An Adaptive Automated Robotic Task-Practice System for Rehabilitation of Arm Functions After Stroke

Younggeun Choi, *Student Member, IEEE*, James Gordon, Duckho Kim, and Nicolas Schweighofer

***Abstract*—We present a novel robotic task-practice system, i.e., adaptive and automatic presentation of tasks (ADAPT), which is designed to enhance the recovery of upper extremity functions in patients with stroke. We designed ADAPT in accordance with current training guidelines for stroke rehabilitation; ADAPT engages the patient intensively, actively, and adaptively in a variety of realistic functional tasks that require reaching and manipulation. A general-purpose robot simulates the dynamics of the functional tasks and presents these functional tasks to the patient. A novel tool-changing system enables ADAPT to automatically switch between the tools corresponding to the functional tasks. The control architecture of ADAPT is composed of three main components: a *high-level task scheduler*, a *functional task model,* and a *low-level admittance controller*. The *high-level task scheduler* adaptively selects the task to practice and sets the task difficulty based on the previous performance of the patients. The *functional task model* generates desired trajectories based on learned models of task dynamics. Tasks dynamics are modeled with receptive field weighted regression (RFWR), such that the feel of the task tools is accurately modeled, and the task difficulty can be easily adjusted. The *low-level admittance controller,* which is also learned with RFWR, implements the selected task trajectory for robot–patient interaction. The results of a preliminary experiment with a healthy subject demonstrate the successful operation of ADAPT.**

***Index Terms*—Admittance controller, locally weighted learning, motor schedule, neurorehabilitation, rehabilitation robotics, stroke, upper extremity.**

I. Introduction

STROKE is the leading cause of disability among American adults. Over 80% of first-time strokes (infarctions only) involve acute hemiparesis of the upper limb [1]. Because a substantial number of activities of daily living involve the use of the upper extremities [2], rehabilitation of reach and grasp skills is critical for patients in their attempts to return to a reasonable quality of life [3].

Recognizing that intensive motor practice is beneficial for recovery of upper extremity functions [4]–[6], and that current medical practice does not adequately allow for the required training intensity [7], [8], a growing number of investigators have been developing robotic systems for the rehabilitation of upper extremities after stroke. In a number of these systems, such as the MIT-MANUS [9], the mirror-image motion enabler robot (MIME) [10], the assisted rehabilitation and measurement (ARM) guide system [11], and the Bi-Manu-Track [12], the robot assists the movements of the affected limb. The MIT-MANUS, in particular, is widely used to retrain reaching movements and has been extensively tested in clinical trials [13], [14]. These robots can enhance performance and function in patients poststroke by providing intensive, cost-effective rehabilitation, e.g., [15] and [16]. Recent developments include using robots that allow retraining of multiple joints [17], a new wrist extension for the MIT-MANUS [18], adaptive algorithms that balance robotic assistance and patients' active movements [19], and electromyographically (EMG) triggered robots [20].

Although shown to be effective to some extent, these systems do not exactly parallel the role of rehabilitation therapists, who typically expend considerable effort to setup functional tasks that require a patient to actively engage in challenging reach-and-grasp practice [21]. Recovery of a variety of functional tasks constituting activities of daily life (ADL) has been shown to improve quality of life [22]. To be effective, the tasks should be meaningful [23]–[25] and should involve the manipulation of real and functional objects [26], [27].

Both scheduling and the difficulty of each task are also presumably important for effective rehabilitation. Research in motor learning has shown that learning is strongly influenced by the number of trials, as well as by the schedules in which the multiple tasks are practiced (e.g., random, blocked, or interval-expanded presentation of tasks) [28]–[30]. Furthermore, challenging tasks, i.e., tasks that are neither too difficult nor too easy, are most likely to elicit motor learning [31]–[34]. Challenging tasks also enhance motivation, which may in turn further enhance learning [35]. Because patients' performance will usually improve during rehabilitation, and because relearning evolves at different rates for each task and each subject, the task difficulty needs to be dynamically adjusted to maintain challenge. Accordingly, we previously showed that an adaptive algorithm,

Manuscript received September 18, 2008; revised February 15, 2009 and March 23, 2009. Current version published June 5, 2009. This paper was recommended for publication by Associate Editor E. Guglielmelli and Editor K. Lynch upon evaluation of the reviewers' comments. This work was supported in part by the Division of Biokinesiology and Physical Therapy, University of Southern California, and in part by the National Institutes of Health under Grant R03 HD050591-02. This paper was presented in part at the International Conference on Robotics and Automation, Pasadena, CA, May 2008.

Y. Choi is with the Department of Computer Science, Los Angeles, CA 90089 USA (e-mail: younggch@usc.edu).

J. Gordon and D. Kim are with the Division of Biokinesiology and Physical Therapy, University of Southern California, Los Angeles, CA 90089 USA (e-mail: jamesgor@usc.edu; ikarus1006@gmail.com).

N. Schweighofer is with the Division of Biokinesiology and Physical Therapy and the Departments of Neuroscience and Computer Science, University of Southern California, Los Angeles, CA 90089 USA (e-mail: schweigh@usc.edu).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TRO.2009.2019787

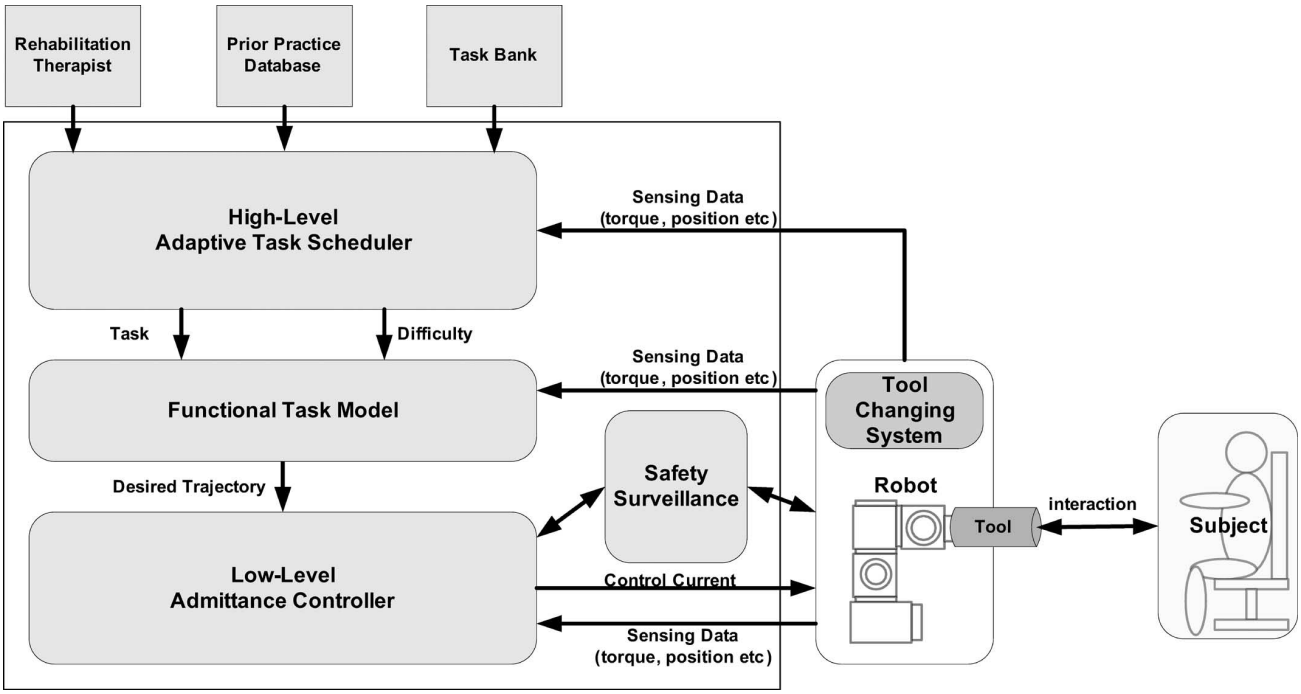


Fig. 1. Conceptual design of the ADAPT system. At each trial, the *high-level adaptive task* scheduler adaptively chooses a task and its difficulty based on subject's performance, prior practice records, and rehabilitation therapist's input. Rehabilitation therapist's input is not implemented in the current version of ADAPT. The tool-changing system automatically selects the tool corresponding to the selected task. The *low-level admittance controller* computes the control current needed to simulate the desired task dynamics from the *functional task model* during robot–patient interaction. The control architecture of the ADAPT controller is implemented on a Linux-operated computer.

which determines both the number of trials and the difficulty for each task based on learner's performance, outperforms fixed random scheduling [28]. Because rehabilitation of arm and hand function after stroke can be seen as relearning motor skills, we anticipate that robotic-based rehabilitation will benefit from the use of such automated adaptive practice schedule and difficulty.

Previous work suggests that effective rehabilitation of upper extremity functions after stroke should be delivered according to the following principles: the rehabilitation should be intensive, task-based, should require active participation, and should be adaptive in the number of trials and difficulty of each task. Several rehabilitation systems have been developed along some, but not all, of these principles. AutoCite [36], which is a semiautomated (nonrobotic) system that allows patients to engage in the practice of functional tasks, has been shown to be as effective as standard constraint-induced (CI) therapy [37]. In AutoCite, however, the number of tasks cannot be increased easily, task selection is manual, and task difficulty is not adjusted automatically. A smart training system with a robotic arm [38], [39] was developed to retrain reaching movements with adaptive difficulty. This system, however, does not contain automatic task selection, and is only designed for reaching. The robotic system ADLER [40] presents multiple functional tasks for activities of daily living with several artifacts. Although it provides realistic functional tasks, ADLER cannot implement adaptive task schedule, because it does not have an automatic artifact changing system. Thus, to our knowledge, no system currently engages the patient intensively, actively, and adaptively in a variety of realistic functional tasks. Notably, no previous systems have been designed to manipulate task schedule to optimize relearning.

