance of 32 controllers that were trained to reach to targets with radii of 7.5 cm that spawned throughout the entire ROM of the arm. Performance peaked after 60 simulated minutes of training. For targets with radii of 7.5 cm, the average workspace of the controllers was approximately 65% of the ROM, and approximately 97% of the targets in the virtual keyboard region were acquired. For targets with radii of 1.0 cm, the controllers could acquire 14% of targets in the ROM and approximately 33% of the targets in the virtual keyboard region. Controllers were unable to acquire targets in the posterior-contralateral region of the ROM.

B. Training on Smaller Targets

Figure 3 shows the average performance of 32 controllers that were trained to reach to targets with radii of 0.5 cm that spawned throughout the entire ROM of the arm. Performance did not plateau during training. For targets with radii of 7.5 cm, the average workspace of the controllers was approximately 56% of the ROM, and approximately 66% of the targets in the virtual keyboard region were acquired. For targets with radii of 1.0 cm, the controllers could acquire 10% of targets in the ROM and approximately 26% of the targets in the virtual keyboard region. Thus, performance was worse when controllers were trained to acquire smaller targets, even when tested on smaller targets.

C. Curriculum Learning

Figure 4 shows the average performance of 32 controllers that were trained using curriculum learning. Controller performance plateaued after 60 simulated minutes of training, which was similar to the learning time of controllers that were trained on targets with radii of 7.5 cm, only. For targets with radii of 7.5 cm, the average workspace of the controllers was approximately 64% of the ROM, and approximately 100% of the targets in the virtual keyboard region were acquired. For targets with radii of 1.0 cm, the controllers could
acquire 21% of targets in the ROM and approximately 51% of the targets in the virtual keyboard region. Controllers were unable to acquire targets in the posterior-contralateral region of the ROM. Curriculum learning did not impact the performance of controllers when tested on large targets, but it increased the number of small targets that could be acquired by as much as 50%. As in previous simulation conditions, there was a region in the posterior-contralateral region of the ROM that could not be reached.

D. Pretraining

Figure 5 shows the average performance of 32 controllers that were trained using pretraining. Performance peaked after 60 simulated minutes of training. For targets with radii of 7.5 cm, the average workspace of the controllers was approximately 66% of the ROM, and approximately 100% of the targets in the virtual keyboard region were acquired. For targets with radii of 1.0 cm, the controllers could acquire 14% of targets in the ROM and approximately 35% of the targets in the virtual keyboard region. Similar to the previous simulation conditions, there was a region in the posterior-contralateral region of the ROM where targets could not be acquired. Unlike in previous simulation conditions, when using pretraining, some targets could be acquired before training occurred (i.e., at time = 0 minutes). Before training occurred, 23% of targets with radii of 7.5 cm and 1% of targets with radii of 1.0 cm could be acquired within the entire ROM. Before training occurred, 64% of targets with radii of 7.5 cm and 5% of targets with radii of 1.0 cm could be acquired within the virtual keyboard workspace.

E. Curriculum Learning With Pretraining

Figure 6 shows the average performance of 32 controllers that were trained using curriculum learning after pretraining. Performance plateaued at approximately 60 simulated minutes of training. For targets with radii of 7.5 cm, the average workspace of the controllers was approximately 66% of the ROM, and approximately 97% of the targets in the virtual keyboard region were acquired. For targets with radii of 1.0 cm, the controllers could acquire 26% of targets in the ROM and approximately 55% of the targets in the virtual keyboard region. Controllers were unable to acquire targets in the posterior-contralateral region of the ROM. At the beginning of training, 24% of targets with radii of 7.5 cm and 1% of targets with radii of 1.0 cm could be acquired within the entire ROM. At the beginning of training, 66% of targets with radii of 7.5 cm and 5% of targets with radii of 1.0 cm could be acquired within the virtual keyboard workspace.

