

Developing a Quasi-Static Controller for a Paralyzed Human Arm: A Simulation Study

Derek N. Wolf and Eric M. Schearer¹

Abstract—Individuals with paralyzed limbs due to spinal cord injuries lack the ability to perform the reaching motions necessary to every day life. Functional electrical stimulation (FES) is a promising technology for restoring reaching movements to these individuals by reanimating their paralyzed muscles. We have proposed using a quasi-static model-based control strategy to achieve reaching controlled by FES. This method uses a series of static positions to connect the starting wrist position to the goal. As a first step to implementing this controller, we have completed a simulation study using a MATLAB based dynamic model of the arm in order to determine the suitable parameters for the quasi-static controller. The selected distance between static positions in the path was 6 cm, and the amount of time between switching target positions was 1.3 s. The final controller can complete reaches of over 30 cm with a median accuracy of 6.8 cm.

I. INTRODUCTION

Functional electrical stimulation (FES) is a promising technology that restores movement to paralyzed muscles by delivering electrical current to the nerves and muscles directly. Using FES to control reaching motions could allow individuals with paralysis from spinal cord injuries to regain their independence.

Achieving reaching motions with FES has proven difficult due to the complexity of human arm motions. For repetitive tasks such as standing [1] and hand function [2], the complexities of the nonlinear, redundant musculoskeletal system have been overcome with predetermined fixed stimulation patterns. The goal-directed nature of reaching motions, however, requires different and potentially new stimulation patterns for every reach.

Many different control strategies have been proposed to achieve FES reaching. The state-of-the-art strategy, demonstrated in the BrainGate2 clinical trial, simultaneously controlled each joint independently [3]. For each joint, the controller selected (based on user intent recorded by an intracortical brain computer interface) a position along a predefined stimulation pattern. While the user's intent was accurately perceived, it was difficult to control multi-joint movements because the independent joint control could not account for the interactions of the joints. To accurately control reaching motions, it is necessary to treat the arm as a complete system.

This work was supported by NIH NINDS grant N01-NS-5-2365, Veterans' Affairs grant B2359-C, NSF grant 1751821, and the Cleveland State Graduate Student Research Award.

¹D. N. Wolf and E. M. Schearer are with the Department of Mechanical Engineering, Cleveland State University and the Cleveland Functional Electrical Stimulation Center, Cleveland, OH USA d.n.wolf@vikes.csuohio.edu, e.scheerer@csuohio.edu.

In simulations, optimized proportional-derivative control [4], combined feedforward-feedback control [5], reinforcement learning [6], and threshold control [7] have all proven successful in controlling reaching motions. Practical implementation of these methods has proven difficult due to the real-world arm dynamics differing from the simulation.

To control the arm as a complete system and determine the real-world arm dynamics necessary for accurate control, model-based methods have been proposed in previous works. Physics-based models have shown some success in controlling two muscles for rehabilitation after stroke [8]. However, identifying the physical parameters of the whole arm requires significant amounts of data. Black-box model-based control methods have been developed to help solve this issue. One such method achieved feedback control of planar arm tasks using an artificial neural network to produce a map of the task space configuration to the forces the muscles can produce [9]. We have used similar nonparametric and semiparametric concepts to produce a model-based controller capable of holding three-dimensional static arm configurations [10][11]. However, extending these methods to full-arm model-based three-dimensional reaching control has not yet been achieved.

Previous work has demonstrated that a semiparametric Gaussian Process Regression (GPR) model could form the basis of a controller for achieving three-dimensional dynamic trajectories [12]. However, including velocity in the controller made it difficult to select trajectories that could be physically achieved with FES. Instead, we have developed a combined feedforward-feedback controller for holding static wrist positions [11]. To achieve reaching movements, we propose using a version of this controller to move the arm through a quasi-static reaching motion. We define a quasi-static path as a series of intermediate static positions which connect the starting arm configuration to the target arm configuration. The controller will shift the desired static target wrist position to the next position in the path until the final target is reached.

Before implementing the quasi-static controller with a human subject, we conducted the current simulation study. The simulation study was designed to replicate the experimental conditions which we work with during our research with human subjects [11]. This includes the use of an elastic arm support which people with high tetraplegia use to assist against gravity to produce a more effective reachable workspace.