We, thus, propose a new system, known as adaptive and automatic presentation of tasks (ADAPTs), that automatically presents functional tasks that require reaching and manipulation, that can accommodate an expanding number of such tasks, and that allows the implementation of performance-based adaptive task scheduling and adaptive modification of task difficulty. ADAPT simulates the dynamics of daily living functional tasks, such as opening a doorknob, opening a jar, turning a key, etc. Like AutoCite, but unlike most other robotic systems, ADAPT does not move the patients. Instead, it presents tasks adaptively, such that each patient can perform doable but constantly challenging tasks. Therefore, the system is designed for patients with some volitional motor capability of the arm and hand, as these patients benefit the most from intensive rehabilitation [3].

The primary aims of this paper are to present the design and the control architecture of ADAPT and to provide preliminary data with a healthy subject to show the successful operation of our approach. The utility of ADAPT for rehabilitation with stroke patients will be tested in future clinical trials. Partial work with preliminary design of the control architecture for ADAPT and task modeling was presented in conference proceedings [41].

II. Design of ADAPT

A. Intelligent Control Architecture of ADAPT

Fig. 1 presents the overall conceptual design of ADAPT system with the control architecture. The *high-level adaptive task scheduler* aims to maximize the relearning of multiple functional tasks and balancing learning among tasks in a limited

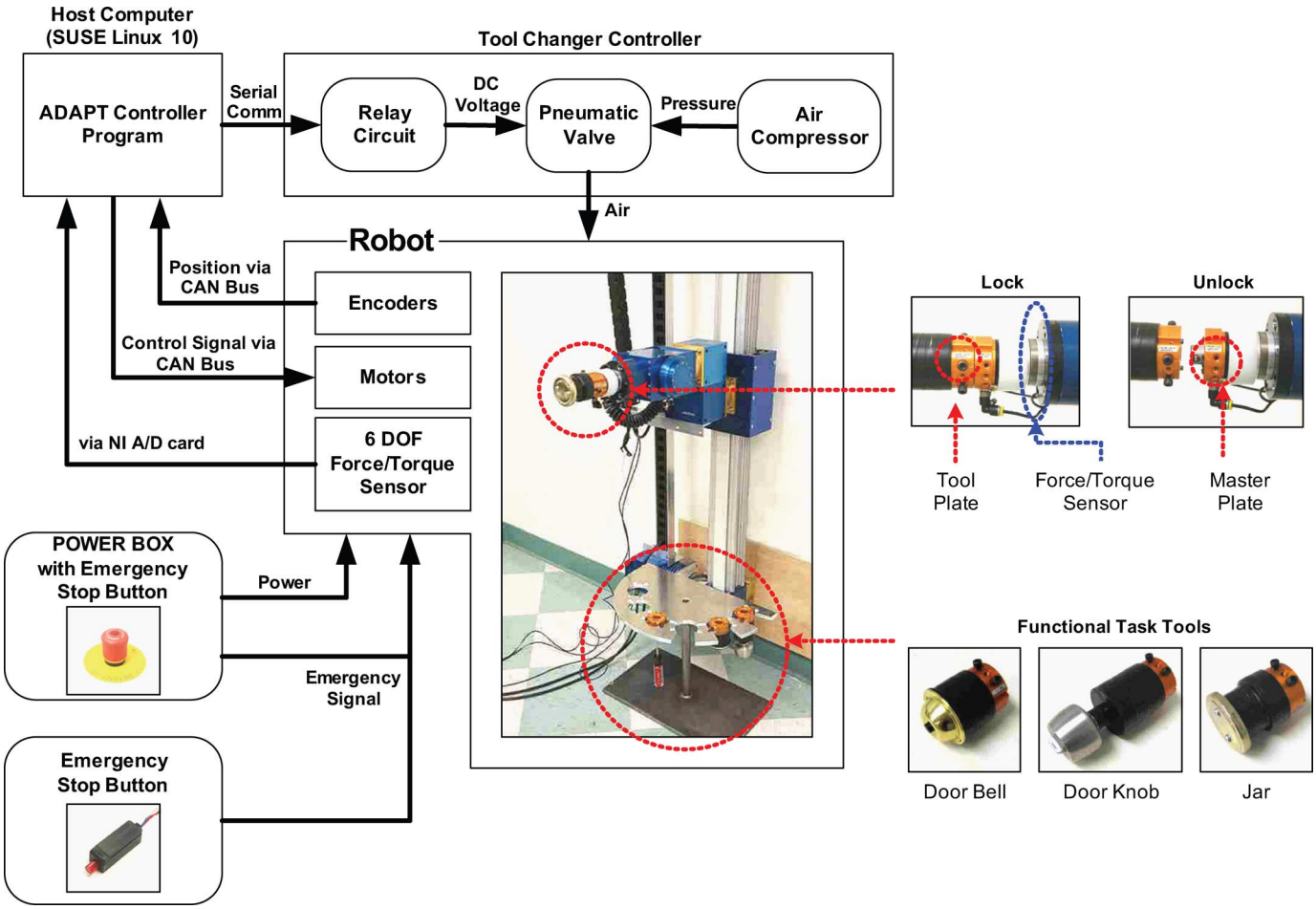


Fig. 2. Robot and devices for ADAPT. ADAPT controller program in Linux PC controls the robot via CAN and the tool changer controller via serial communication (RS-232 C). Tool changer controller controls the master plate pneumatically to lock and unlock a tool equipped with a tool plate. Two kinds of emergency buttons are present: One is on top of the power box, and the other is in hand of an unaffected arm. Two sensors: The torque sensor is attached between the master plate and robot's end-effector, and the encoder is embedded in each module of the robot. (Thick line: signal path; thin solid line: functional blocks; dotted line: highlighted component.)

training time; at each trial, it selects both the task to practice within a task bank and the task difficulty based on the patient's previous performance, input from the therapist, and constraints given by the adaptive schedules. The high-level adaptive task scheduler also sends a command to the *tool-changing system* that picks up the tool corresponding to the selected task. The *functional task model* generates a desired trajectory to simulate the dynamics of the task with the difficulty specified by the *high-level adaptive task scheduler*. The *low-level admittance controller* computes the control current corresponding to the desired trajectory. The computed control current is then applied to a general-purpose robot to simulate the task, and the patient feels the simulated dynamics by reaching and grasping the selected tool. The *safety surveillance* module continuously monitors the status of the robot and the controllers in the ADAPT system and provides appropriate actions to minimize any risk to patients.

B. Robot and Devices

The robot used in ADAPT is a modular and reconfigurable robot from AMTEC Robotics [42] (see Fig. 2). The current configuration consists of a 3 degree-of-freedom (DOF) wrist mounted on a 1-DOF linear actuator. This configuration allows the system to present the end-effector at different linear vertical locations and to rotate the end-effector in almost any orientation. Although it provides less haptic fidelity than backdrivable haptic devices, our general-purpose robot can generate higher torque, which is needed for a number of functional tasks (e.g., opening a jar). The low backdrivability also makes it easier for the robot to automatically pick up new tools with our tool-changing system. Furthermore, due to both the simplified computations of kinematics and dynamics, and the small workspace compared with traditional multiple degrees of freedom robotic arms, our robot provides safe task presentation to patients. Finally, note that the current configuration can be relatively easily extended with additional AMTEC linear and angular modules to increase the number of possible tasks and ranges of motion.

A 6-DOF ATI force/torque (F/T) sensor (MINI SI-580-20) is attached to the end-effector to measure interaction forces between the subject and the robot. The tool changer connects a functional task tool, such as a doorknob, to the F/T sensor. Encoders in the motor module provide position data of each joint. A Pentium-4 3.4-Ghz PC with a Linux operating system (SUSE

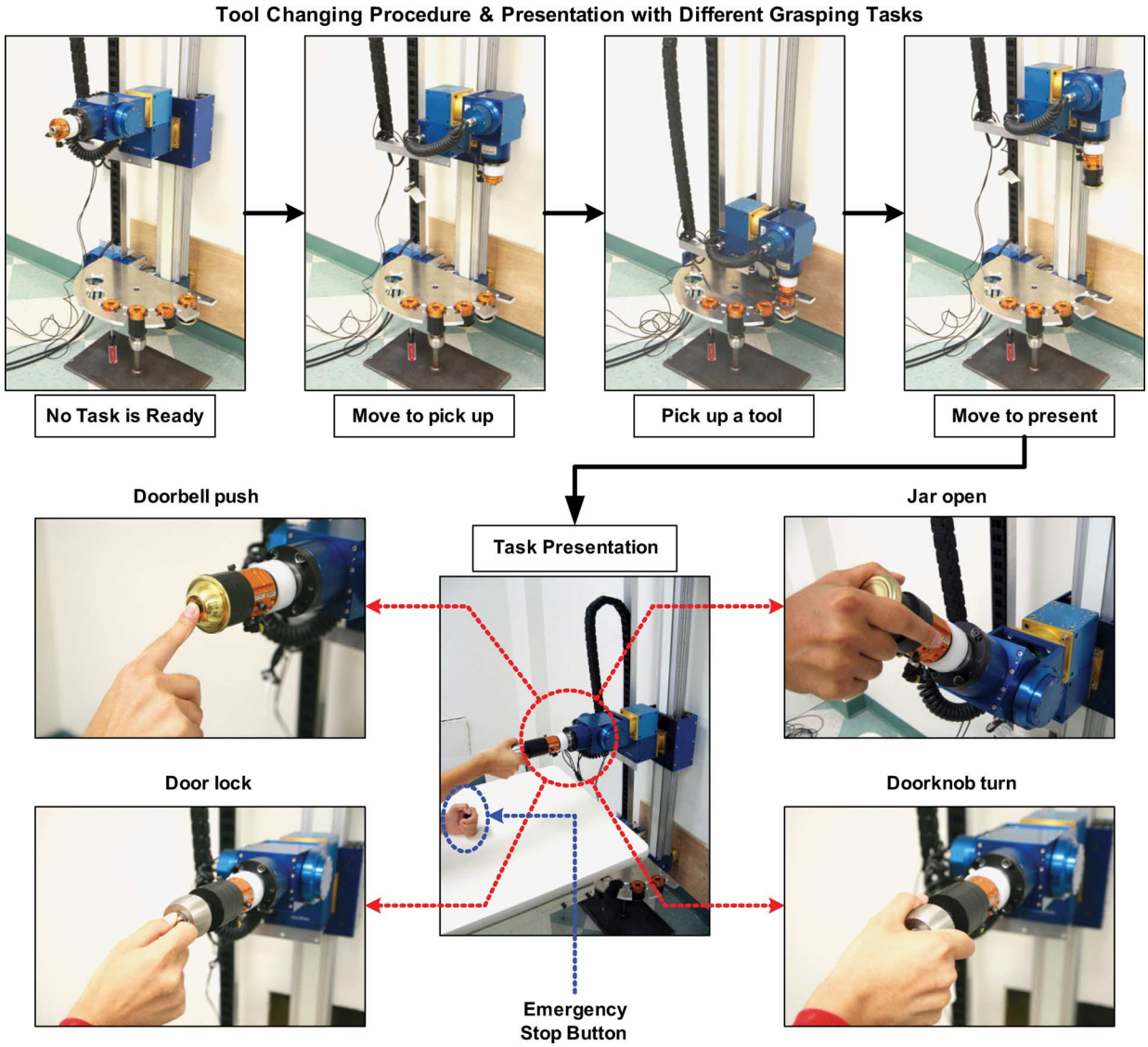


Fig. 3. Tool-changing process and different functional task tools in use. Four different grasping tasks with three different functional task tools (doorbell, jar, and doorknob) are developed. A subject holds the emergency stop button in the hand of an unaffected arm.