F. Transfer Learning

Figure 7 shows the average performance of 31 controllers (1 low-performing controller was removed before transfer learning was attempted) that were trained using transfer learning. All controllers were trained to control a model of the arm of a woman with 5th percentile height and weight and 70% of expected muscle strengths. The controllers were then asked to control a model of the arm of a man with 95th percentile height and weight and 130% of expected muscle strengths.

On targets with a radii of 7.5 cm, transferred controllers acquired an average of 63% of targets before model-specific retraining and 61% of targets after retraining. On targets with
a radii of 1.0 cm, transferred controllers acquired an average of 15% of targets before model-specific retraining and 17% of targets after retraining. Supplementary Figures 1-5, show that transfer learning generally worked regardless of which musculoskeletal models were used for training and testing. Generally, model-specific retraining after transfer learning did not improve controller performance, except for some simulation conditions when tested on small (1.0 cm radii) targets.

G. Functional Performance

To simulate a functional task, we optimized the placement of a keyboard-sized rectangle within the ROM. When controllers were trained using curriculum learning, controller performance within the virtual keyboard region remained higher than 90% for target radii larger than 2.5 cm (27 keys) and remained higher than 50% until target radii decreased below 1.0 cm (154 keys).

IV. Discussion

A. Controller Workspace

Here, we demonstrated that RL FES controllers could acquire targets in up to 72% of the arm’s ROM, depending on the size of the target and the anatomical parameters of the model. Determining the upper bound on controller performance is difficult because the upper bound will depend, not only on the position of the target and the anatomical parameters of the model, but also on task parameters, such as dwell time. However, we note that the region where the controllers failed to reach (the posterior-contralateral region) is difficult for non-disabled people to reach, and it is not nearly as functionally relevant as more medial locations in front of the head and ipsilateral shoulder. Therefore, we do not believe further work is needed to expand the workspace of the controller; rather, further work should focus on increasing controller accuracy within functionally critical regions.

Previous works [7], [9], [29] spawned targets only within part of the ROM that was known to be reachable. Here, targets were spawned within the entire ROM, which included regions of the workspace where the arm could not reach using only internally-generated muscle forces. Spawning targets in regions of the workspace where the arm could not reach made the task more difficult for RL because the controller spent time trying to learn unlearnable tasks. In theory, if the unreachable region of the workspace was large enough, RL, as described here, could fail because the controller could learn to minimize muscle forces before it learned how to acquire targets. Fortunately, the uncertainty in the estimation of the reachable workspace did not cause training to fail, nor did it dramatically increase training times compared to previous works. Compared to [9], training times were increased by a factor of 3 (when not using transfer learning), possibly because of a few factors: 1) the workspace was larger, 2) the workspace included an un-reachable region, and 3) the reward function was optimized for smaller targets, which might make learning slower. However, compared to [9], the area of the workspace of the controller was also increased by a factor of 3, which suggests that the investment of additional training time was worthwhile.
The optimized position of the virtual keyboard was typically outside of the “Previous Workspace” (see Section II-C for workspace definitions and Figure 1b for an illustration of the typical placement of the virtual keyboard). This fact suggests that the expanded workspace could meaningfully improve the functional performance of controllers.

B. Curriculum Learning

Figure 2 demonstrates that controllers could be trained quickly to acquire targets with radii of 7.5 cm. However, controllers that were trained to acquire targets with radii of 7.5 cm had difficulty acquiring smaller (1.0 cm radius) targets. Figure 3 shows that controllers learned more slowly and acquired fewer small targets when only trained to acquire targets with radii of 0.5 cm. Notably, the performance of controllers did not plateau by the end of training when only trained to acquire targets with radii of 0.5 cm; additional training time might have improved controller performance. However, controllers that were trained to acquire targets with radii of 0.5 cm were given twice the time required for previous controllers to learn to acquire 7.5 cm targets. Rather than allowing for additional training time, which might be impractical during actual human experiments, we chose to use curriculum learning.