Our previous work has demonstrated that the elasticity of the arm support can combine with the time delays inherent

to an FES system (the relatively slow frequency of switching the stimulation signal and muscular activation dynamics) to produce oscillations (which can be uncomfortable for the subject) that basic derivative control is unable to eliminate [13]. Advanced control techniques have been developed to account for these delays in single joint movements [14] as well as in a model-based controller of a simulated arm [15]. Applying these techniques to an FES-controlled human arm, where the parameters of the model are not easily determined, has yet to be implemented. As a simple solution to practically solving the oscillation issue, we propose the addition of damping to the arm support.

The goals of this study are to demonstrate the feasibility of a quasi-static controller for controlling reaching and to determine the conditions for successful reaching. A secondary goal is to study the effects of adding physical damping to the arm support.

The result of this simulation study is a starting set of parameters (to be further tuned in practice) for a quasi-static controller capable of being practically implemented to drive a paralyzed human arm through three-dimensional reaching motions.

II. METHODS

The goal of the experiment was to simulate the experimental setup presented in [11] where we used an implanted neuroprosthesis to actuate the arm of an individual with high tetraplegia (We will refer to this experiment as the laboratory study). We used a MATLAB based dynamic simulation of the arm to recreate the conditions of the laboratory study. As in human experiments, we assume that our controller does not have access to the true dynamics of the arm, and we identified a non-parametric model of the response of the simulated human arm to muscle activation inputs. We used this model as the basis of a feedforward-feedback controller. We tuned the controller and added damping to the arm-support to achieve good performance at static positions. The controller was then used to move the arm to points of varying distances to determine the best distance and time between intermediate wrist positions in a quasi-static path. The best parameters were used to control the arm through complete reaching motions and compared to a controller without a path of intermediate static positions.

A. Simulation Experiment Setup

The goal of the simulation study was to recreate the conditions of the laboratory study. The computer simulation thus consisted of a musculoskeletal model of the arm, an elastic arm support, and a robot.

To simulate the subject's arm, we used the Dynamic Arm Simulator, a MATLAB based dynamic model of the arm [16]. The model has seven links, eleven degrees of freedom, and 138 muscle elements. The model includes the multibody dynamics of the links as well as muscle activation dynamics. In all parts of the present study, the time step used in the simulation was 3 ms.

The Dynamic Arm Simulator model is actuated by inputting the neural excitations, $\mathbf{u}$, which correspond to the desired muscle activations. The neuroprosthesis used in the laboratory study applies stimulation at a frequency of 13 Hz to induce the desired muscle activations. Though the desired control input can be calculated at every time step, switching the input can only occur at the stimulation frequency. In the Dynamic Arm Simulator simulation, we modeled this by restricting $\mathbf{u}$ to only change at 13 Hz. We controlled only the muscle elements which are able to be controlled by the neuroprosthesis in the laboratory study. In the Dynamic Arm Simulator, we controlled the muscle elements related to the triceps, deltoids, latissimus dorsi, serratus anterior, biceps/brachialis, supraspinatus/infraspinatus, rhomboids, lower pectoralis, and upper pectoralis. When a neural excitation is applied to a muscle group, all elements in the group receive the same excitation.

We simulated a passive arm support that is typically used by individuals with spinal cord injuries to assist against the force of gravity and create a more functional workspace. The support used in the laboratory study uses elastic bands to create a resting equilibrium position of the wrist approximately at the top of the thorax slightly forward from the body, and we aimed to simulate the behavior of this arm support. The stiffness of the support were 0 N/m in the X direction and 30 N/m in both the Y and Z directions (due to the orientation of the elastic bands in the laboratory study's arm support, the majority of force is applied in the Y and Z directions). The equilibrium point for the support was placed at [0 m, 0.3 m, -0.15 m] (see the coordinate frame in Fig. 2). Choosing the damping of the support is one of the goals of the study.

In the laboratory study, a robot is used during model identification and at the start of control to hold the wrist in the desired position. In the computer simulation, a PID controller was used to apply force at the wrist to mimic this robot.