10) receives the position data via a controller area network (CAN) bus, receives the interaction force data via a National Instrument A/D converter, and sends control commands via the CAN bus.

Functional task tools such as a doorknob, a screwdriver, a jar, a faucet, keys, etc. are arranged in a tool rack. After the *high-level adaptive task scheduler* chooses a task, the 4-DOF robot positions each joint so that it can pick up the appropriate tool in the tool rack. As shown in Fig. 2, the robot's end-effector is equipped with a master plate, and each tool is equipped with an interface plate from ATI Corporation. A pneumatic system ensures the locking and unlocking of the tools by a 4/2-way pneumatic valve (V5 A-3341-BX1, MEAD Corporation), which is computer-controlled via RS-232 C, which is a serial communication. The PC sends a lock command to a 24-V_{dc} relay circuit, which sets the direction of the air flow in the pneumatic valve such that the master plate of the tool changer docks with the tool plate. After the two plates are docked, the robot repositions to present the new tool to the subject. In the current version, four tools are arranged in the rack, and up to six tools can be included. The tool changing process of these tools in use is demonstrated in Fig. 3.

The subject is seated on a chair facing the robot with an in-between table, lays his/her hands on the table, and practices the tasks simulated by the robot with the more affected upper limb. As shown in Fig. 3, with different task tools, patients can practice a variety of different reaching and grasping tasks.

C. *Safety*

Safety was a crucial issue in the design of ADAPT. From the initial robot design process, we made special efforts to maximize operational safety. Our choice of design makes our robot safer than a traditional multidegrees of freedom robotic arm because

of the small overall workspace. The linear degree of freedom is only used for tool positioning and not for task dynamics simulation. Furthermore, the patient is not strapped to the robot.

For simplicity and safety reasons, we chose functional tasks that require movements around a single degree of freedom during robot–subject interactions. Because many functional tasks in daily living (such as turning a key or door handle, steering, opening a jar, turning a water faucet, wrist supination/pronation, etc.) need only a single rotary degree of freedom, this configuration does not overly restrict the number of tasks. After a task is setup for presentation to the subject, the magnetic brakes that are built into the robotic articulations are engaged on the other 3 DOFs during subject–robot interactions. This single degree of freedom method simplifies kinematics and dynamics computation; the robot, thus, never falls into the wrist-singular posture, which can occur in PUMA-like manipulators [43].

Several surveillance routines are implemented to limit the maximal torque output and cap the maximum velocity of the linear and rotational motors. Watchdog routines, which continuously check for failure of the position and force sensors, computer crashes, and electrical failures, can automatically freeze the robot by engaging the magnetic breaks in all degrees of freedom at any time.

Furthermore, two emergency stop buttons can stop all robot operation and turn-ON magnetic brakes to disable any movement of all 4 DOFs of the robot. The main emergency red stop button of the power box is accessible to the therapist. The patient holds the second emergency stop button at all times with his/her less-affected hand (see Figs. 2 and 3).

Finally, to limit possible patient–robot collisions, subjects are seated with their trunk fastened to the back of the chair by a seat belt. This seat belt has the additional advantage of limiting compensatory movements with the trunk.

III. Control of ADAPT

As described in the conceptual design of ADAPT in Section II, the control architecture consists of three main components, *high-level adaptive task scheduler*, *functional task model*, and *low-level admittance controller*. In this section, we describe the development of the three components, the methodologies with supporting theoretical background, and the relationship between these components.

A. High-level Adaptive Task Scheduler

The high-level adaptive task scheduler has two adaptive components: The first determines the task schedules, and the second sets the difficulty of the task selected at each trial. We review these two components next.

1) *Adaptive task schedules:* In learning multiple visuomotor tasks, we previously showed that an adaptive algorithm, which determines the number of trials for each task based on previous learner's performance, outperformed fixed random scheduling in learning multiple visuomotor tasks [28]. Although a number of other adaptive task scheduling algorithms can be used in ADAPT, we use this algorithm in our current high-level adaptive task scheduler. Briefly, in this algorithm, the probability that each task is selected in the current session is proportional to the (normalized) product of the performance measured in the previous session and the performance measured in the current training session. See [28] and Section IV for more details.
2) *Adaptive task difficulty:* We recently showed in a learning experiment involving multiple visuomotor tasks that adaptive difficulty algorithms outperform fixed difficulty [28]. Although a number of other adaptive task difficulty algorithms can be used in ADAPT, we use this algorithm in the high-level adaptive task scheduler. Briefly, in this algorithm, the difficulty is updated in order to constantly maintain performance around a "challenging point." See [28] and Section IV for more details.

B. Functional Task Model

Once the task has been selected, the tool chosen and mounted on the robot by the tool changer, and the task difficulty determined, the *functional task model* generates the desired trajectories for the selected task and difficulty.

The simplest way for stroke patients to practice functional tasks would be to practice with real functional task tools such as a door knob, a jar, etc. The problem with using such passive tools, however, is that the difficulty of the tools cannot be adapted to the individual and (presumably) improving patient's performance. Here, we present a novel modeling method for functional tasks such that the feel or difficulty of the task tools is adjustable. The captured task model is then embedded as a *functional task model* in the control architecture to generate the desired trajectory for the selected task with specified difficulty. State-dependent nonlinear dynamics (e.g., position-dependent stiffness or damping) of passive tools can be modeled by the combination of multiple local linear models [44]. Here, we formulate the modeling of the functional tasks as locally weighted regression problem with receptive field weighted regression (RFWR) [45]. RFWR has two main advantages. First, compared to the method used in [44], it provides a computationally efficient way of modeling position-dependent dynamics, because it adaptively creates and prunes the local models. Second, as only few local linear models are created, it allows good generalization.

Friction is present in all passive tools. Although a variety of friction models have been proposed [46], the modified Karnopp model has been shown to be effective for simulating the dynamics of passive tools with a haptic interface [47]. The inherently nonlinear Karnopp model can be formulated as a linear equation by expanding the number of state variables [47]. In the remainder of this section, we describe 1) how we implemented RFWR for our purpose; 2) how we used the Karnopp model as local linear basis functions for RFWR; and 3) how we generated the desired trajectories for the desired tasks with desired difficulty.

1) Receptive Field Weighted Regression (RFWR): In RFWR, the regression function values are approximated by a combination of N individually weighted locally linear models and normalized by the sum of all weights. Therefore, given a

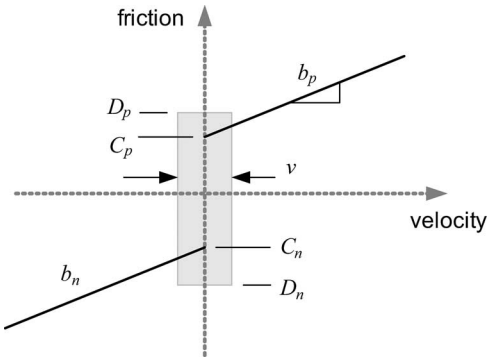


Fig. 4. Modified Karnopp model. C_p and C_n are the positive and negative values of the dynamic friction, b_p and b_n are the positive and negative values of the viscous friction, velocity is the relative velocity between two surfaces, and D_p and D_n are the positive and negative values of the static friction. The width of shaded area defined by v is where the velocity is considered to be zero.

training data point x, the predicted regression value y is

$$y = \frac{\sum_{k=1}^{K} w_k y_k}{\sum_{k=1}^{K} w_k}$$

$$y_k = x^T b + b_{0,k} = \vec{x}^T \beta_k, \qquad \vec{x} = (x^T, 1)^T \tag{1}$$

where y_k is the individual prediction from each local linear model (each has a receptive field), β_k is the local linear model parameter, and the weights w_k corresponds to the activation strength of each receptive field. The weights are determined from the size and shape of each receptive field, characterized by a Gaussian kernel function, as commonly used in RFWR

$$w_k = \exp\left(-\frac{1}{2}(x_{\mathrm{dep}} - c_k)^T D_k (x_{\mathrm{dep}} - c_k)\right) \tag{2}$$

where D_k is a positive definite distance matrix, x_{dep} is the state variable for the dependency of local models (e.g., position), and c_k is the center of kth linear model in the dimension of x_{dep}. The state variable x_{dep} is made of one or several elements of the input vector x in (1). For example, if x_{dep} is an angle, then the local model y_k for RFWR is created or pruned in terms of angle space. While learning with RFWR, the parameter β_k is updated for each local linear model, and the distance matrix D_k is updated to determine the shape and size of the receptive field. For each of our 1-DOF task, y is the torque output of the dynamics model for the current state x (e.g., angle, velocity, and acceleration). Each y_k in (1) is formulated by a modified Karnopp friction model function, which we now describe.

2) Karnopp Friction Model: Here, we show how we included the Karnopp model within the RFWR framework to model the dynamics of any 1-DOF rotary tool with friction.

