Proceedings of IMECE 2008
ASME International Mechanical Engineering Congress and Exposition
October 31 - November 6, 2008, Boston, Massachusetts

IMECE2008-68234-DRAFT

COMPLIANT CONTROL OF POST-STROKE REHABILITATION ROBOTS: USING MOVEMENT-SPECIFIC MODELS TO IMPROVE CONTROLLER PERFORMANCE

Giulio Rosati*
Dept. of Innovation in Mechanics and Management
Faculty of Engineering - University of Padua
via Venezia, 1 - 35131 Padova, Italy
giulio.rosati@unipd.it

James E. Bobrow, David J. Reinkensmeyer
Dept. of Mechanical and Aerospace Engineering
Henry Samueli School of Engineering
4200 Engineering Gateway Building
University of California, Irvine
Irvine, CA 92697-3975, USA
dreinken@uci.edu

ABSTRACT

Post-stroke neurorehabilitation is an emerging application field of robotics, aiming to design new treatment systems and protocols based on the use of robotic technology and virtual reality to improve patient recovery after stroke. One goal in this field is to develop robotic therapy devices that are compliant but can still assist weakened patients in making desired movements. It is hypothesized that, in this way, the interaction with the robotic system can maintain patient engagement and effort, and promote and stimulate the motor learning process of the patient. One way that has been proposed to maintain compliance while assisting weak patients is to use an adaptive controller with a forgetting term, which allows the robotic system to learn a model of the forces needed to assist the patients during exercises while encouraging patient effort. A limitation of such an approach is that the adaptive gain must be large enough to rapidly change the model for different target movements, which decreases the compliance of the robot. We show here in simulation that by building independent models for different target movements, robot compliance can be increased while still accurately achieving the target movements.

INTRODUCTION

Stroke is a leading cause of movement disability in the U.S. and Europe [1]. Improving treatment of movement disability after stroke is a societal goal of major importance that is receiving intense interest from many domains of medical and engineering research, in part because the number of people requiring movement training after stroke is large and increasing, but also because there is increasing evidence that the motor system is plastic following stroke and can be influenced by motor training [2]. One of the domains being explored is robotic therapy [3]. Robotic devices have the potential to help automate repetitive training after stroke in a controlled fashion. One paradigm being explored is to develop robotic devices that attach to the limbs of patients and help them reproduce desired movements as they interact with computer games or in a virtual environment. Initial clinical results with this approach are positive but modest [4]. Patients can significantly improve their movement ability with training on such devices, but the improvements typically produce only a small change in functional ability, if any. Results from applying this approach to patients in the sub-acute phase of stroke appear better [5–7], perhaps because the brain has added capacity for plasticity earlier after stroke [8]. However, in both the subacute and chronic phases an important goal is to try to improve the benefits of robotic therapy, by building on the initially positive results [9].

Two potential ways to improve the effectiveness of robotic therapy are 1) to build robotic devices that allow more natural movements and 2) to build robotic devices that are more compliant when they assist patients in moving. The rationale for building robotic devices that allow more natural movements is

*Address all correspondence related to this paper to this author.

that motor training shows specificity of learning; that is, people improve most at the movements they practice [10]. If the goal is to have people improve in their ability to make functional movements, then it would seem best to have patients practice functional movements. But functional movements typically use a large number of degrees of freedom of the arm and hand, thus requiring the development of more sophisticated, multiple degrees-of-freedom robotic therapy devices. Recent examples of device development along this line include [7, 11–14]

The second potential approach to improve robotic therapy is to make robotic assistance during movement training more compliant. Compliance has long been recognized as a desirable feature for robotic therapy, for example to promote safe human-robot interactions [15]. Another rationale for using compliant robotic devices is that compliance preserves the causal relationship between patient effort and resulting arm movement, even when robotic assistance is provided. If the patient has the ability to influence the way an ongoing movement occurs, this may encourage patient engagement and effort. For example, a study of patient effort when training in a non-compliant robotic gait training device found that the patient consumed less energy than when compared with training with the compliant arms of a human therapist [16]. Compliance may also help stimulate the motor learning process, since compliance allows patients to make movement errors, and errors drive the motor learning process [10, 17, 18].

Note that these two goals - to make robots that allow more naturalistic movement and to make robots that are also compliant - are in some ways at odds with each other. It is straightforward to assist patients in making movements in a wide workspace against gravity using a relatively stiff robot. However, assisting in such movements using a compliant robot requires more complex control strategies that model the forces applied by the environment to the arm. This paper describes a method for improving the model-based, compliant control of a rehabilitation robot.

ADAPTIVE CONTROL WITH FORGETTING

A recently proposed method for maintaining compliance while assisting post-stroke patients in achieving desired arm movements is to use a standard adaptive controller that has been modified with a forgetting term [14, 19]. The adaptive controller uses a measurement of tracking error to build a model of the forces needed to assist the arm in moving. In the implementation of [14], the model is represented as a function of the position of the arm, using radial basis functions whose parameters are updated with a standard adaptive control law.