Curriculum learning allowed controllers to learn quickly in the beginning of training because targets were large. Then, as the controllers became proficient at acquiring large targets, target size was decreased, allowing controllers to learn to acquire small (1.0 cm radius) targets. By using curriculum learning, controller accuracy was improved, while maintaining reasonable learning rates.

Learning rates may have been slower when controllers were trained on targets with radii of 0.5 cm because RL controllers choose random actions in the beginning of training, and, when targets are smaller, random actions are less likely to result in target acquisition. RL controllers might learn more from successful reaches than from unsuccessful reaches because, for successful reaches, the controllers learn one, of relatively few, reach strategies that work, rather than learning one, of relatively many, reach strategies that do not work.

C. Pretraining

At the beginning of training, RL controllers take random actions, resulting in random actuator movements. In this work, and in others [22], random actions were drawn from uniform distributions. In systems with non-redundant actuators that can apply torques in both the positive and negative directions (e.g., most robots), sampling from a uniform distribution is reasonably efficient because non-zero actions will produce non-zero movements. However, in a system with redundant, uni-directional actuators, (e.g. the human musculoskeletal system), uniform sampling is less efficient because many actions result in little or no movement. Such inefficiencies can occur, for example, when the joint torques applied by multiple actuators sum to zero.

In order to make learning more efficient in the beginning of training, the controller can be initialized to produce non-random movements. One way to do this is to use preexisting data to pretrain the controller. Such data might already exist because of the evaluation of electrode placement in the operating room, or because of clinical evaluation of FES system capabilities post-operatively. Here, we chose to use 2 simulated datasets simultaneously to
pretrain the neural networks. The combined datasets included both low- and high-magnitude velocities that spanned most of the ROM of the arm.

Pretraining, when used alone, allowed controllers to acquire many large (7.5 cm radius) targets, but few small (1.0 cm radius) targets, before training occurred. After 30 minutes of simulated training, controllers that were not pretrained performed comparably to controllers that were pretrained. These results suggest that pretraining, alone, did not improve the learning rate of FES controllers, nor did it improve the accuracy of trained controllers.

When added to curriculum learning, pretraining did not noticeably improve the learning rates of controllers. Compared to controllers that were trained using only curriculum learning, controllers that were trained using curriculum learning with pretraining acquired 2% more (overall) of the targets with radii of 7.5 cm and acquired 5% more (overall) of the targets with radii of 1.0 cm within the ROM. Pretraining may have improved the percentage of targets acquired without noticeably improving controller learning rates by exposing the controllers to more diverse states, allowing the controllers to generalize better.

Even in cases where pretraining does not improve learning speed or accuracy, pretraining may be useful to confirm that an FES system is “controllable” before full training is performed. This pre-check on training could be useful when training controllers for biomechanical systems with more actuators and more mechanical degrees of freedom because such controllers will take more time to train.

D. Transfer Learning

Figure 7, as well as Supplementary Figures 1-5, shows that controllers that were trained to control musculoskeletal models of human arms could be used to control other arm models, even without model-specific retraining.

Notably, transfer learning worked even when controllers were transferred between models with radically different parameters. These results suggest that transfer learning will work even when applied to FES systems implanted in arms with radically different sizes. In general, model-specific retraining after transfer learning did not improve the percentage of targets that could be acquired. These results suggest that transfer learning can dramatically improve controller learning rates to the extent that patient-specific training examples may not be necessary under many circumstances.

E. Comparison to Previous Initialization Techniques

In this work, we tested two ways to initialize the RL controllers to increase the learning rate. The first technique, pretraining, attempted to initialize the controllers using data that was collected from the same biomechanical system, but using a dissimilar task. The second method, transfer learning, attempted to initialize the controllers using data collected during a similar task, but using dissimilar biomechanical systems. In this study, initializing controllers using data from a similar task seemed to increase learning rates more than initializing the controllers using data from similar biomechanical systems. Pretraining may have caused the controllers to learn suboptimal control strategies that had to be partially unlearned before better strategies could be learned. Thus, pretraining did not noticeably
improve learning rates. However, with transfer learning, the controller used a good control strategy in the beginning.