We used the included Dynamic Arm Simulator OpenSim model to visualize all trials and experiments [17][18].

B. Model Identification

The model identification procedures are developed in detail in [11] and [12]. We present a summary here.

We developed a two-part model consisting of the inverse statics (the mapping from arm configuration to joint torque required to hold the configuration) and the muscle torque production (the mapping from configuration and muscle activation to the joint torques produced). In the same way as the laboratory study, to gather data for the model, the robot held the wrist of the simulated arm in a series of 27 positions throughout the workspace. At each position, each muscle group was individually activated at 100% neural excitation for 0.5 seconds, and for one period of 0.5 seconds, no muscles were activated, $\mathbf{u} = 0$. The force required for the robot controller to hold the wrist at a static position and the configuration of the arm were recorded. The configuration of the arm, $\mathbf{q}$, is defined by the angles between the thorax and the humerus (shoulder elevation plane, shoulder elevation,

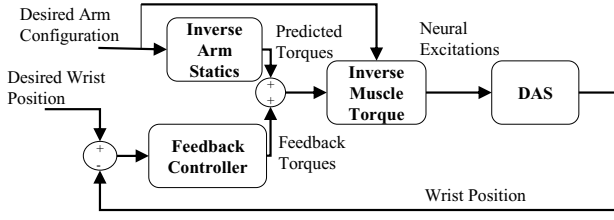


Fig. 1. The block diagram for our controller. The controller uses model-based blocks and a feedback controller to automatically calculate the neural excitations to apply to the dynamic arm simulator (DAS) to achieve the desired wrist position that corresponds to the desired arm configuration.

and shoulder rotation) as defined in [19]) as well as the elbow flexion and pronation angles. The force required to hold the wrist position and joint configurations were averaged over the last 10% of each trial. The kinematic Jacobian was used to transform the recorded robot controller force to the joint torques, τ_j , about the shoulder and elbow which produce the equivalent force (j represents the muscle group being activated with 0 representing no muscles being active).

The torques needed to hold the wrist in a static position, $\mathbf{p}(\mathbf{q}) \in \mathbb{R}^{4 \times 1}$, (The torque about elbow pronation is not included as it does not affect the position of the wrist.) with no muscles activated represent the arm statics, and therefore,

$$\tau_0 = \mathbf{p}(\mathbf{q}). \quad (1)$$

The difference between the torques recorded with no muscles active and the torques recorded with muscle group j active represents the amount of torque produced by muscle group j . The amount of torque produced by a muscle being activated is represented by $\mathbf{R}(\mathbf{q})\alpha$ where $\alpha \in \mathbb{R}^{9 \times 1}$ is the vector of muscle group activations and $\mathbf{R}(\mathbf{q}) \in \mathbb{R}^{4 \times 9}$ is the mapping from muscle group activation to joint torque. The j^{th} column of $\mathbf{R}(\mathbf{q})$, $\mathbf{R}_j(\mathbf{q})$, represents torques about the shoulder elevation plane, shoulder elevation, shoulder rotation, and elbow flexion produced by muscle group j . Therefore,

$$\mathbf{R}_j(\mathbf{q}) = \tau_0 - \tau_j. \quad (2)$$

The data from the set of 27 training positions was used to train semiparametric GPR models [20] which are used to predict τ_j for $j = 0, 1, \dots, 9$ for a given configuration. The models can therefore be used to determine the static arm torques $\mathbf{p}(\mathbf{q})$ and the muscle force mapping $\mathbf{R}(\mathbf{q})$ for a desired arm configuration.

C. Controller

Our controller (see Fig. 1) automatically determines the muscle activations required to hold a static wrist position. The input to the controller is the desired arm configuration, $\mathbf{q}_*$, defined by the three angles between the humerus and the thorax (shoulder elevation plane, shoulder elevation, and shoulder rotation) and elbow flexion and pronation. This controller maps the desired arm configuration to the predicted torques needed to maintain the configuration. These torques are modified by the feedback controller to produce the desired torques necessary to be achieved by the muscles. The

output of the controller is the set of neural excitations which correspond to the muscle activations that produce the desired torques, and these excitations are applied to the Dynamic Arm Simulator arm.