Building a model of the forces needed to move the arm allows the robot to be made more compliant, since the robot need no longer rely solely on position feedback to decrease tracking error. However, [14] found that when stroke patients interacted with the standard adaptive controller, they tended to allow the controller to take over, reducing their own effort. This problem was addressed by modifying the standard adaptive controller to include a forgetting term that continuously attempts to reduce the assistance forces from the robot. Essentially, the resulting controller models the forces needed to assist the patient, as learned from tracking errors, and reduces its effort with time on an exponential basis.

Specifically, the control method provides an assistive robot force that is the sum of the force from the model and a proportional derivative controller according to:

$$\mathbf{F}_r = \mathbf{Y}\hat{\mathbf{a}} - \mathbf{K}_P(\mathbf{x} - \mathbf{x_d}) - \mathbf{K}_D(\dot{\mathbf{x}} - \dot{\mathbf{x}}_\mathbf{d}) = \mathbf{Y}\hat{\mathbf{a}} - \mathbf{K}_P\tilde{\mathbf{x}} - \mathbf{K}_D\dot{\tilde{\mathbf{x}}} \quad (1)$$

where $\mathbf{x}$ and $\mathbf{x_d}$ are the current and the desired (planned) location of patient's hand, $\mathbf{K}_P$ and $\mathbf{K}_D$ are the proportional and derivative gain matrices, whereas the regressor matrix $\mathbf{Y}$ (with p grid points, n degrees of freedom) is

$$\mathbf{Y}^{n\times(np)} = \left[\begin{array}{c|c|c} \mathbf{g}^T & \cdots & 0 \\ \hline 0 & \ddots & 0 \\ \hline 0 & \cdots & \mathbf{g}^T \end{array}\right] \quad (2)$$

where $\mathbf{g}$ is the $(p \times 1)$ vector of Gaussian radial basis functions, defined as

$$g_i(\mathbf{x}) = \exp(-\|\mathbf{x} - \mu_i\|^2 / 2\sigma^2) \quad (3)$$

where g_i is the $i-th$ radial basis function, μ_i is the location of the $i-th$ function and σ is a scalar smoothing constant that determines the width of the basis function. In this way, the regressor matrix is a function of position only. The magnitudes of the radial basis functions are updated as follows

$$\dot{\hat{\mathbf{a}}} = -\frac{1}{\tau}\mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-1}\mathbf{Y}\hat{\mathbf{a}} - \Gamma^{-1}\mathbf{Y}^T\mathbf{s} \quad (4)$$

where the *sliding surface* $\mathbf{s}$ is given by

$$\mathbf{s} = \dot{\tilde{\mathbf{x}}} + \Lambda\tilde{\mathbf{x}} \quad (5)$$

The first term of equation (4) implements an exponential decay in the radial basis function amplitudes with forgetting time constant τ, and therefore decays the assisting force from the model when tracking error is small. The second term of equation (4) adjusts the radial basis function amplitudes based on tracking errors with adaptive gain Γ, in order to construct the representation of the forces needed to assist in achieving accurate arm movements.

Limitations to Achieving Compliance with Adaptive Control

A limitation of such an approach is that the adaptive gain must be large enough to rapidly change the model for different target movements, which effectively decreases the compliance of the robot. In particular, the effective stiffness G_P of the controller can be calculated as follows:

$$\mathbf{G}_P = \frac{1}{\tau}\mathbf{K}_D + \Gamma^{-1}\mathbf{Y}\mathbf{Y}^T + \mathbf{K}_P \quad (6)$$

which is obtained by substituting (4) into the derivative of (1) (see [14] for details).

From this equation, it can be seen that the parameters of the adaptive controller, including the adaptive update rate Γ and the forgetting time constant τ, influence the effective stiffness of the controller.

Note that the required assisting forces change from movement to movement. For example, the assisting forces depend on the movement direction. The adaptive controller must therefore update the model of the forces relatively quickly when movement direction changes in order to still achieve accurate movements. A faster parameter update rate Γ translates into a higher effective stiffness, as equation (6) shows. Thus, with this controller, there is an inherent tradeoff between achieving accurate movements and maintaining compliance.

In this paper, two improvements of the assist-as-needed adaptive control algorithm are presented that address this tradeoff by using movement-specific models. The results with the improved controllers are compared to the original control system in simulations.

MULTIPLE a-VECTOR ADAPTIVE CONTROL

A first modification of the control algorithm consists in using independent $\hat{\mathbf{a}}$ vectors of parameter estimates for different target movements. In this way, independent models of assistance force are built for different portions of the trajectory. This choice leads to parameter estimates that become more stabilized with time, and whose variation with time can be controlled by choosing a proper forgetting time constant for the exponential decay.