Fig. 4 depicts the modified Karnopp friction model used in [47]. The corresponding linear equation for the dynamics of a 1-DOF rotary tool (e.g., doorknob) can be expressed by

$$\begin{aligned}\tau_{\mathrm{measured}} &= \tau_{\mathrm{inertia}} + \tau_{\mathrm{friction}} + \tau_{\mathrm{misc}} + \varepsilon \\ &= I_{\mathrm{tool}} \times \mathrm{acc} + C_p \,\mathrm{sgn}(\mathrm{vel}_p) + b_p \mathrm{vel}_p \\ &\quad + C_n \,\mathrm{sgn}(\mathrm{vel}_n) + b_n \mathrm{vel}_n + \tau_{\mathrm{misc}} + \varepsilon \end{aligned} \tag{3}$$

where I_{tool} is the inertia of the tool, τ_{measured} is the measured force, C_p and C_n are the positive and negative values of the dynamic friction, b_p and b_n are the positive and negative values of the viscous friction, acc is the angular acceleration, vel_p is the positive value of the angular velocity (if the velocity is not positive, it is zero), vel_n is the negative values of the angular velocity (if the velocity is not negative, it is zero), ε is a noise factor, and τ_{misc} is other physical properties, such as stiffness in case of a doorknob. In (3), by adding τ_{misc}, we generalized the modified Karnopp model to any passive 1-DOF rotary tools. The vector form corresponding to (3) to be used in (1) is

$$\begin{aligned}&\tau_{\mathrm{measured}} - \tau_{\mathrm{inertia}} \\ &= [\mathrm{sgn}(\mathrm{vel}_p) \quad \mathrm{vel}_p \quad \mathrm{sgn}(\mathrm{vel}_n) \quad \mathrm{vel}_n] \begin{bmatrix} C_p \\ b_p \\ C_n \\ b_n \end{bmatrix} + \tau_{\mathrm{misc}} + \varepsilon \\ &= x\beta' + \tau_{\mathrm{misc}} + \varepsilon \\ &= x\beta + \varepsilon \end{aligned} \tag{4}$$

where we assumed that τ_{misc} is a linear function of *x,* the state vector, and β is a parameter vector that can be formulated as in (1). If we assume that τ_{misc} is only due to stiffness (e.g., a doorknob), then the state vector and parameter vector will be

$$\begin{aligned} x &= \begin{bmatrix} \mathrm{sgn}(\mathrm{vel}_p) & \mathrm{vel}_p & \mathrm{sgn}(\mathrm{vel}_n) & \mathrm{vel}_n & \mathrm{ang}_p & \mathrm{ang}_n \end{bmatrix} \\ \beta &= [C_p \quad b_p \quad C_n \quad b_n \quad K_p \quad K_n]^T \end{aligned} \tag{5}$$

where K_p and K_n are the stiffness values for positive and negative velocities, and ang_p and ang_n are angles. To adjust the difficulty of a functional task, we can either multiply β by a difficulty value Diff or, if we want to change only a specific property for difficulty (e.g., stiffness), we can assume Diff to be a vector and multiply β element-wise by Diff. Inertia is excluded from the parameter vector because we assume that inertia is not state-dependent. Other parameters (D_p, D_n, and v) in Fig. 4 are obtained empirically, as shown in [47] and [48].

We obtained motion and torque data to train the aforementioned model with the F/T sensor introduced in Section II and a magnetic motion sensor (MINI bird, Ascension Technology Corporation) attached to the passive tool to be modeled. The tool is attached to the torque sensor, which is positioned not to move. The subject then manipulates the passive tool, and the torque and angle data are simultaneously recorded. The details of obtaining the training data are described in [49].

3) Numerical Integration for Desired Trajectories: After the model of (1), which uses with the local linear model of (4), is learned from the training data, the dynamics model for the functional task tsk, and difficulty Diff can be assumed to be

$$\begin{aligned} y(\mathrm{tsk}, \mathrm{Diff}) &= \frac{\sum_{k=1}^{K} w_k y_k}{\sum_{k=1}^{K} w_k} \\ y_k &= x^T b_k + b_{0,k} = \vec{x}^T \beta_k \mathrm{Diff} \\ \vec{x} &= (x^T, 1)^T \end{aligned} \tag{6}$$

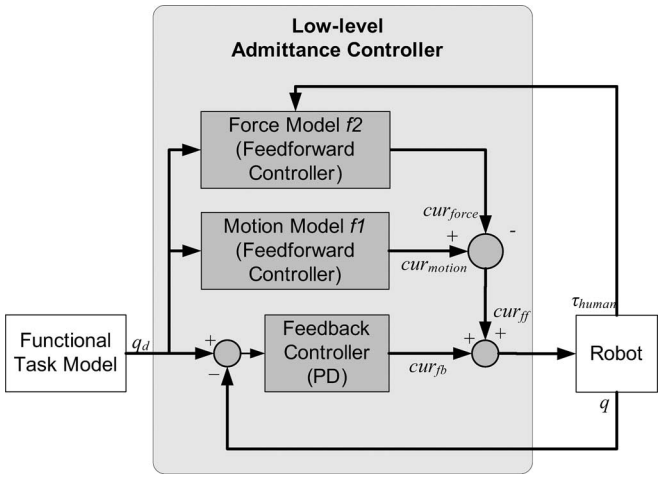


Fig. 5. *Low-level admittance controller* for ADAPT. The controller generates the desired motion in response to the external force by the subject. The measured external force is input to the *functional task model* that defines the dynamics of the functional tasks. Based on the dynamics of the task, the *functional task model* generates the ideal desired motion. Model *f1* computes the control current for the desired motion, and *f2* computes the control current to compensate against the external force. The desired state vector q_d and the current state vector q correspond to $(\mathrm{ang}_d, \mathrm{vel}_d, \mathrm{acc}_d)$ and (ang, acc, vel) in the text.

where we can assume that β of (4) or (5) (the modified Karnopp model) is equal to each β_k. Then, by inserting (6) into (4), the desired trajectory can be derived

$$\begin{aligned}\tau_{\mathrm{measured}} - \tau_{\mathrm{inertia}} &= \tau_{\mathrm{measured}} - I_{\mathrm{tool}}\mathrm{acc} = y(\mathrm{tsk}, \mathrm{Diff}) \\ \mathrm{acc}_d = \mathrm{acc} &= \frac{\tau_{\mathrm{measured}} - y(\mathrm{tsk}, \mathrm{Diff})}{I_{\mathrm{tool}}} \\ \mathrm{vel}_d &= \mathrm{vel} + \frac{\tau_{\mathrm{measured}} - y(\mathrm{tsk}, \mathrm{Diff})}{I_{\mathrm{tool}}}\Delta t \\ \mathrm{ang}_d &= \mathrm{vel}_d \Delta t \end{aligned} \tag{7}$$

where (ang, acc, vel) is a current state vector, $(\mathrm{ang}_d, \mathrm{vel}_d, \mathrm{acc}_d)$ is a desired state vector, and Δt is a sampling time of the robot. The current state vector is measured by the encoder of the robot at each time step.

C. Low-level Admittance Controller

The *low-level controller* (see Fig. 5) receives the desired trajectory from the *functional task model*. Because our general-purpose robot has low backdrivability, we used an admittance control strategy to compute the control signal for the desired motion response to external force. The control signal is the motor current and is computed from the outputs of two feedforward control modules and a typical potential difference (PD) feedback controller. The first module is a controller for motion without interaction force (motion model *f1*), and the second is a controller for interaction force without motion (force model *f2*). We trained both modules offline with RFWR. Such modular design of the *low-level admittance controller* with RFWR has three main advantages: First, learning the control currents directly allows accurate control even when, as in our system, the robot's dynamics are unknown, and when the control current is (presumably) not proportional to the motor torque. Second, compared with a combined controller, our modular controller simplifies the training of the controllers, because separation of the force and motion models allows us to train each model with fewer training data points. Third, the force model can be used to simulate infinite stiffness or damping, which is needed to implement a virtual wall or tasks with high friction. The motion model is trained with a typical sinusoidal excitation with no human–robot interaction. Training data for the force model are recorded from the F/T sensor while a subject exerts force on the tool with the robot immobilized with position control.

We assume the dynamics of our robot to be

$$\tau_{\mathrm{motor}} = I_{\mathrm{robot}}\mathrm{acc}_d + n(\mathrm{ang}_d, \mathrm{vel}_d) - \tau_{\mathrm{measured}} \tag{8}$$

where $(\mathrm{ang}_d, \mathrm{vel}_d, \mathrm{acc}_d)$ are the desired angles, angular velocity, and angular acceleration, respectively, τ_{motor} is the control torque, τ_{measured} is the external torque measured by F/T sensor, I_{robot} is the inertia of the haptic interface, and $n(\mathrm{ang}_d, \mathrm{vel}_d)$ models possible nonlinear effects. As τ_{measured} in (8) is the interaction torque exerted by the subject, the other terms in (8) contribute to τ_{motor} only via motion. Thus, from the dynamics equation for motion only (with no human–robot interaction), we assume that the current output function for motion is only given by the motion model *f1* with desired trajectory as an input

$$\begin{aligned}\tau_{\mathrm{motor}} &= I_{\mathrm{robot}}\mathrm{acc}_d + n(\mathrm{ang}_d, \mathrm{vel}_d) \\ &\rightarrow \mathrm{cur}_{\mathrm{motion}} = f(\tau_{\mathrm{motor}}) = f1(\mathrm{ang}_d, \mathrm{vel}_d, \mathrm{acc}_d).\end{aligned} \tag{9}$$

Similarly, we assume that the current-output function for interaction is only given by *f2* with τ_{measured} as an input

$$\rightarrow \mathrm{cur}_{\mathrm{force}} = f2(\tau_{\mathrm{measured}}). \tag{10}$$

From (8)–(10), the real control current to the motor is given by

$$\mathrm{cur}_{\mathrm{motor}} = f1(\mathrm{ang}_d, \mathrm{vel}_d, \mathrm{acc}_d) - f2(\tau_{\mathrm{measured}}). \tag{11}$$