Transfer learning improved learning rates more than pretraining while also using less data. Transfer learning required only 60 minutes (1 hour) of data, whereas the dataset that we used for pretraining contained 7.75 hours of data. While it is unlikely that 7.75 hours of data would be available for pretraining, we chose to simulate such a large dataset in order to exclude the possibility that pretraining performed worse because the dataset was small and lacked diverse training examples.

Many works have relied on initialization techniques to train RL controllers for the human arm [7], [14], [29]. Other works noted that controller initialization was not necessary for RL to occur [9]. This work supports the idea that controller initialization may improve the learning rate of RL controllers for the human arm, but initialization may not be necessary, depending on the specific RL architecture chosen.

F. Limitations

The simplified model of the human arm that we used did not consider gravity or time-varying dynamics (e.g. muscle spasticity). These simplifications are common in the literature [7], [9]-[11], [13], [14], [29]. We expect that, in more complex musculoskeletal models, controller learning rates will be decreased. Thus, the methods that we have presented, here, will be even more pertinent when applied to more complex musculoskeletal systems. Future work should focus on applying these methods to more complete models of the human arm.

Preliminary studies (Supplementary Figure 6) suggest that trained controllers are robust to high levels of fatigue that decrease muscle strengths by up to 90%. Future studies should consider how fatigue may impact the training process since time-varying systems are difficult to learn to control.

V. Conclusion

We have demonstrated several enhancements to RL algorithms that can be used to train controllers for multi-input, multi-output musculoskeletal models of the human arm. The results suggest that the following process should be used to train RL controllers for FES-actuated arms. If an FES controller for a participant already exists, that controller can likely be used on another participant (transfer learning). If controllers from multiple participants are available, the controller for the largest participant’s arm should be used for transfer learning.

If no controller from another participant is available, transfer learning should be attempted by training the controller on a model of the human arm [14]. Curriculum learning should be used to efficiently train the controller to reach smaller targets. If stimulation and movement data are available, they can be used to pretrain the controller. Pretraining allows the investigator/clinician/participant to immediately know that the system is controllable, and, when combined with curriculum learning, pretraining may slightly improve performance.
The results of this study should be confirmed in a more complete model of the human arm that includes, for instance, gravity and muscle fatigue during the training phase.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgment

The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health. This work made use of the High Performance Computing Resource in the Core Facility for Advanced Research Computing at Case Western Reserve University. The authors would like to thank Dr. Musa Audu for his contributions to the transfer learning studies.

This work was supported in part by the National Institute of Biomedical Imaging and Bioengineering of the National Institutes of Health under Award T32EB004314 (DCC) and in part by the National Science Foundation Graduate Research Fellowship under Grant 1451075 (DCC).

References

[1]. Ajiboye AB et al., “Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: A proof-of-concept demonstration,” Lancet, vol. 389, no. 10081, pp. 1821–1830, 2017. [PubMed: 28363483]

[2]. Koike Y, Gonzalez J, Gomez J, and Yu W, “Implementing feedback error learning for FES control,” in Proc. 4th Int. Conf. Biomed. Eng. Informat. (BMEI), Oct. 2011, pp. 1324–1328.

[3]. Kurosawa K, Futami R, Watanabe TA, and Hoshimiya N, “Joint angle control by FES using a feedback error learning controller,” IEEE Trans. Neural Syst. Rehabil. Eng, vol. 13, no. 3, pp. 359–371, Sep. 2005. [PubMed: 16200759]

[4]. Blana D, Kirsch RF, and Chadwick EK, “Combined feedforward and feedback control of a redundant, nonlinear, dynamic musculoskeletal system,” Med. Biol. Eng. Comput, vol. 47, no. 5, pp. 533–542, May 2009. [PubMed: 19343388]

[5]. Balaghi MHE, Vatankhah R, Broushaki M, and Alasty A, “Adaptive optimal multi-critic based neuro-fuzzy control of MIMO human musculoskeletal arm model,” Neurocomputing, vol. 173, no. 3, pp. 1529–1537, Jan. 2016.