In the original version of the controller developed by [14]., the regressor matrix $\mathbf{Y}$ is a function of position only. For this reason, time-dependent changes of the adaptive force in a specific target position can be obtained only through a variation of the model, which in turn can be yielded by a suitably high Γ gain. This is the case, for example, of an exercise in which the patient is requested to reach multiple targets from the same starting point: since the target trajectory passes through the starting point several times requiring different robot forces, the parameter estimates in that region of the workspace must change during the execution of each repetition of the exercise. So, we can say that the robot cannot learn a single model of the forces required to assist the exercise in that point of the workspace.

To illustrate this issue, we simulated a task in which the robot must repeatedly move a $1kg$ mass back and forth along a horizontal trajectory in the presence of gravity. The mass models the combined mass of the robot and the patient's arm, and the patient is assumed for simplicity to be completely paralyzed: i.e. the robot must learn how to move the mass on its own. We assumed the desired trajectory is a minimum jerk trajectory starting at $P_1 = (0,0)$ and ending at $P_2 = (0.25m, 0)$, and that the forward motion lasts $3s$, and the backward motion between the same two points lasts $2s$. Control gains were chosen to be $K_P = 10N/m$, $K_D = 35Ns/m$ and $\Gamma = 0.01m/N$. Figure 1 shows the evolution of robot force and of a-vector parameters after the robot has performed the movement 15 times, continuously updating the model. The plot show the forces and a-vector parameters in the x (horizontal) direction, and were obtained with no forgetting ($\tau \to \infty$). The top plot depicts the force required to complete the task (black line), the adaptive force (green line), the PD force (red line) and the total force produced by the robot (blue line). The bottom plot depicts the evolution of the adaptive parameters with time during a single movement, showing that the controller must continuously vary the model of the forces during the movement in order to follow the target trajectory.

By splitting the trajectory into several point-to point movements and by using independent parameter estimates for each segment of the trajectory, we can get a stabilization of the parameter estimates for each trajectory segment (i.e., the robot builds an independent stable representation of the assistance

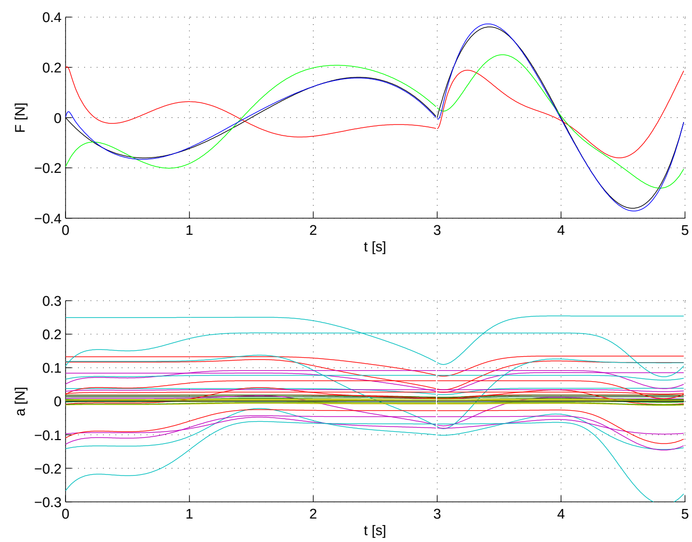

Figure 1. ADAPTIVE FORCE IN THE x DIRECTION (TOP) AND CORRESPONDING PARAMETERS (BOTTOM) VERSUS TIME: SINGLE-a CONTROLLER.

force model for each target movement). Figure 2 shows how this concept can be applied to the sample trajectory of figure 1: after 15 movements, two models are built by the adaptive controller, one for the forward motion and one for the backward motion. The parameter estimates shown in the bottom plot do not vary as much during the movement using this approach.

A drawback of this approach is that a discontinuity in the adaptive force $\mathbf{F}_r$ is produced every time a switch of the parameter estimate vector $\hat{\mathbf{a}}$ occurs (in the case of figure 2, this happens for $t = 0$ and $t = 3s$). This problem can be solved by adding a correction term to the new parameters before switching the vectors. Let $\hat{\mathbf{a}}_i$ be the parameter vector for the $i-th$ target movement. The correction force needed to compensate for the discontinuity in the transition between the $i-th$ and the $(i+1)-th$ target movements is

$$\mathbf{F}_{corr} = \mathbf{Y}\hat{\mathbf{a}}_i - \mathbf{Y}\hat{\mathbf{a}}_{i+1} \tag{7}$$

This amount of additional force can be produced by adding to vector $\hat{\mathbf{a}}_{i+1}$ the following correction term

$$\hat{\mathbf{a}}_{corr} = \mathbf{Y}^T(\mathbf{Y}\mathbf{Y}^T)^{-1}\mathbf{F}_{corr} \tag{8}$$

which is the minimum norm solution for equation $\mathbf{Y}\hat{\mathbf{a}}_{corr} = \mathbf{F}_{corr}$, calculated by using the pseudo-inverse of the regressor matrix $\mathbf{Y}$. In this way, the discontinuity is completely eliminated. This result is shown in Figure 3. It is to be noticed that the norm of vector $\hat{\mathbf{a}}_{corr}$ tends to zero after very few repetitions of the trajectory.