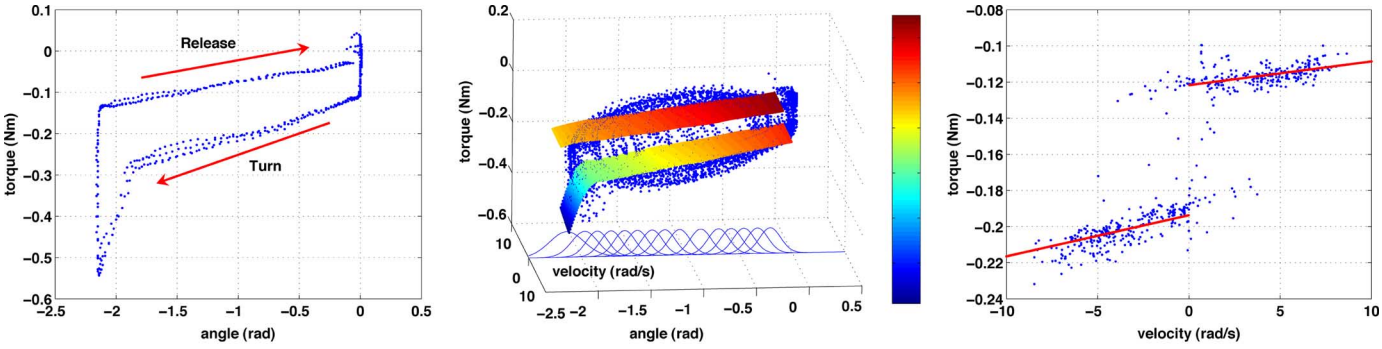
Finally, to drive the errors between the desired motion and the actual motion to zero, feedback terms are added to (11) as

$$\begin{aligned}\mathrm{cur}_{\mathrm{motor}} = {} & f1(\mathrm{ang}_d, \mathrm{vel}_d, \mathrm{acc}_d) - f2(\tau_{\mathrm{measured}}) \\ & + K_d(\mathrm{vel}_d - \mathrm{vel}) + K_p(\mathrm{ang}_d - \mathrm{ang}).\end{aligned} \tag{12}$$

The desired trajectory $(\mathrm{ang}_d, \mathrm{vel}_d, \mathrm{acc}_d)$ is numerically integrated from (7), as described in the previous section.

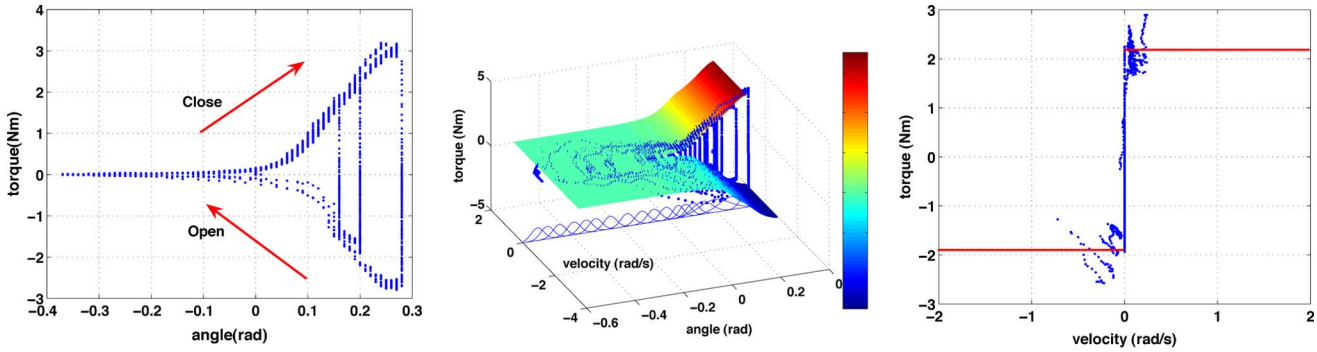
IV. Results

Here, we first present results for the *functional task model* for two tasks, turning and jar closing–opening, and we then show how these two tasks were simulated by the *low-level admittance controller* for varying level of difficulty. We then describe the current implementation of *high-level adaptive task schedule for* three functional tasks: doorknob turning, jar opening–closing, and doorbell pushing. We, finally, validated the overall ADAPT system by adaptively presenting these three tasks for a total of 180 trials to a healthy male subject. Some of these results are also discussed in [41].



(a) (b) (c)

Fig. 6. Doorknob dynamics model. (a) Torque versus angle trajectory of a real doorknob turned and released three times. (b) Training data (dots) and the doorknob dynamics model (surface) after training by RFWR with the modified Karnopp friction model. Solid lines on the bottom are 15 weight functions given by (2). (c) Local torque versus velocity relation by the trained model (solid lines) and the training data (dots) around angle $= -1.5$ rad ($-1.55 <$ angle < -1.45).



(a) (b) (c)

Fig. 7. Jar dynamics model. (a) Torque versus angle trajectory of a real jar closed and opened three times. (b) Training data (dots) and the jar dynamics model (surface) after training by RFWR with the modified Karnopp friction model. Solid lines on the bottoms are 18 weight functions corresponding to (2). (c) Local torque versus velocity relation by the trained model (solid lines) and the training data (dots) around angle $= 0.2$ rad ($0.19 <$ angle < 0.21).

Training data to model doorknob turning and jar closing–opening were obtained as explained in Section III-C. Fig. 6(a) demonstrates the sample trajectory of turning and releasing a real doorknob in a torque versus angle plane. One important aspect of doorknob dynamics is that turning (positive velocity) movements require more torque than releasing (negative velocity) movements because of friction: this is clearly shown near zero velocity in Fig. 6(c). This confirms that the Karnopp model is well suited for modeling doorknob dynamics. The important aspect of doorknob dynamics is that both the stiffness and the friction magnitude are changing with position (angle). For example, the slope near -2 rad is larger than the slope near -0.5 rad, and the torque difference between turning and releasing at angle -2 rad is larger than the difference at -0.5 rad [see Fig. 6(a)]. Position (angle) dependent local Karnopp models are thus necessary. We used local Karnopp models with the state input vector and the parameters of (5) to the RFWR regression of (3). Fig. 6(b) shows the collected training data, the doorknob dynamics model built on the training data, and Gaussian weight functions. The final learned dynamics model contains 15 local Karnopp models, which are combined after learning with RFWR to generate the model. The combined model corresponds to y in (1) and (6), and the weight function is defined by (2). In Fig. 6(c), the dots represent the collected training data at angles between -1.55 and -1.45 rad, and the solid lines are the torque values predicted from the model with inputs of the velocity range between -10 and 10 rad/s, and at angle -1.5 rad. As can be seen in the figure, the model has the same shape as the modified Karnopp model (c.f., Fig. 4), and fits the data very well.

A sample trajectory and the training data of jar closing–opening are plotted in Fig. 7(a) and (b), respectively. The jar dynamics is almost a pure friction problem, with no stiffness, with the friction magnitude changing depending on angle (higher friction with more closing). As shown in Fig. 7(a) and (b), the jar does not move until the exerted torque exceeds a positive threshold for jar closing or is less than a negative threshold for jar opening. Because it was hard for the subject to generate high velocity for jar closing, too few training data were collected to obtain reasonable viscosity values [b_p and b_n in (5)]. We, therefore, set these viscosity values to zero. Fig. 7(c) shows that this assumption is reasonable, as the velocity range of the training data is narrow. The input state vector of (5) without stiffness and with zero viscosity is then given by

$$x = \lfloor \mathrm{sgn}(\mathrm{vel}_p) \quad \mathrm{vel}_p \quad \mathrm{sgn}(\mathrm{vel}_n) \quad \mathrm{vel}_n \rfloor$$
$$\beta = [C_p \quad 0 \quad C_n \quad 0]^T . \tag{13}$$

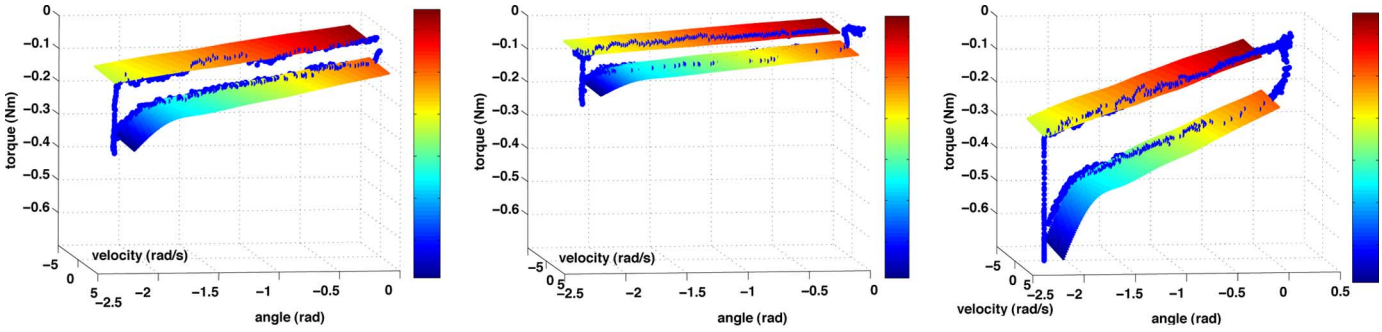


Fig. 8. Dynamics models of the doorknob and trajectories simulated by ADAPT for three different difficulty levels. (a) Doorknob dynamics model (surface) after training by RFWR with the modified Karnopp friction model and the trajectory (dots) simulated by ADAPT during interaction with a subject. The difficulty of this model is used as a reference. (b) Difficulty was decreased by half and the trajectories simulated by ADAPT during interaction with a subject. (c) Difficulty was increased by twice and the trajectories simulated by ADAPT during interaction with a subject.