The inverse arm statics block uses the model developed in section II-B to calculate the open-loop joint torques, $\mathbf{p}(\mathbf{q}_*)$, about the shoulder and elbow to hold wrist at the desired position. A PID controller calculates corrective forces in each cardinal direction (X, Y, and Z directions) required to hold the wrist at the desired position. The kinematic Jacobian is used to transform these forces to the equivalent torques about the shoulder and elbow. These feedback torques, τ_{FB} , are added to the open-loop torques to produce the total desired torque, $\tau_{total} = \mathbf{p}(\mathbf{q}_*) + \tau_{FB}$.

The controller next maps the total torque to the muscle group activations which produce the desired torques. The controller uses the GPR models of muscle torque production to determine the muscle-torque mapping, $\mathbf{R}(\mathbf{q}_*)$, for the desired configuration. (Note: The inverse arm statics and inverse muscle torques are found for the desired configuration and not the current configuration.) Since the system is redundant, we determine the muscle activations, $\alpha \in \mathbb{R}^{9 \times 1}$, which produce the desired total torques by solving an optimization problem which minimizes the muscle activations such that the desired torques are produced and $\alpha \in [0, 1]$. The neural excitations input to the Dynamic Arm Simulator model are equivalent to the desired muscle activations, $u_j = \alpha_j$ for the j^{th} muscle group. The actual activation achieved is determined by the muscle activation dynamics of the Dynamic Arm Simulator.

The model was used to determine a set of feasible wrist positions throughout the workspace. A grid of wrist positions within the positions used during model identification with 1 cm spacing was produced. The feasibility of each wrist position was determined by the ability of the model to find a set of muscle activations which can produce the required open-loop torques for the configuration that corresponds to the desired wrist position. The map of feasible points is shown in Fig. 2.

To tune the controller, 15 wrist positions were randomly selected from the map of feasible points. The open-loop portion of the controller was used with the goal of holding the static wrist position for three seconds. The final distance from the desired position was recorded for all trials. The feedback controller was tuned at the position with the median error. The controller was tuned with the goal of improving the accuracy of holding a static position for three seconds while minimizing oscillations. The gains of the PID controller were tuned to increase the accuracy of the holding the static position while also limiting oscillations. As previously discussed, the existence of time delays in the system caused by the 13 Hz frequency of changing neural excitations as well as the muscle activation dynamics meant that simple derivative gain in the FES controller was unable to eliminate oscillations (see Fig. 4). Instead, damping was added to the arm support. The final FES controller was a PI controller with tuned parameters of 250 N/m for the proportional gain

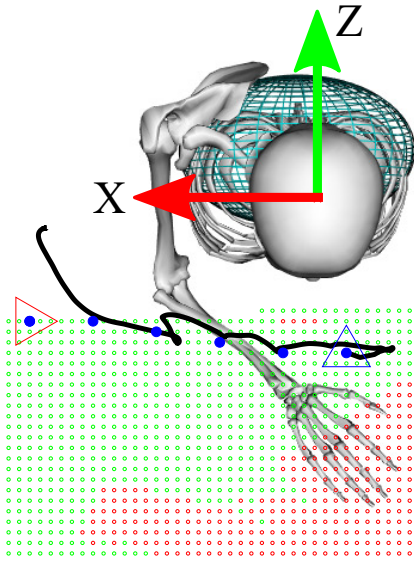


Fig. 2. A 2-D projection of the 3-D map of the feasible wrist positions in the modeled workspace. Feasible points (green) are where the controller is able to determine predict a set of muscle activations capable of achieving the model predicted static arm torques. The map was produced with a 1 cm spaced grid of wrist positions. The image also shows an example reaching motion completed by the final quasi-static controller. The intermediate points are shown by blue circles and the triangles represent the start (blue) and end target positions (red). The reach shown was a 32 cm reach with an accuracy of 9.5 cm. As seen, the reach has fairly good accuracy until the arm moves to the edge of the feasible workspace.

and 80 N/m-s for the integral gain. 120 N-s/m was chosen for the damping of the arm support. The gains were the same for the X, Y, and Z directions.