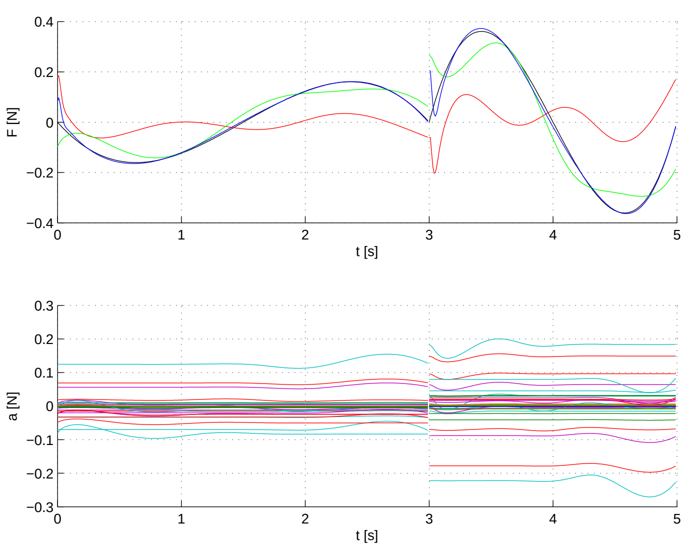

Figure 2. ADAPTIVE FORCE IN THE *x* DIRECTION (TOP) AND CORRESPONDING PARAMETERS (BOTTOM) VERSUS TIME: MULTIPLE-a CONTROLLER.

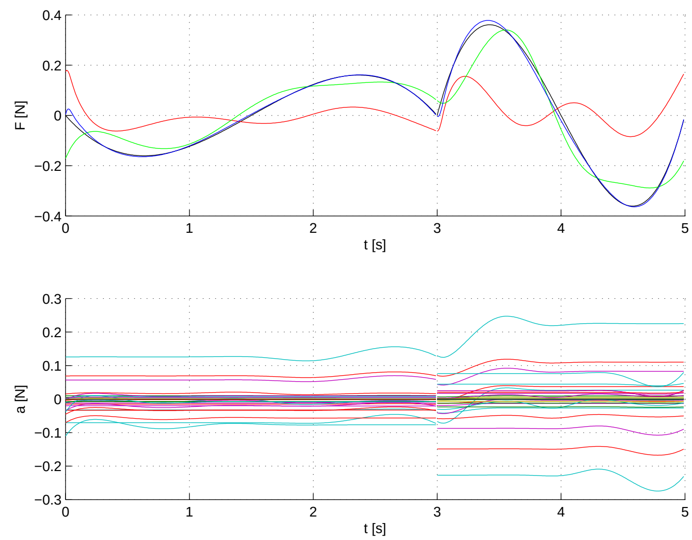

Figure 3. ADAPTIVE FORCE IN THE *x* DIRECTION (TOP) AND CORRESPONDING PARAMETERS (BOTTOM) VERSUS TIME: MULTIPLE-a CONTROLLER WITH THE CORRECTION TERM.

TIME-BASED ADAPTIVE CONTROL

Another way of improving the stabilization of parameter estimates is to employ a time-based Gaussian basis functions grid, so that the weight of each parameter changes with time rather than with end-effector position. To achieve this result, the following functions are used instead of the ones introduced in (3):

$$\hat{g}_i(t) = \exp(-\|t - t_i\|^2 / 2\sigma^2) \tag{9}$$

where t_i is the location (instant) of the $i-th$ Gaussian function. In this way, if the trajectory involves multiple positioning in the same point of the workspace, a different set of parameters will be employed to generate the adaptive force each time the target position is reached, since the target point is reached in a different instant (i.e. in a different location within the Gaussian basis functions grid). This kind of approach is well suited also for those parts of the trajectory in which end-effector velocity is very low. In this case, in fact, the Cartesian grid would require a change in parameter estimates to achieve a change in robot force. On the contrary, by using the time-based grid new parameters will be employed to generate the force as far as time changes, even if position does not. The final result is that the assistance force model does not need to change with time once learned (see Figure 4).

This approach can produce a discontinuity in robot force every time the trajectory is completed and a new repetition of the exercise starts. In fact, at the end of the trajectory the last components of vector $\hat{\mathbf{a}}$ are employed to generate the force, whereas at the beginning of the next cycle the adaptive force is provided by the first ones [1]. To overcome this problem, a modified set of time-based Gaussian functions $\tilde{g}_i$ can be employed:

$$\tilde{g}_i(t) = \begin{cases} \hat{g}_i(t) + \hat{g}_i(t+T) & 0 \le t \le \alpha T \\ \hat{g}_i(t) & \alpha T \le t \le (1-\alpha)T \\ \hat{g}_i(t) + \hat{g}_i(t-T) & (1-\alpha)T \le t \le T \end{cases} \quad (10)$$

where T is trajectory duration (cycle time) and $0 \le \alpha \le 0.5$ is a scalar factor that determines the amount of overlapping of the Gaussian functions at the boundaries of the cycle. For $\alpha = 0$, no overlapping is obtained and the discontinuity in robot force still occurs. For $\alpha > 0$, the first and last parameter estimates are employed together to calculate robot force at the beginning and at the end of the trajectory, producing a sort of continuity in the time-based grid. The amount of overlapping must be adjusted according to the value of the σ parameter: the higher the σ, the lower α can be employed.