[6]. Schearer EM et al., “Multi-muscle FES force control of the human arm for arbitrary goals,” IEEE Trans. Neural Syst. Rehabil. Eng, vol. 22, no. 3, pp. 654–663, May 2014. [PubMed: 24122573]

[7]. Jagodnik KM, Thomas PS, van den Bogert AJ, Branicky MS, and Kirsch RF, “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, Oct. 2016.

[8]. Jagodnik KM, Thomas PS, van den Bogert AJ, Branicky MS, and Kirsch RF, “Training an actor-critic reinforcement learning controller for arm movement using human-generated rewards,” IEEE Trans. Neural Syst. Rehabil. Eng, vol. 25, no. 10, pp. 1892–1905, Oct. 2017. [PubMed: 28475063]

[9]. Crowder DC, Abreu J, and Kirsch RF, “Hindsight experience replay improves reinforcement learning for control of a MIMO musculoskeletal model of the human arm,” IEEE Trans. Neural Syst. Rehabil. Eng, vol. 29, pp. 1016–1025, 2021. [PubMed: 33999822]

[10]. Golkhov V, Lucas C, and Parnianpour M, “Application of actor-critic reinforcement learning method for control of a sagittal arm during oscillatory movement,” Biomed. Eng., Appl., Basis Commun, vol. 16, no. 6, pp. 305–312, Dec. 2004.

[11]. Golkhov V, Parnianpour M, and Lucas C, “The role of multisensor data fusion in neuromuscular control of a sagittal arm with a pair of muscles using actor-critic reinforcement learning method,” Technol. Health Care, vol. 12, no. 6, pp. 425–438, Jan. 2005.

[12]. Tahami E, Jafari AH, and Fallah A, “Learning to control the three-link musculoskeletal arm using actor-critic reinforcement learning algorithm during reaching movement,” Biomed. Eng., Appl., Basis Commun, vol. 26, no. 5, Oct. 2014, Art. no. 1450064.
[13]. Caligiore D, Guglielmelli E, Borghi AM, Parisi D, and Baldassarre G, “A reinforcement learning model of reaching integrating kinematic and dynamic control in a simulated arm robot,” in Proc. IEEE 9th Int. Conf. Develop. Learn., Aug. 2010, pp. 211–218.

[14]. Febbo DD et al., “Does reinforcement learning outperform PID in the control of FES-induced elbow flex-extension?” in Proc. IEEE Int. Symp. Med. Meas. Appl. (MeMeA), Jun. 2018, pp. 1–6.

[15]. Taylor P, Esnouf J, and Hobby J, “The functional impact of the freehand system on tetraplegic hand function. Clinical results,” Spinal Cord, vol. 40, no. 11, pp. 560–566, Nov. 2002. [PubMed: 12411963]

[16]. Sutton RS and Barto AG, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[17]. Man-System Integration Standards, National Aeronautics and Space Administration, Houston, TX, USA, 1995.

[18]. Winter DA, Biomechanics Motor Control Human Movement, 4th ed. Hoboken, NJ, USA: Wiley, 2009.

[19]. Garner BA and Pandy MG, “Estimation of musculotendon properties in the human upper limb,” Ann. Biomed. Eng, vol. 31, no. 2, pp. 207–220, Feb. 2003. [PubMed: 12627828]

[20]. Bhushan N and Shadmehr R, “Computational nature of human adaptive control during learning of reaching movements in force fields,” Biol. Cybern, vol. 81, no. 1, pp. 39–60, Jul. 1999. [PubMed: 10434390]

[21]. Mitchell CJ, “Muscle strength and activation characteristics of power-trained and non-athlete boys and men,” M.S. thesis, Dept. Appl. Health Sci, Brock Univ., Catharines, ON, Canada, 2009.