D. Quasi-static Path Following

The controller was used to control the wrist to follow a quasi-static path. A series of static wrist positions was selected from the feasibility map (Fig. 2) which connected the starting position to the target position. As the arm moved along the path, the desired configuration shifted to the next static point in the path and was input to the controller.

To determine a suitable distance and amount of time between points in a quasi-static path, point to point reaches of different distances were completed. A starting wrist position and target wrist position a set distance, d , away from the start position were randomly selected from the feasibility map. The corresponding target arm configuration was input to the controller and the controller drove the arm for three seconds. The final distance from the target wrist position and the 80% rise time were recorded for each point. This process was repeated at 100 start wrist positions for distances, $d = 2, 4, 6, \dots, 16$ cm for a total of 800 reaches. The mean accuracy and mean rise time of all targets of a single d were recorded.

From the single point to point studies, the best parameters (the distance between positions, d_* and the time between switching positions, t_{switch}) were selected to test on complete quasi-static reaching paths. The required distance between positions, d_* was selected based upon the accuracy of the point to point reaches. The required time between

switching positions, t_{switch} , was determined by the average rise time of the trials. The reasoning for using the 80% rise time was to switch to the next target after being close enough to the current target to still have accurate control. If switching occurred fast enough, the motion could become smoother (less stops and starts).

Using the selected parameters, complete quasi-static reaching paths were completed. Start and end positions at least 30 cm apart were randomly selected from the feasibility map. A path of wrist positions, each a maximum distance of d_* cm from the previous point, was selected which connected the two points. As the arm moved along the path, the desired wrist position shifted to the next point in the path every t_{switch} . The final position is held for $2t_{switch}$ to allow for the controller to settle at the position. The quasi-static paths were compared to the same controller but with no path of intermediate positions (the goal position is the final target from the initial time step) for the same average speed of reach as the final controller.

The final accuracy (the distance from the average hand position over the final 0.3 seconds of a trial to the final target position) was recorded for each trial. For a grouping of trials, the overall accuracy was determined by the mean of the accuracy for all trials in the group. A t-test was used to compare the quasi-static controller to a simple PI controller with no intermediate positions (referred to as the simple controller).

III. RESULTS

We have developed a quasi-static controller that is capable of moving the wrist of a simulated arm along a path between a starting position and target position. Using a controller with a distance between positions of 6 cm and a time between switching points of 1.3 seconds, we were able to achieve a median accuracy of 6.8 cm (mean: 10.5 cm, standard deviation: 9.4 cm) over a series of 200 reaches longer than 30 cm. This is significantly better than the simple controller that does not use intermediate points (mean: 20 cm, $p < 0.01$). Fig. 2 shows an example trial of the final controller achieving a 31 cm reach with an accuracy of 0.6 cm.

Fig.3 shows the time histories in the X direction of both the quasi-static controller and the simple controller for achieving a 32 cm reach with accuracy of 1.1 cm and 27.4 cm respectively. The quasi-static controller achieved better accuracy, but required a significantly longer period of time to reach its final position (nearly 10 s vs 3 s). Also, the quasi-static controller results in a motion that stops and starts at each position.

To achieve the final controller, we first had to tune the parameters of the controller. Tuning was performed at a static position with the goal of improving accuracy while limiting oscillations. As seen in Fig. 4, while tuning the controller, oscillations arose due to the proportional gain. Adding derivative gain to the controller was unable to eliminate the oscillations because of the 13 Hz frequency for changing neural excitations as well as the muscle activation dynamics. Instead, 120 N-s/m of damping was added to the