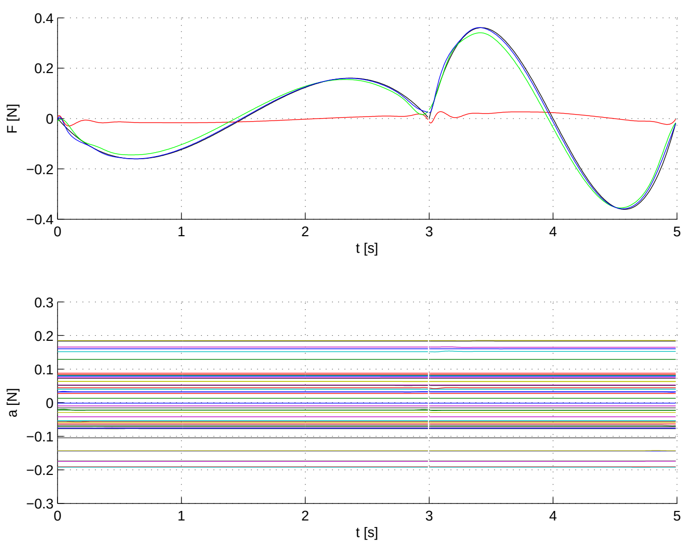

Figure 4. ADAPTIVE FORCE IN THE x DIRECTION (TOP) AND CORRESPONDING PARAMETERS (BOTTOM) VERSUS TIME: TIME-BASED CONTROLLER WITH THE MODIFIED GAUSSIAN FUNCTIONS.

[1]This kind of situation cannot be observed if the Cartesian functions grid is employed, since no position discontinuity can be produced at end-effector level during robot motion.

EFFECT OF USING MOVEMENT-SPECIFIC MODELS ON ROBOT COMPLIANCE

We analyzed the effect of the two improvements described above on the robot compliance needed to produce accurate tracking. The target trajectory was the same used in the previous sections (i.e. back-and-forth point-to-point motions of a $1kg$ mass in the horizontal direction). A grid with 49 points was employed both with the Cartesian-based controllers (7×7 equally spaced grid points in the xy plane, where x is horizontal and y is vertical) and with the time-based controller (49 equally spaced points in the time interval $[0, 5s]$). For the following figures, we allowed the controller to adapt for 15 movement repetitions.

We varied the controller parameters and plotted effective stiffness of the controller versus tracking error for each set of parameters. Specifically, we performed simulations with different Γ gains and different values of σ and τ, keeping all other control system parameters constant (in particular, $K_P = 10N/m$ and $K_D = 35Ns/m$). As all the gain matrices of the controller were chosen diagonal, the effective controller stiffness along each degree of freedom can be written as

$$G_P = \frac{1}{\tau} K_D + \gamma \mathbf{g}^T \mathbf{g} + K_P \quad (11)$$

where K_P, K_D and $\gamma = \frac{1}{\Gamma}$ are the scalar control gains for the considered degree of freedom [2].

In the next figures, two tracking performance parameters are plotted versus robot effective stiffness G_P:

- the absolute mean deviation of the x component of sliding surface $\mathbf{s}$

$$s_x = \sqrt{\frac{1}{N} \sum_{h=1}^{N} \{\mathbf{s}_h\}_x^2} \quad (12)$$

  where $\mathbf{s}_h$ is the $h-th$ sample of $\mathbf{s}$, whereas N is the number of samples considered in the simulation of each repetition of the target trajectory; this parameter entails the ability of the controller in following the trajectory against dynamic loads;
- the mean y coordinate of the end effector m_y, which describes the ability of the controller in maintaining the end effector in horizontal position against gravity.

Figure 5 plots the mean vertical tracking error m_y versus G_P. Robot effective stiffness varied in the range [40 325] N/m when we varied $\gamma \in [20, 100]\, N/m$, $\tau \to \infty$ and $\sigma = \alpha \Delta x$, where $\alpha \in [0.75, 1]$ and Δx is the spacing of the basis functions grid. Square spots are single-a controller simulations, round spots are results from the multiple-a controller and crosses are results from

[2]Also a change in σ parameter affects robot effective stiffness. In fact, larger values of σ yield larger values of product $\mathbf{g}^T \mathbf{g}$ in (11).