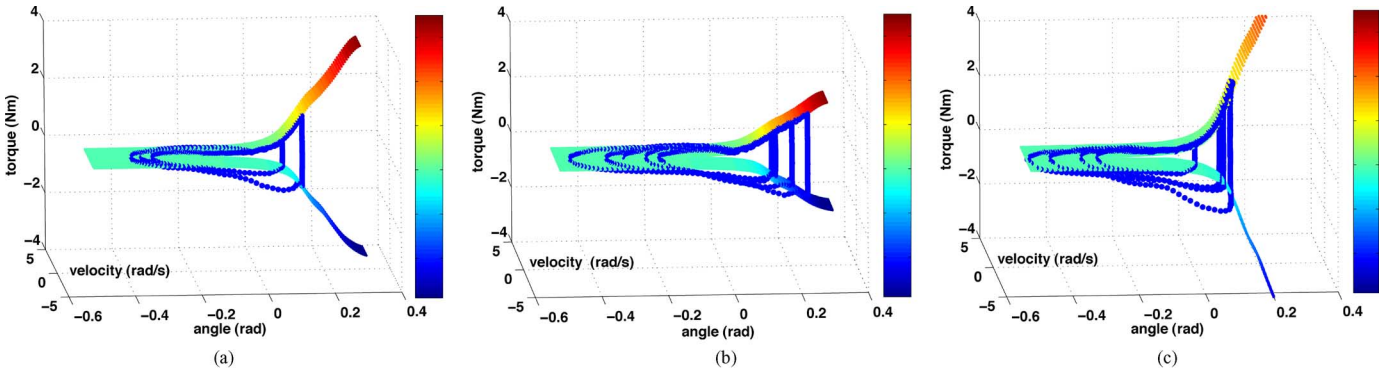


Fig. 9. Dynamics models of the jar and trajectories simulated by ADAPT for three different difficulty levels. (a) Jar dynamics model (surface) after training by RFWR with the modified Karnopp friction model and the trajectory (dots) simulated by ADAPT during interaction with a subject. The difficulty of this model is used as a reference. (b) Difficulty was decreased by half and the trajectories simulated by ADAPT during interaction with a subject. (c) Difficulty was increased by twice and the trajectories simulated by ADAPT during interaction with a subject.

As for the doorknob, we applied the local Karnopp models with the state input vector and the parameters of (13) to the RFWR regression of (3). Fig. 7(b) shows the collected training data, the jar dynamics model built with the training data with our proposed method, and the 18 weight functions given by RFWR after training. The local training data and the model at angle 0.2 rad in the Fig. 7(c) illustrate the exact shape of the modified Karnopp model with zero viscosity (c.f., Fig. 4). The good model fit validates the use of the local Karnopp friction model for the jar as well.

Therefore, the results in Figs. 6(b) and (c) and 7(b) and (c) clearly show that our novel modeling methodology can be used to closely predict the dynamics of passive tools with different physical properties.

Next, we tested 1) the performance of the *low-level admittance controller* with these two tasks and 2) the feasibility of the proposed modeling method for adjustment of task difficulty. Fig. 8 shows results of the robot simulation of the doorknob task for three levels of difficulty. In Fig. 8(a), the model of Fig. 6(b) was used. Although some errors are present because of sensor noise and inherent limitations of our robot due to its low backdrivability, the trajectory closely follows the learned model (surface). In Fig. 8(b) and (c), the difficulty was adjusted by multiplying β_k in (6) by a constant, i.e., Diff. The surfaces in Fig. 8(b) and (c) show the desired model after scaling all terms of the model with the difficulty Diff $= 0.5$ and Diff $= 2$, respectively. High-stiffness simulation, as in Fig. 8(c), showed good trajectory. As shown in the releasing trajectory of Fig. 8(b), very low stiffness seems to generate more errors, which we believe are due to the inherent limitations of our robot. However, the trajectory itself does not deviate much from the model, and the subject reported good "doorknob feel" in all difficulty conditions.

Fig. 9 shows the robot performance for jar closing–opening for the three difficulty levels. Although the overall shape of the trajectory follows the model, the jar-opening trajectory has a relatively higher starting velocity than the trajectory of the training data in Fig. 7(b), which may be due to our assumption of zero viscosity parameters in the Karnopp model. Nonetheless, the subject reported that the feeling of opening a jar with ADAPT was very similar to opening a jar for real. As shown in Fig. 9(b) and (c), the simulations with different difficulties successfully followed the desired model; in both cases, the subject also reported good feeling during jar closing–opening. Therefore, the *low-level admittance controller* successfully displayed the two functional tasks with adaptive difficulties in ADAPT.

Finally, we tested the scheduling properties and overall functionality of ADAPT with the adaptive scheduling of three tasks: doorknob turning, jar closing–opening, and doorbell pushing.

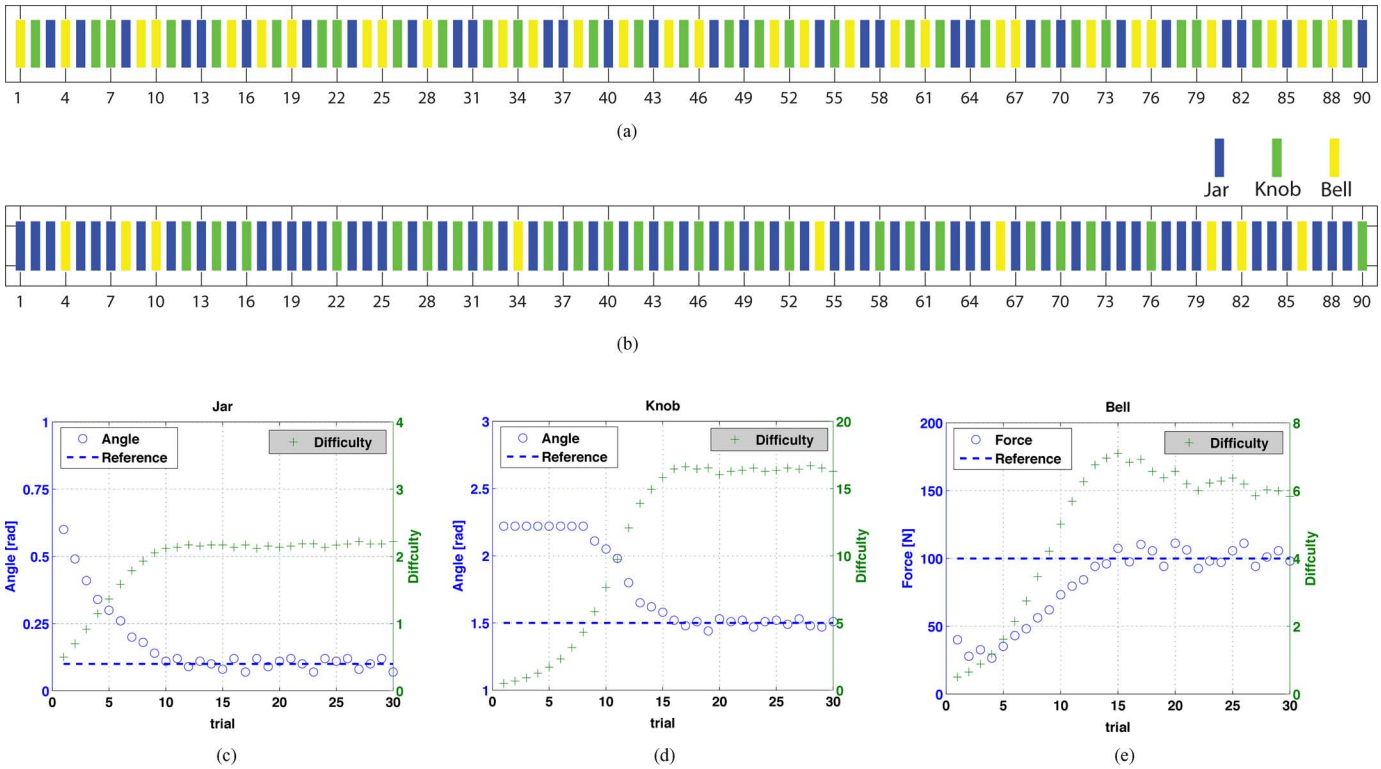


Fig. 10. Function of the high-level adaptive task scheduler. (a) Random task schedule of the first practice session with three tasks (jar closing–opening, doorknob turning, and doorbell pushing) with 30 trials per each task. (b) Task schedule of the second practice session with adaptive number of trials for each task. The more difficult task (jar closing–opening) was allocated more trials than the others tasks. (c) Difficulty of jar closing–opening and (d) doorknob turning was adapted such that the performance (angle) remained near the challenging point (reference). (e) Adaptive task difficulty of doorbell pushing in the first training session. In this case, the performance (force) increased as the difficulty increased.

The doorbell-pushing task was implemented by playing a bell sound when the pushing force reached a specific threshold in limited time. The experiment contained two practice sessions, each consisting of 90 trials and separated by 1 h (in future therapeutic applications of ADAPT, sessions may be separated by days or weeks). In the first session, we only adjusted the task difficulty, keeping the number of trials equal for all tasks. In the second session, we adjusted both difficulty and the number of trials based on performance. Specifically, in the first practice session, the three tasks were randomly scheduled, as shown in Fig. 10(a): Each task was presented 30 times, and the difficulty for each task was adaptively updated based on performance [see Fig. 10(c)–(e)]. At each trial, the instructions (ready, go, back, feedback) were shown sequentially on a computer screen along with corresponding audio signals. Feedback was displayed as "failure" or "success." For the doorknob-turning task, the trial is a success if the subject turns the doorknob more than a target angle. For the jar closing–opening task, the trial is a success if the subject closes the jar lid more than a target angle and opens it again. For the doorbell-pushing task, the trial is a success if the pushing force is higher than a threshold value.

Here, we explain how to compute the two adaptive components of *high-level adaptive task scheduler*: the number of trials for the task schedule and the difficulty of the task at each trial. The algorithm used to adaptively determine the number of trials is based on a performance test following training on the previous session and on a performance test preceding the current session [28]. We, thus, gave one posttest session of 30 trials (ten trials per task) immediately after the first training session and, 1 h later, another 30-trial test. The number of trials for each task in the second practice session is determined with

$$\mathrm{NumOfTrial(tsk)} \propto \frac{\mathrm{PE}_{s,\mathrm{pre}}(\mathrm{tsk})\mathrm{PE}_{s-1,\mathrm{pos}}(\mathrm{tsk})}{\sum_i \mathrm{PE}_{s,\mathrm{pre}}(\mathrm{tsk}_i)\mathrm{PE}_{s-1,\mathrm{pos}}(\mathrm{tsk}_i)} \quad (14)$$

where $\mathrm{PE}_{s,_{\mathrm{pre}}}(\mathrm{tsk})$ is the performance error for the task tsk in the pretest session of the second practice session, and $\mathrm{PE}_{s-1,_{\mathrm{pos}}}(\mathrm{tsk})$ is the performance error for the task tsk in the posttest session of the first practice session. In this experiment, we assumed that the performance error is the number of failure trials at the specific difficulty in the test session. The maximum and minimum performance errors are 10 and 0, respectively. Thus, the number of trials for each task is proportional to the multiplication of the performance errors at the two test sessions. The number of trials in the second session were 56 for the more difficult jar closing–opening task, 25 for the doorknob-turning task, and nine for the easier doorbell-pushing task. Fig. 10(b) illustrates the randomly distributed schedule with adjusted number of trials for each task.