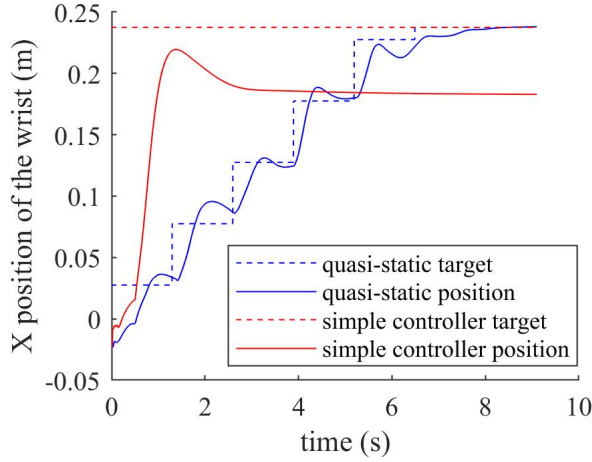


Fig. 3. An example of a time history of the X position of the wrist for a reaching motion controlled by the quasi-static controller (blue) and the simple controller where the target position was always the final goal wrist position (red). The accuracy of the quasi-static controller was 1.1 cm and the accuracy of the simple controller was 27.4 cm.

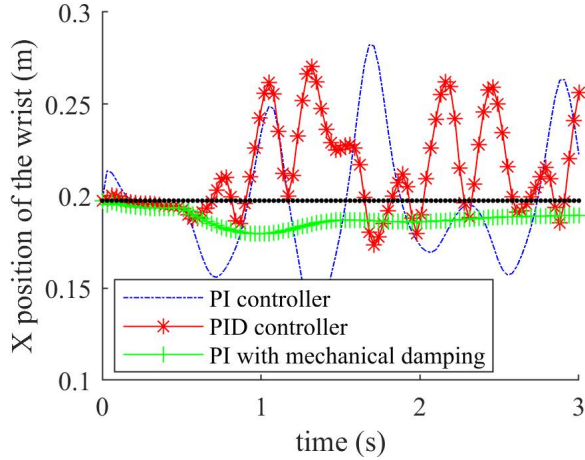


Fig. 4. This figure compares the effect of using a PID controller (red) vs a PI controller with mechanical damping (green). Both controllers use a damping constant of 120 N-s/m. The controller used had a proportional gain of 250 N/m and an integral gain of 80 N/m-s and the goal position was the starting position (black). Adding derivative control is unable to eliminate oscillations compared to the undamped system (blue) because of the time delays in switching neural excitation inputs and the muscle activation dynamics. The addition of physical damping to the arm support eliminates the oscillations.

arm support in each direction. This was effective in limiting oscillations during tuning.

We used the tuned controller to move the wrist over single point reaches of different distances. The average accuracy and rise times of each distance are shown in Table I. Since the rise times were similar for all reach distances, we used the overall average value of 1.3 s as the position switching time, t_{switch} for our quasi-static controller. To select a distance between points, the goal was to select the largest distance possible while maintaining accuracy. With that goal in mind, we selected 6 cm spacing for our paths. (12 cm spacing also produced low error with the point to point moves, however

TABLE I
AVERAGE RESULTS FOR POINT TO POINT REACHES

Reach distance (cm)	Accuracy (cm)	80% rise time (s)
2	2.2	1.2
4	3.5	1.0
6	3.5	1.2
8	4.1	1.4
10	6.6	1.3
12	3.3	1.3
14	5.5	1.5
16	10.3	1.5

a few test trials of paths with 12 cm spacing demonstrated that this accuracy was not maintained over several steps.)

The final parameters selected for the controller were a distance between positions in the path of $d_* = 6$ cm, the switching time of $t_{switch} = 1.3$ s, and the mechanical damping of 120 N-s/m in all directions.

IV. DISCUSSION

We have presented a quasi-static control architecture which is capable of accurately controlling the position of the wrist of a simulated human arm during reaching movements. Using a model-based, quasi-static method proved more accurate than a simple model-based PI controller. The accuracy of the final controller is worse than the accuracy found in [11] for holding static positions but similar. While this is a simulation and so does not have the issues of a practical experiment, maintaining a similar accuracy over the course of an entire reach is encouraging.

This paper determined the key parameters necessary which can be used as a starting point to achieve reaching motions with a practically implemented quasi-static controller. Further tuning of the parameters will be necessary when used with a human subject. A relatively simple improvement prescribed by the results was the addition of a damper to the passive arm support. Oscillation within a reaching motion can be discomforting to an individual with a paralyzed limb controlled with FES. The inability to eliminate oscillations using derivative control due to the delays in the system limited the ability to improve the performance of the controller. More complex methods have been developed to compensate for the delays inherent in an electrically stimulated neuromuscular system, but these are generally developed only for single joints or often require some knowledge (with some uncertainty allowed) of the parameters of the system [14][15]. Our modeling method avoids parametric modeling due to the difficulty in defining the parameters and therefore does not fit these compensation methods well. The physical addition of the damper to the system is a simple way to improve the system performance and achieve similar results.