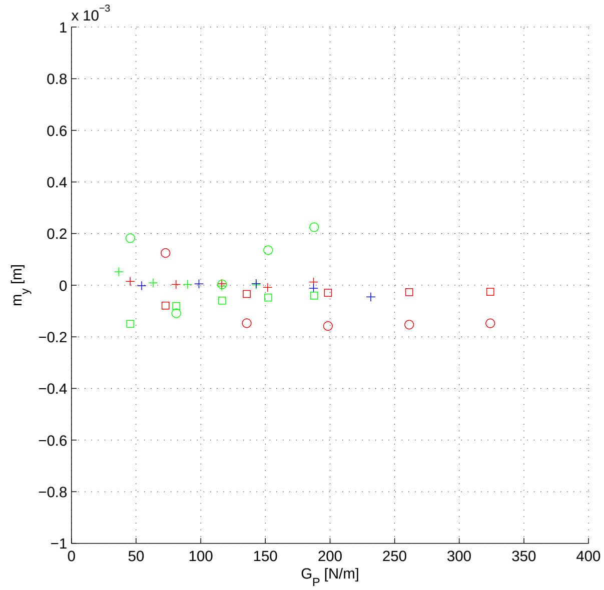

Figure 5. MEAN y POSITION OF THE END-EFFECTOR VERSUS G_P, WITH $\tau \to \infty$. TRACKING ACCURACY IS NEITHER AFFECTED BY ROBOT STIFFNESS NOR BY THE CONTROL ALGORITHM, SINCE THE FORCE TO BE LEARNED IS A CONSTANT (I.E. THE FORCE DUE TO GRAVITY ON THE MASS).

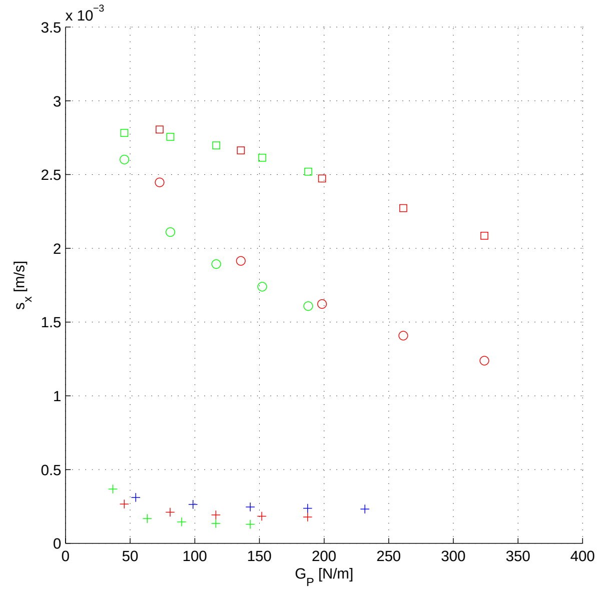

Figure 6. HORIZONTAL TRACKING ERROR (x COMPONENT) VERSUS EFFECTIVE ROBOT STIFFNESS G_P, WITH $\tau \to \infty$. ACCURACY INCREASES WITH G_P FOR ALL CONTROL ALGORITHMS, WHILE TIME-BASED CONTROL (CROSSES) IS BETTER THAN THE MULTIPLE-A (ROUND SPOTS) CONTROLLER, WHICH IS BETTER THAN SINGLE-A CONTROLLER (SQUARE SPOTS).

the time-based controller. It is clear that neither robot stiffness nor the control algorithm chosen affected tracking error in the y direction, which was less than 0.02 cm for all simulations. This is because the force that the controller was required to learn in the vertical direction was a constant force (gravity). All three controllers learned this force accurately.

Figure 6 plots the horizontal tracking error s_x versus G_P for the same simulations of Figure 5, showing that tracking accuracy against dynamic loads can be increased by increasing G_P or by introducing the modifications to the original control algorithm. In particular, the multiple-a controller produces better tracking at lower stiffness than the single-a controller, and the time-based controller is better than the multiple-a controller. As a result, by using the modified control algorithms robot compliance can be increased while still accurately achieving target movements. The results are similar in the case when forgetting is included in the controller ($\tau = 2s$, Figure 7), except total error increases for all three controllers due to the forgetting.

Figure 8 plots the vertical tracking error versus effective robot stiffness for the same simulations as in Figure 7. This figure shows that, with the forgetting term ($\tau = 2s$), tracking accuracy in the y direction is a function of G_P, independent of the control algorithm used. Adding forgetting increases the vertical tracking error, but the amount of increase depends on the effective stiffness of the controller, not on the type of model it uses.

CONCLUSION

This paper described a method for improving model-based adaptive control of compliant rehabilitation robots. The approach is to learn a model of the required assistive forces for each target movement, either by breaking the target movements into different spatial segments, or by representing the assistive forces as a function of time. We showed that this approach allows the robot to be more compliant while achieving the same amount of tracking error. This improvement in the compliance/tracking tradeoff occurs only in movement directions for which there are dynamically-changing forces, such as in the direction of movement itself.