We now discuss how difficulty is automatically adjusted based on patient's performance in *high-level adaptive task scheduler*. The difficulty update function is similar to [28] and is given by

$$D_t(\mathrm{tsk}) = D_{t-1}(\mathrm{tsk})(1 + \alpha(P_t(\mathrm{tsk}) - P_{\mathrm{ref}}(\mathrm{tsk})) \quad (15)$$

where $D_t(\text{tsk})$ is the difficulty for the current trial t and the task tsk, α is learning rate, $P_t(\text{tsk})$ is the performance, and P_{ref} is the reference challenging performance. In jar closing–opening [see Fig. 10(c)] and doorknob turning [see Fig. 10(d)], the performance is measured by the range of motion (angle), i.e., how far the subject turned the knob or the jar in a trial. The performance in doorbell pushing [see Fig. 10(e)] is measured by the force exerted on the doorbell. Note that in future work, the reference challenging performance will be set after pilot testing. Here, we simply took values close to maximum possible performance that the subject could exert in the three tasks. Thus, the subject had to learn how to increase the force exerted in the three tasks. Here, the difficulty Diff $= 1$ means that the simulated dynamics by ADAPT reflects that of the real passive tool. For doorbell pushing, our subject exerted typical pushing forces of around 15 N, which we used as threshold force for Diff $= 1$. Note that the update rate α must be negative for doorbell pushing and positive for the other two tasks.

Therefore, the trials number functionalities and adaptive difficulty of the *high-level adaptive task scheduler* were successfully implemented, and the system did not experience any failure in the tool changing process for 180 task trials.

V. Concluding Remarks

A novel robotic task-practice system, i.e., ADAPT, was designed in accordance to training principles for effective rehabilitation of upper extremity functions after stroke. Our preliminary test with a healthy subject demonstrates the operation and validated the feasibility of the system. Due to our novel *functional task model* and *low-level admittance controller*, the robot precisely simulated the desired task dynamics of a doorknob and jar with the difficulties specified by the *high-level adaptive task scheduler*. The preliminary experiment further demonstrated the functionality, robustness, and safety of ADAPT.

At this stage, we have implemented only three unimanual tasks (doorknob turning, jar closing–opening, and doorbell pushing). Using the current configuration of a 3-DOF wrist mounted on a 1-DOF linear actuator, we will, in future versions, develop additional tasks that require active manipulation of concrete objects (e.g., turning a key or door handle, steering, turning a water faucet, lifting jugs, etc.) and will allow a larger number of possible types of grasps (e.g., overhand, precision, lateral pinch, and power) and individual finger motions. We will also develop bimanual tasks, as a growing number of studies have shown the potential of bilateral training on the recovery of the paretic limb poststroke [50], [51]. Finally, in later versions, we will add two extra linear modules to allow ADAPT to present an even larger task repertoire of tasks with greater range of (notably shoulder) motions.

In the current version, a seat belt is used not only for safety reasons but to also prevent compensatory trunk movements, which are known to interfere with recovery of reaching movements [52]. However, even with the belt, patients may still be able to generate abnormal movements, such as twisting at the elbow for tasks that require twisting at the wrist or rotating the back for tasks that require rotation at the shoulder. If future testing shows that these types of compensatory movements are important, we will track shoulder, elbow, and wrist movements with motion-tracking sensors to identify, and then try to correct by providing feedback, these abnormal movement patterns.

While the realism of functional task is a crucial factor in task-oriented training for stroke rehabilitation, training on the robot should also be motivating. In this respect, our system can be largely improved. For instance, performing the current tasks even with adaptive difficulty was not motivating to our subject. Since all interaction data are recorded; however, it will be straightforward to provide short-term and long-term performance feedback to the subjects to increase motivational levels.

Safety concerns were strongly addressed in the initial design process of ADAPT, and the subject did not feel threatened from the robot movement. However, to maximize safety, we need to perform long-term, intensive, and systematic tests with a greater number of subjects who are both healthy and with stroke.

The preliminary experiment with a healthy subject does not validate the efficacy of ADAPT for the rehabilitation of stroke patients. Therefore, after fully testing ADAPT with healthy subjects, we will begin pilot studies to test the safety and feasibility of our system with stroke patients. We will then test the efficacy of ADAPT in phase I clinical trials. We will compare pre- and posttraining scores on a number of instruments, with a focus on these measuring arm and hand function, and use, e.g., the Wolf motor function test [53] and the motor activity log [54]. Furthermore, we will test the possibility of using ADAPT as an automated tool for measuring arm and hand function in patients with stroke. In parallel with these clinical trials, we will use ADAPT to conduct experiments to determine highly effective adaptive task schedules.

Acknowledgment

The authors would like to thank M. Sandusky for his technical support and M. Mistry and C. Winstein for their comments on a previous draft.

References

[1] J. Desrosiers, F. Malouin, D. Bourbonnais, C. L. Richards, A. Rochette, and G. Bravo, "Arm and leg impairments and disabilities after stroke rehabilitation: Relation to handicap," *Clin. Rehabil.*, vol. 17, pp. 666–673, 2003.

[2] W. J. Coster, S. M. Haley, P. L. Andres, L. H. Ludlow, T. L. Bond, and P. S. Ni, "Refining the conceptual basis for rehabilitation outcome measurement: Personal care and instrumental activities domain," *Med. Care*, vol. 42, pp. I62–I72, 2004.

[3] J. H. Carr and R. B. Shepherd, *Stroke Rehabilitation—Guidelines for Exercise and Training to Optimize Motor Skill*. London, U.K.: Butterworth/Heinemann, 2003.

[4] S. Wolf, S. Blanton, H. Baer, J. Breshears, and A. J. Butler, "Repetitive task practice: A critical review of constraint induced therapy in stroke," *Neurologist*, vol. 8, pp. 325–338, 2002.

[5] G. Kwakkel, R. C. Wagenaar, J. W. Twisk, G. J. Lankhorst, and J. C. Koetsier, "Intensity of leg and arm training after primary middle-cerebral artery stroke: A randomized trial," *Lancet*, vol. 354, pp. 191–196, 1999.

[6] E. Taub, G. Uswatte, and R. D. Pidikiti, "Contstraint-induced (CI) movement therapy: A new family of techniques with broad application to physical rehabilitation—A clinical review," *J. Rehabil. Res. Dev.*, vol. 36, pp. 237–251, 1999.

[7] N. B. Lincoln, D. Willis, S. A. Philips, L. C. Juby, and P. Berman, "Comparison of rehabilitation practice on hospital wards for stroke patients," *Stroke*, vol. 27, pp. 18–23, 1996.

[8] R. A. Keith and K. S. Cowell, "Time use of stroke patients in three rehabilitation hospitals," *Soc. Sci. Med.*, vol. 24, pp. 529–533, 1987.

[9] M. L. Aisen, H. I. Krebs, N. Hogan, F. McDowell, and B. T. Volpe, "The effect of robot-assisted therapy and rehabilitative training on motor recovery following stroke," *Arch. Neurol.*, vol. 54, pp. 443–446, 1997.

[10] C. G. Burgar, P. S. Lum, P. C. Shor, and H. F. M. Van Der Loos, "Development of robots for rehabilitation therapy: The Palo Alto VA/Stanford experience," *J. Rehabil. Res. Dev.*, vol. 37, pp. 663–673, 2000.

[11] D. J. Reinkensmeyer, J. L. Emken, and S. C. Cramer, "Robotics, motor learning, and neurologic recovery," *Ann. Rev. Biomed. Eng.*, vol. 6, pp. 497–525, 2004.

[12] S. Hesse, G. Schulte-Tigges, M. Konrad, A. Bardeleben, and C. Werner, "Robot-assisted arm trainer for the passive and active practice of bilateral forearm and wrist movements in hemiparetic subjects," *Arch. Phys. Med. Rehabil.*, vol. 84, pp. 915–920, 2003.

[13] S. E. Fasoli, H. I. Krebs, J. Stein, W. R. Frontera, and N. Hogan, "Effects of robotic therapy on motor impairment and recovery in chronic stroke," *Arch. Phys. Med. Rehabil.*, vol. 84, pp. 477–482, 2003.

[14] L. R. MacClellan, D. D. Bradham, J. Whitall, B. Volpe, P. D. Wilson, J. Ohlhoff, C. Meister, N. Hogan, H. I. Krebs, and C. T. Bever, "Robotic upper-limb neurorehabilitation in chronic stroke patients," *J. Rehabil. Res. Dev.*, vol. 42, pp. 717–722, 2005.

[15] B. T. Volpe, H. I. Krebs, N. Hogan, L. Edelsteinn, C. M. Diels, and M. L. Aisen, "Robot training enhanced motor outcome in patients with stroke maintained over 3 years," *Neurology*, vol. 53, pp. 1874–1876, 1999.

[16] P. Lum, D. Reinkensmeyer, R. Mahoney, W. Z. Rymer, and C. Burgar, "Robotic devices for movement therapy after stroke: current status and challenges to clinical acceptance," *Top Stroke Rehabil.*, vol. 8, pp. 40–53, 2002.

[17] H. I. Krebs, M. Ferraro, S. P. Buerger, M. J. Newbery, A. Makiyama, M. Sandmann, D. Lynch, B. T. Volpe, and N. Hogan, "Rehabilitation robotics: Pilot trial of a spatial extension for MIT-Manus," *J. Neuroeng. Rehabil.*, vol. 1, p. 5, 2004.

[18] H. I. Krebs, B. T. Volpe, D. Williams, J. Celestino, S. K. Charles, D. Lynch, and N. Hogan, "Robot-aided neurorehabilitation: A robot for wrist rehabilitation," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 15, pp. 327–335, 2007.

[19] L. E. Kahn, W. Z. Rymer, and D. J. Reinkensmeyer, "Adaptive assistance for guided force training in chronic stroke," presented at the 26th Annu. Int. Conf. IEEE EMBS, San Francisco, CA, 2004.