The average speed of the reaching motions prescribed by this controller (4.6 cm/s) is very slow compared to other controllers. A planar arm controlled by FES has been driven with maximum velocities of 25 m/s [9]. Even the simple controller without intermediate points much more

quickly achieved its final position. Also, the simple controller produced smoother motions without stopping and starting. However, for many every day motions, slow arm movements are acceptable as accuracy is more important (for example, when eating off a plate). However, it is still necessary to improve the speed and smoothness of the movements to make the reaching similar to pre-injury abilities. Since the rise times for all single point-to-point trials were similar, the best way to improve the speed of the controller is to increase the distance between positions in the quasi-static path. However, this change must be done carefully because the increase in distance between points in this paper generally led to a decrease in accuracy.

There were many trials with larger distances that still had very good accuracy (and some small distance trials with bad accuracy). These good accuracy trials often occurred when the path was well within the feasible region. For many of the trials with bad accuracy, points along the path were near the border of the feasible region. This can be seen in Fig. 2 where the controller is accurate until the portion of the path near the boundary of the feasible space where errors in the model are not well compensated for. Model errors on the border of the feasible space could mean the point is actually infeasible or could cause the arm to move to infeasible configuration which cannot be recovered from. Additionally, the ability of the controller to compensate for errors is lower in this region as the muscles are able to achieve less compensatory forces. Also, because the feasible space is not convex, large distances between points can lead to straight line paths which cross through infeasible regions. This is most likely one reason that the smaller distances between points have better accuracy. The current method of selecting a path is a simple nearest neighbor search. A more intelligent path selection method which selected feasible points away from the boundary could allow for larger distances between points and, in turn, higher speed of movement.

Overall, this work presents a quasi-static control architecture capable of achieving FES-driven reaching motions.

V. ACKNOWLEDGMENTS

The authors would like to thank Dr. Ton van den Bogert at Cleveland State University for continuous support and advice in working with the Dynamic Arm Simulator.