The multiple model approach is more effective than the single model approach used here because the single model approach uses a position-dependent model, but the forces required to move the arm depend not only on position but also on velocity and acceleration. Thus, a single model learned for one robot position will not be appropriate for different movements passing through that position. As a result, the single model approach requires a high adaptive update rate to rapidly change the model for different movements, which in turn increases the effective stiffness of the controller. Using different models to assist in different movements allows the adaptive update rate to be lowered, which translates into increased compliance while still achieving the same tracking error.

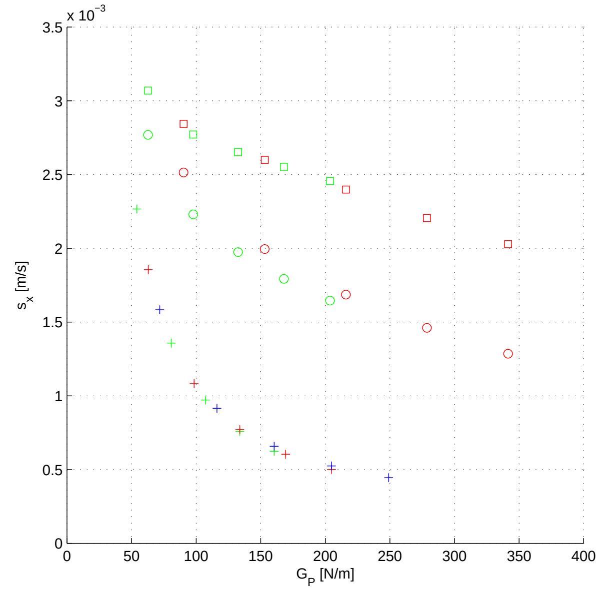

Figure 7. HORIZONTAL TRACKING ERROR (x COMPONENT) VERSUS EFFECTIVE ROBOT STIFFNESS G_P, WITH $\tau = 2s$. ACCURACY INCREASES WITH G_P FOR ALL CONTROL ALGORITHMS, WHILE TIME-BASED CONTROL (CROSSES) IS BETTER THAN MULTIPLE-A CONTROLLER (ROUND SPOTS), WHICH IS BETTER THAN SINGLE-A CONTROLLER (SQUARE SPOTS).

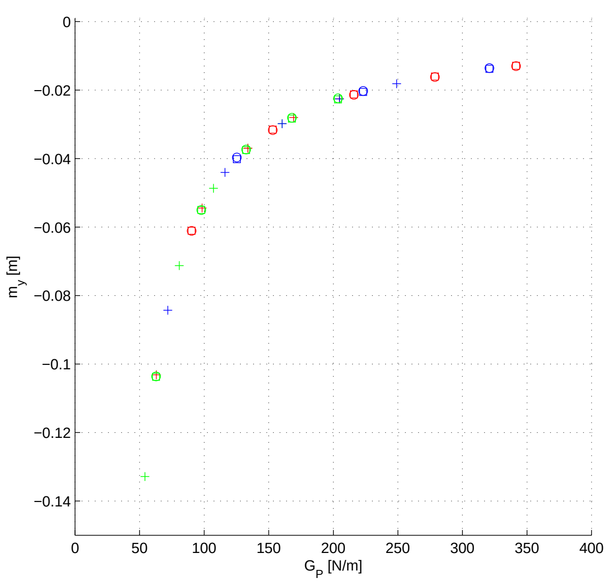

Figure 8. VERTICAL TRACKING ERROR y VERSUS EFFECTIVE ROBOT STIFFNESS G_P, WHEN THE ADAPTIVE CONTROLLER INCLUDES FORGETTING ($\tau = 2s$). TRACKING ACCURACY IS A FUNCTION OF ROBOT STIFFNESS ONLY.

An alternate approach to those studied here would be to use a single model, but to represent the model as a function of velocity and acceleration as well as position. This would make more sense for a robot whose parameters do not vary much. Stroke patients, on the contrary, need a time-varying assistive force that is trajectory dependent and can not be parameterized easily in some other way like mass and damping can. Two other possible drawbacks of this approach would be the need for accurate velocity and acceleration signals, and a resulting exponential increase in model size with the number of dimensions represented. The use of movement-specific models increases the number of model parameters (but not the number of grid points and the size of the regressor matrix) linearly in the number of movements represented, or linearly in the total time duration of the movements for the time-based model. This approach is thus practical in terms of the total parameter size that would be required for rehabilitation exercises, in which a fixed sequence of a limited number of movements is typically prescribed.

The ultimate validation of this adaptive controller will depend on rigorous clinical testing with stroke patients. A key hypothesis to be evaluated is ”Does increased compliance during robot-assisted movement training result in better clinical rehabilitation outcomes?” The controller described here provides a method for achieving lower compliance and thus will allow this hypothesis to be better evaluated.

ACKNOWLEDGMENT

This work was supported in part by NIH N01-HD-3-3352. Thanks go to R. Secoli and F. Milani for their contribution to the development of the software simulation tool.