[22]. Fujimoto S, van Hoof H, and Meger D, “Addressing function approximation error in actor-critic methods,” in Proc. 35th Int. Conf. Mach. Learn., 2018, pp. 1582–1591.

[23]. Andrychowicz M et al., “Hindsight experience replay,” in Proc. 31st Conf. Neural Inf. Process. Syst., 2017, pp. 5055–5065.

[24]. Hill A et al. (2018). Stable Baselines. [Online]. Available: https://github.com/hill-a/stable-baselines

[25]. Bengio Y, Louradour J, Collobert R, and Weston J, “Curriculum learning,” in Proc. 26th Annu. Int. Conf. Mach. Learn., 2009.

[26]. Fang M, Zhou T, Du Y, Han L, and Zhang Z, “Curriculum-guided hindsight experience replay,” in Proc. 33rd Conf. Neural Inf. Process. Syst., 2019.

[27]. Schaal S, “Learning from demonstration,” in Proc. Adv. Neural Inf. Process. Syst. (NIPS), vol. 9, 1997.

[28]. Goodfellow I, Bengio Y, and Courville A, Deep Learning, 1st ed. Cambridge, MA, USA: MIT Press, 2017.

[29]. Jagodnik KM, Blana D, van den Bogert AJ, and Kirsch RF, “An optimized proportional-derivative controller for the human upper extremity with gravity,” J. Biomech, vol. 48, no. 13, pp. 3692–3700, 2015. [PubMed: 26358531]
Fig. 1. Overview of the reinforcement learning algorithm and training enhancements.

(A) The musculoskeletal model of the arm, superior view. The arm was composed of an upper segment (blue) and a lower segment (yellow) with simple pin joints at the shoulder and elbow. The model contained the following Hill-type muscles: (a) anterior deltoid, (b) posterior deltoid, (c) brachialis, (d) triceps brachii (short head), (e) biceps brachii, (f) triceps brachii (long head). Joint angles at the shoulder and elbow are indicated by $\theta_1$ and $\theta_2$, respectively. $r_t$ indicates the radius of the target, $T$. The controller was provided with the location of the target and required to acquire the target by causing the endpoint of the arm (red dot) to dwell within the target region. Adapted from [9].

(B) The range of motion (denoted as “ROM”) of the arm was defined as the region of space where the arm could reach if unimpeded by objects (such as the torso). In previous works [7]-[9], targets were spawned uniformly in joint-angle space between 20° and 90° of flexion for both the shoulder and the elbow (denoted as “Prev.”). The position of a virtual keyboard (denoted as “Key.”) was optimized to assess functional performance.

(C) During reinforcement learning, controllers chose actions (muscle activation values). The actions were applied to the environment (the arm model, denoted as “Environ.”), which produced observations (joint position, joint velocity, goal position, and reward, denoted as “Obs.”). The observations were used by a parameter update function (denoted $F_{update}$) to train the controllers. The controllers used the observations to choose the next actions. (D) Curriculum learning was used to progressively decrease target radius every time controller performance reached a predefined threshold. (E) In an attempt to improve learning rates, controllers were pretrained using previously-simulated data. The reinforcement learning model (denoted as “RL”) observed the muscle activation levels and the corresponding movements of the arm. At the end of pretraining, control of the arm was switched from preprogrammed stimulation patterns to the RL model. (F) Controllers (denoted as “RL”) were trained to control a
musculoskeletal arm model and then “transferred” to another musculoskeletal arm model with a different set of anatomical parameters.
Fig. 2. Percent of targets acquired by controllers that were trained on targets with radii of 7.5 cm.
Controllers were trained to reach to targets with radii of 7.5 cm. Every 30 minutes of simulated training time, controllers were evaluated on targets that had radii in the range [0.5, 7.5] cm. Controller success was plotted as a function of target location in Cartesian space, with warm colors indicating good performance. By 60 minutes of training, performance plateaued. Nearly 100% of targets with radii of 7.5 cm were acquired, except in the posterior-contralateral region, which likely could not be reached using only internally-generated muscle forces. Performance on smaller targets quickly dropped off. Ipsi. = Ipsilateral, Ant. = Anterior.
Fig. 3. Percent of targets acquired by controllers that were trained on targets with radii of 0.5 cm.