[20] L. Dipietro, M. Ferraro, J. J. Palazzolo, H. I. Krebs, B. T. Volpe, and N. Hogan, "Customized interactive robotic treatment for stroke: EMG-triggered therapy," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 13, no. 3, pp. 325–334, Sep. 2005.

[21] J. Gordon, "Assumptions underlying physical therapy intervention: Theoretical and historical perspectives," in *Movement Science: Foundations for Physical Therapy in Rehabilitation*, 2nd ed., J. H. Carr and R. B. Shepherd, Eds. Rockville, MD: Aspen, 2000, pp. 1–31.

[22] B. Ahlsio, M. Britton, V. Murray, and T. Theorell, "Disablement and quality of life after stroke," *Stroke*, vol. 15, pp. 886–890, 1984.

[23] B. E. Fisher and K. J. Sullivan, "Activity-dependent factors affecting poststroke functional outcomes," *Top Stroke Rehabil.*, vol. 8, pp. 31–44, 2001.

[24] R. P. V. Peppen, G. Kwakkel, S. Wood-Dauphinee, H. J. Hendriks, P. J. V. der Wees, and J. Dekker, "The impact of physical therapy on functional outcomes after stroke: What's the evidence," *Clin. Rehabil.*, vol. 18, pp. 833–862, 2004.

[25] N. A. Bayona, J. Bitensky, K. Salter, and R. Teasell, "The role of task-specific training in rehabilitation therapies," *Top Stroke Rehabil.*, vol. 12, pp. 58–65, 2005.

[26] P. van Vliet, D. G. Kerwin, M. R. Sheridan, and P. Fentem, "The influence of goals on the kinematics of reaching forllowing stroke," *Neurol. Rep.*, vol. 19, pp. 11–16, 1995.

[27] C. Wu, C. A. Trombly, K. Lin, and L. Tickle-Degnen, "A kinematic study of contextual effects on reaching performance in persons with and without stroke: Influences of object availability," *Arch. Phys. Med. Rehabil.*, vol. 81, pp. 95–101, 2000.

[28] Y. Choi, F. Qi, J. Gordon, and N. Schweighofer, "Performance-based adaptive schedules enhance motor learning," *J. Mot. Behav.*, vol. 40, pp. 273–280, 2008.

[29] H. Goh, Y. G. Choi, J. C. Stewart, R. Lewthwaite, C. J. Winstein, and N. Schweighofer, "Grip force modulation after stroke: What is the optimal schedule?," presented at the WCPT, Vancouver, BC, Canada, 2007.

[30] R. A. Magill and K. G. Hall, "A review of the contextual interference effect in motor skill acquisition," in *Human Movement Science*, 9th ed. New York: Elsevier, 1990.

[31] R. J. Nudo, B. M. Wise, F. SiFuentes, and G. W. Milliken, "Neural substrates for the effects of rehabilitative training on motor recovery after ischemic infarct," *Sci.*, vol. 272, pp. 1791–1794, 1996.

[32] E. Plautz, G. W. Milliken, and R. J. Nudo, "Effects of repetitive motor training on movement representations in adult squirrel monkeys: Role of use versus learning," *Neurobiol. Learn Mem.*, vol. 74, pp. 27–55, 2000.

[33] T. D. Sanger, "Failure of motor learning for large initial errors," *Neural Comput.*, vol. 16, pp. 1873–1886, 2004.

[34] L. S. Vygotsky, *Mind in Society*. Cambridge, MA: Harvard Univ. Press, 1978.

[35] E. Deci and R. M. Ryan, *Intrinsic Motivation and Self-Determination in Human Behavior*. New York: Plenum, 1985.

[36] P. S. Lum, E. Taub, D. Schwandt, M. Postman, P. Hardin, and G. Uswatte, "Automated constraint-induced therapy extension (AutoCITE) for movement deficits after stroke," *J. Rehabil. Res. Dev.*, vol. 41, pp. 249–258, 2004.

[37] E. Taub, P. S. Lum, P. Hardin, V. W. Mark, and G. Uswatte, "AutoCITE: Automated delivery of CI therapy with reduced effort by therapists," *Stroke*, vol. 36, pp. 1301–1304, 2005.

[38] R. F. Erlandson, P. deBear, K. Kristy, M. Dijkers, and S. Wu, "A robotic system to provide movement therapy," presented at the 5th Int. Service Robot Conf., Detroit, MI, 1990.

[39] M. P. Dijkers, P. C. deBear, R. F. Erlandson, K. Kristy, D. M. Geer, and A. Nichols, "Patient and staff acceptance of robotic technology in occupational therapy: A pilot study," *J. Rehabil. Res. Dev.*, vol. 28, pp. 33–44, 1991.

[40] M. J. Johnson, K. J. Wisneski, J. Anderson, D. Nathan, and R. Smith, "Development of ADLER: The activities of daily living exercise robot," presented at IEEE-EMBS Biomed. Robot. (BioRob), Pisa, Italy, 2006.

[41] Y. G. Choi, J. Gordon, and N. Schweighofer, "ADAPT—Adaptive automated robotic task practice system for stroke rehabilitation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Pasadena, CA, 2008, pp. 2471–2476.

[42] C. Smith and H. I. Christensen, "Using COTS to construct a high performance robot arm," in *Proc. IEEE Int. Conf. Robots Automation*, Roma, Italy, April 10–14, 2007, pp. 4056–4063.

[43] Y. Yokokohji, R. L. Hollis, and T. Kanade, "WYSIWYF display: A visual/haptic interface to virtual environment," *Presence-Teleoperators Virtual Environ.*, vol. 8, pp. 412–434, 1999.

[44] B. M. Colton and J. M. Hollerbach, "Identification of nonlinear passive devices for haptic simulations," in *Proc. Symp. Haptic Interfaces Virtual Environ. Teleoperator Syst.*, 2005, pp. 363–368.

[45] S. Schaal and C. G. Atkeson, "Constructive incremental learning from only local information," *Neural Comput.*, vol. 10, pp. 2047–2084, 1998.

[46] H. Olsson, K. Astrom, C. de Wit, M. Gafvert, and P. Lischinsky, "Friction models and friction compensation," *Eur. J. Contr.*, vol. 4, pp. 176–195, 1998.

[47] C. Richard, "On the identification and haptic display of friction," Ph.D dissertation, Dept. Mech. Eng., Stanford Univ., Stanford, CA, 2000.

[48] C. Swindells and K. E. MacLean, "Capturing the dynamics of mechanical knobs," in *Proc. World Haptics Conf.*, 2007, pp. 194–199.

[49] Y. G. Choi and N. Schweighofer, "Capturing and modeling a tool dynamics for adaptive task practice in stroke rehabilitation," presented at the Workshop Assistive Technol. IROS, San Diego, CA, 2007.

[50] J. H. Cauraugh and S. Kim, "Two coupled motor recovery protocols are better than one—Electromyogram-triggered neuromuscular stimulation and bilateral movements," *Stroke*, vol. 33, pp. 1589–1594, 2002.

[51] J. Whitall, S. M. Waller, K. H. C. Silver, and R. F. Macko, "Repetitive bilateral arm training with rhythmic auditory cueing improves motor function in chronic hemiparetic stroke," *Stroke*, vol. 31, pp. 2390–2395, 2000.

[52] M. C. Cirstea, B. Leduc, and M. Levin, "Hemiparetic patients recruit additional degrees of freedom to compensate lost motor function," presented at 12th Congr. Int. Soc. Electrophysiol. Kinesiol., Montreal, QC, Canada, 1998.

[53] S. L. Wolf, P. A. Catlin, M. Ellis, A. L. Archer, B. Morgan, and A. Piacentino, "Assessing Wolf motor function test as outcome measure for research in patients after stroke," *Stroke*, vol. 32, pp. 1635–1639, 2001.

[54] G. Uswatte, E. Taub, D. Morris, M. Vignolo, and K. McCulloch, "Reliability and validity of the upper-extremity motor activity log-14 for measuring real-world arm use," *Stroke*, vol. 36, pp. 2493–2496, 2005.

Younggeun Choi (S'07) received the B.S. degrees in aerospace engineering and electrical engineering in 1998 from the Korea Advanced Institute of Science and Technology, Taejon, Korea, and the M.S. degree in computer science in 2005 from the University of Southern California, Los Angeles, where he is currently working toward the Ph.D. degree in computer science.

His current research interests include the design and control of rehabilitation robots, haptics, neuroscience, and machine-learning-based modeling.

James Gordon received the B.S. degree in physical therapy from State University of New York Downstate, New York, NY, and the M.S. and Ed.D. degrees from the Teachers College, Columbia University, New York.

He was a Physical Therapist at Kings County Medical Center, Brooklyn, NY, which has given him a broad vision of the need for rehabilitation of the upper extremities. He was a Researcher at Columbia University, where he was engaged in motor learning and motor control in healthy subjects and patients. He is currently the Chair of the Division of Biokinesiology and Physical Therapy, University of Southern California, Los Angeles. His current research interests include understanding the neural control of arm movements, especially the roles of proprioceptive information in control of reaching movements, and the development of effective rehabilitation based on this understanding.

Duckho Kim received the B.S. degree in electrical engineering from Seoul National University, Seoul, Korea, in 2006 and the M.S. degree in computer science from the University of Southern California (USC), Los Angeles, in 2008.

He is currently with the Division of Biokinesiology and Physical Therapy, USC.

Nicolas Schweighofer received the Engineering diploma in control systems and mechanical engineering from Ecole Supérieure de Mécanique de Nantes (newly Ecole Centrale de Nantes), Nantes, France, in 1990 and the Ph.D. degree in neuroscience from the University of Southern California (USC), Los Angeles, in 1995.

He was a Researcher in computational neuroscience with ATR, Kyoto, Japan. He was the Head of Research in a start-up company in Tokyo, Japan, where he was engaged in the development of adaptive learning software. He is currently an Assistant Professor with the Division of Biokinesiology and Physical Therapy, USC, where he holds joint appointments in Neuroscience and Computer Science. His current research interests include theoretical and experimental approaches to motor learning and decision making, with applications to neurorehabilitation.