REFERENCES

[1] Rosamond, W., Flegal, K., et al., 2007. “Heart disease and stroke statistics-2007 update: A report from the american heart association statistics committee and stroke statistics subcommittee”. *Circulation*, **115**, pp. 69–171.

[2] Nudo, R. J., 2007. “Postinfarct cortical plasticity and behavioral recovery”. *Stroke*, **38**(2), pp. 840–845.

[3] Harwin, W. S., Patton, J. L., and Edgerton, V. R., 2006. “Challenges and opportunities for robot-mediated neurorehabilitation”. *Proceedings of the IEEE*, pp. 1717–1726.

[4] Kwakkel, G., Kollen, B. J., and Krebs, H. I., 2007. “Effects of robot-assisted therapy on upper limb recovery after stroke: A systematic review”. *Neurorehabilitation and Neural Repair*, **22**, pp. 111–121.

[5] Hesse, S., Werner, C., Pohl, M., Rueckriem, S., Mehrhoz, J., and Lingnau, M. L., 2005. “Computerized arm training improves the motor control of the severely affected arm after stoke”. *Stroke*, **36**, pp. 1960–1966.

[6] Rosati, G., Gallina, P., and Masiero, S., 2007. "Design, implementation and clinical tests of a wire-based robot for neurorehabilitation". *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* **15**(4), pp. 560–569.
[7] Masiero, S., Celia, A., Rosati, G., and Armani, M., 2007. "Robotic-assisted rehabilitation of the upper limb after acute stroke". *Archives of Physical Medicine and Rehabilitation,* **88**(2), pp. 142–149.
[8] Biernaskie, J., Chernenko, G., and Corbett, D., 2004. "Efficacy of rehabilitative experience declines with time after focal ischemic brain injury". *Journal of Neuroscience,* **24**(5), pp. 1245–1254.
[9] Reinkensmeyer, D. J., Galvez, J. A., Marchal, L., Wolbrecht, E. T., and Bobrow, J. E., 2007. "Some key problems for robot-assisted movement therapy research: a perspective from the University of California". In Proceedings of the IEEE 10th Int. Conf. on Rehabilitation Robotics ICORR2007, pp. 1009–1015.
[10] Schmidt, R. A., 1988. *Motor Control and Learning*. Human Kinetics Publishers.
[11] Mayr, A., Kofler, M., and Saltuari, L., 2008. "ARMOR: an electromechanical robot for upper limb training following stroke. a prospective randomised controlled pilot study". *Handchir Mikrochir Plast Chir,* **40**(1), pp. 66–73.
[12] Nef, T., and Riener, R., 2005. "ARMin - design of a novel arm rehabilitation robot". In Proceedings of the IEEE 9th International Conference on Rehabilitation Robotics ICORR2005, pp. 57–60.
[13] Sugar, T. G., He, J., Koeneman, E. J., Koeneman, J. B., Herman, R., Huang, H., Schultz, R. S., Herring, D. E., Wanberg, J., Balasubramanian, S., Swenson, P., and Ward, J. A., 2007. "Design and control of RUPERT: a device for robotic upper extremity repetitive therapy". *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* **15**(3), pp. 336–346.
[14] Wolbrecht, E. T., Chan, V., Reinkensmeyer, D. J., and Bobrow, J. E., 2008. "Optimizing compliant, model-based robotic assistance to promote neurorehabilitation". *IEEE Transactions on Neural Systems and Rehabilitation Engineering*. in press.
[15] Krebs, H. I., Hogan, N., Aisen, M. L., and Volpe, B. T., 1998. "Robot-aided neurorehabilitation". *IEEE Transactions on Rehabilitation Engineering,* **6**(1), pp. 75–87.
[16] Israel, J. F., Campbell, D. D., Kahn, J. H., and Hornby, T. G., 2006. "Metabolic costs and muscle activity patterns during robotic- and therapist-assisted treadmill walking in individuals with incomplete spinal cord injury". *Physical Therapy,* **86**(11), pp. 1466–78.
[17] Cai, L. L., Fong, A. J., Otoshi, C. K., Liang, Y. Q., Cham, J. G., Zhong, H., Roy, R. R., Edgerton, V. R., and Burdick, J. W., 2005. "Effects of consistency vs. variability in robotically controlled training of stepping adult spinal mice". In Proceedings of the IEEE 9th International Conference on Rehabilitation Robotics ICORR2005, pp. 575–579.
[18] Emken, J. L., Benitez, R., Sideris, A., Bobrow, J. E., and Reinkensmeyer, D. J., 2007. "Motor adaptation as a greedy optimization of error and effort". *Journal of Neurophysiology,* **97**, pp. 3997–4006.
[19] Wolbrecht, E. T., Chan, V., Le, V., Cramer, S. C., Reinkensmeyer, D. J., and Bobrow, J. E., 2007. "Real-time computer modeling of weakness following stroke optimizes robotic assistance for movement therapy". In Proceedings of the 3rd International IEEE/EMBS Conference on Neural Engineering, pp. 152–158.