Controllers were trained to reach to targets with radii of 0.5 cm. Every 30 minutes of simulated training time, controllers were evaluated on targets that had radii in the range [0.5, 7.5] cm. Controller success was plotted as a function of target location in Cartesian space, with warm colors indicating good performance. Performance did not plateau by the end of the 120 minute training period. Despite being trained on smaller targets, controllers that were trained on targets with radii of 0.5 cm were not able to acquire small targets better than controllers that were trained on targets with radii of 7.5 cm. Ipsi. = Ipsilateral, Ant. = Anterior.
Fig. 4. Percent of targets acquired by controllers that were trained using curriculum learning. Controllers were trained to reach to targets with radii of 7.5 cm. Once controllers were able to successfully acquire 50% of targets, target size was decreased. Every 30 minutes of simulated training time, controllers were evaluated on targets that had radii in the range [0.5, 7.5] cm. Controller success was plotted as a function of target location in Cartesian space, with warm colors indicating good performance. By 60 minutes of training, performance plateaued. Performance on large targets was comparable to the performance of controllers that were trained using only targets with radii of 7.5 cm without curriculum learning. Performance on small targets (1.0-2.0 cm radii) was noticeably improved compared to controllers that were trained using targets with radii of 7.5 cm or 0.5 cm, only. Ipsi. = Ipsilateral, Ant. = Anterior.
Fig. 5. Percent of targets acquired by controllers that were trained using targets with radii of 7.5 cm with pretraining. Controllers were pretrained on previously-acquired data and then trained to reach to targets with radii of 7.5 cm. No curriculum learning was used. Every 30 minutes of simulated training time, controllers were evaluated on targets that had radii in the range [0.5, 7.5] cm. Controller success was plotted as a function of target location in Cartesian space, with warm colors indicating good performance. By 60 minutes of training, performance plateaued. Performance on large targets was comparable to the performance of controllers that were trained using only targets with radii of 7.5 cm. Performance on small targets (1.0-2.0 cm radii) was comparable to controllers that were trained using only targets with radii of 7.5 cm but noticeably decreased compared to controllers that were trained using curriculum learning. Controllers that were trained using pretraining were able to acquire targets before training occurred (time = 0 minutes). Ipsi. = Ipsilateral, Ant. = Anterior.
Fig. 6. Percent of targets acquired by controllers that were trained using curriculum learning with pretraining.
Controllers were pretrained on previously-acquired data and then trained to reach to targets of radius 7.5 cm. Every time controllers were able to reach to targets more than 50% of the time, target size was decreased. Every 30 minutes of simulated training time, controllers were evaluated on targets that had radii in the range [0.5, 7.5] cm. Controller success was plotted as a function of target location in Cartesian space, with warm colors indicating good performance. By 60 minutes of training, performance plateaued. Like in the pretraining-only simulation condition, the controllers were able to acquire some targets even before training occurred. After 30 minutes of training, performance on all target sizes was comparable to controllers trained using only curriculum learning. Ipsi. = Ipsilateral, Ant. = Anterior.
Fig. 7. Percent of targets acquired by controllers that were trained using transfer learning. Controllers were trained to control the arm of a virtual woman with 5th percentile height and weight and 70% of expected muscle strengths using curriculum learning, but not pretraining. The controllers were then used to control, with or without retraining, the arms of a man with 95th percentile height and weight and 130% of expected muscle strengths. Controllers successfully controlled musculoskeletal models of arms with radically different anatomical parameters. Model-specific retraining slightly improved controller performance, particularly for small target sizes. Similar results were achieved when controllers were trained or tested on models with different parameters (Supplementary Figures 1-5). Ipsi. = Ipsilateral, Ant. = Anterior.