REFERENCES

- [1] H.-R. Kobravi and A. Erfanian, "Decentralized adaptive robust control based on sliding mode and nonlinear compensator for the control of ankle movement using functional electrical stimulation of agonist-antagonist muscles," *Journal of Neural Engineering*, vol. 6, no. 4, p. 046007, 2009.
- [2] P. H. Peckham and J. S. Knutson, "Functional Electrical Stimulation for Neuromuscular Applications," *Annual Review of Biomedical Engineering*, vol. 7, no. 1, pp. 327–360, 2005.
- [3] A. B. Ajiboye, F. R. Willett, D. R. Young, W. D. Memberg, B. A. Murphy, J. P. Miller, B. L. Walter, J. A. Sweet, H. A. Hoyen, M. W. Keith, P. H. Peckham, J. D. Simeral, J. P. Donoghue, L. R. Hochberg, and R. F. Kirsch, "Restoration of reaching and grasping movements through brain-controlled muscle stimulation in a person with tetraplegia: a proof-of-concept demonstration," *The Lancet*, vol. 389, no. 10081, pp. 1821–1830, may 2017.
- [4] K. M. Jagodnik and A. J. van den Bogert, "Optimization and evaluation of a proportional derivative controller for planar arm movement," *Journal of Biomechanics*, vol. 43, no. 6, pp. 1086–1091, apr 2010.
- [5] D. Blana, R. F. R. Kirsch, and E. K. E. Chadwick, "Combined feedforward and feedback control of a redundant, nonlinear, dynamic musculoskeletal system," *Medical and Biological Engineering and Computing*, vol. 47, no. 5 SPEC. ISS., pp. 533–542, 2009.
- [6] K. M. Jagodnik, P. S. Thomas, A. J. Van Den Bogert, M. S. Branicky, and R. F. Kirsch, "Training an Actor-Critic Reinforcement Learning Controller for Arm Movement Using Human-Generated Rewards," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 10, pp. 1892–1905, oct 2017.
- [7] L. Lan, K. Y. Zhu, and C. Y. Wen, "Design of a Threshold FES Control System for Arm Movement," *Journal of Mechanics in Medicine and Biology*, vol. 09, no. 04, pp. 449–479, dec 2009.
- [8] C. T. Freeman, "Upper Limb Electrical Stimulation Using Input-Output Linearization and Iterative Learning Control," *Control Systems Technology, IEEE Transactions on*, vol. PP, no. 99, p. 1, 2014.
- [9] R. Sharif Razavian, B. Ghannadi, N. Mehrabi, M. Charlet, and J. McPhee, "Feedback Control of Functional Electrical Stimulation for 2-D Arm Reaching Movements," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 10, pp. 2033–2043, oct 2018.
- [10] D. N. Wolf and E. M. Scheerer, "Evaluating an open-loop functional electrical stimulation controller for holding the shoulder and elbow configuration of a paralyzed arm," in *2017 International Conference on Rehabilitation Robotics (ICORR)*. IEEE, jul 2017, pp. 789–794.
- [11] —, "Holding Static Arm Configurations With Functional Electrical Stimulation: A Case Study," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 26, no. 10, pp. 2044–2052, oct 2018.
- [12] E. M. Scheerer, Y.-W. Liao, E. J. Perreault, M. C. Tresch, W. D. Memberg, R. F. Kirsch, and K. M. Lynch, "Semiparametric Identification of Human Arm Dynamics for Flexible Control of a Functional Electrical Stimulation Neuroprosthesis," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 12, pp. 1405–1415, dec 2016.
- [13] D. N. Wolf and E. M. Scheerer, "Simple quasi-static control of functional electrical stimulation-driven reaching motions," in *20019 9th International IEEE/EMBS Conference on Neural Engineering*. IEEE, in press.
- [14] N. Alibei, N. Kirsch, S. Farrokhi, and N. Sharma, "Further Results on Predictor-Based Control of Neuromuscular Electrical Stimulation," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 23, no. 6, pp. 1095–1105, 2015.
- [15] P. Cooman and R. F. Kirsch, "Control of a time-delayed 5 degrees of freedom arm model for use in upper extremity functional electrical stimulation," in *2012 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*. IEEE, aug 2012, pp. 322–324.
- [16] E. K. Chadwick, D. Blana, R. F. Kirsch, and A. J. Van Den Bogert, "Real-Time Simulation of Three-Dimensional Shoulder Girdle and Arm Dynamics," *IEEE Transactions on Biomedical Engineering*, vol. 61, no. 7, pp. 1–10, 2014.
- [17] S. L. Delp, F. C. Anderson, A. S. Arnold, P. Loan, A. Habib, C. T. John, E. Guendelman, and D. G. Thelen, "OpenSim: Open-source software to create and analyze dynamic simulations of movement," *IEEE Transactions on Biomedical Engineering*, vol. 54, no. 11, pp. 1940–1950, 2007.
- [18] A. Seth, J. L. Hicks, T. K. Uchida, A. Habib, C. L. Dembia, J. J. Dunne, C. F. Ong, M. S. DeMers, A. Rajagopal, M. Millard, S. R. Hamner, E. M. Arnold, J. R. Yong, S. K. Lakshminanth, M. A. Sherman, J. P. Ku, and S. L. Delp, "OpenSim: Simulating musculoskeletal dynamics and neuromuscular control to study human and animal movement," *PLOS Computational Biology*, vol. 14, no. 7, p. e1006223, jul 2018.
- [19] G. Wu, F. C. Van Der Helm, H. E. Veeger, M. Makhsous, P. Van Roy, C. Anglin, J. Nagels, A. R. Karduna, K. McQuade, X. Wang, F. W. Werner, and B. Buchholz, "ISB recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion - Part II: Shoulder, elbow, wrist and hand," *Journal of Biomechanics*, vol. 38, no. 5, pp. 981–992, 2005.
- [20] C. Rasmussen and C. Williams, *Gaussian processes for machine learning*, 2006.