7

Design and Implementation of a Control Architecture for Rehabilitation Robotic Systems

Duygun Erol & Nilanjan Sarkar
Vanderbilt University
USA

1. Introduction

Stroke is a highly prevalent condition especially among the elderly that results in high costs to the individual and society (Matchar & Duncan, 1994). According to the American Heart Association, in the U.S., approximately 700,000 people suffer a first or recurrent stroke each year (American Heart Association, 2006). It is a leading cause of disability, commonly involving deficits of motor function.

Recent clinical results have indicated that movement assisted therapy can have a significant beneficial impact on a large segment of the population affected by stroke or other motor deficit disorders. Experimental evidence suggests that intensive movement training of new motor tasks is required to induce long-term brain plasticity. The availability of movement training techniques, however, is limited by the amount of costly therapist's time they involve and the ability of the therapist to provide controlled, quantifiable and repeatable assistance to arm movement. Consequently, robot assisted rehabilitation that can quantitatively monitor and adapt to patient's progress, ensure consistency during rehabilitation may provide a solution to these problems.

In the last few years, robot-assisted rehabilitation for physical rehabilitation of the stroke patients has been an active research area to assist, monitor, and quantify rehabilitation therapies (Krebs et al., 2004, Lum et al., 2006, Kahn et al., 2006a, Kahn et al., 2006b, Loureiro et al., 2003). These robotic devices are used to recover arm movement after stroke, which provide opportunities for repetitive movement exercise and more standardized delivery of therapy with the potential of enhancing quantification of the therapeutic process. The first robotic assistive device used as a therapeutic tool, the MIT Manus (Krebs et al., 2003, Krebs et al., 2004) uses impedance controller to provide assistance to move patient's arm to the target position in an active assisted mode, where patients can visually see their movement and target location. In (Krebs et al., 2004) they expand the capabilities of MIT Manus to include motion in a three-dimensional workspace to rehabilitate other muscle groups and limb segments than shoulder and elbow. The Mirror Image Movement Enabler (MIME) and the Assisted Rehabilitation and Measurement (ARM) Guide, expanded the investigations of therapeutic applications of robots into the chronic stroke population. MIME uses a PUMA 560 manipulator to provide assistance to move the participant's arm with a pre-programmed position trajectory using Proportional-Integral-Derivative (PID) controller (Lum et al., 2006). ARM Guide is capable of generating both horizontal and vertical motion, and giving resistance and support to the patient (Kahn et al., 2006a, Kahn et al., 2006b). The

Source: Rehabilitation Robotics, Book edited by Sashi S Kommu,
ISBN 978-3-902613-04-2, pp.648, August 2007, Itech Education and Publishing, Vienna, Austria

GENTLE/s (Loureiro et al., 2003) is a haptic robot used to provide assistance to patients to move to the target positions along with a predefined path using admittance control. The participant's movement trajectory is represented in the virtual environment.

The promising results of the above-mentioned rehabilitation robotic systems indicate that robots could be used as effective rehabilitation tools. Current theories of stroke rehabilitation point towards paradigms of intense and repetitive use of the affected limb as a means for motor program reorganization. However, it has also been demonstrated in (Carey et al., 2005) that repetitive execution of simple motor tasks may not be as effective as execution of more complex motor tasks that involve in-depth cognitive processing. Precision-demanding tasks that challenge motor learning processes create richer conditions for change in the brain reorganization on rats (Black et al., 1990, Kleim et al., 2002), primates (Plautz et al., 2000, Nudo et al., 1996) and human (Pascual-Leone et al., 1995). It was shown that movement tracking training that requires cognitive processing achieved greater gains in performance than that of movement training that did not require cognitive processing (Carey et al., 2005). Additionally, it was shown that finger movement tracking training produced greater gains in the range of motion and tracking accuracy compared to finger movement training that required no temporospatial processing (Carey et al., 2006). Thus, it would be useful if a tracking movement training method can be developed, where the patients not only make repetitive movement but also pay attention to tracking accuracy. However, in such a tracking task, patients may not be able to track the desired motion because of their impairments. Thus, a low-level controller can be designed to provide assistance to the patient to track the desired motion accurately based on his/her performance.

The existing robotic rehabilitation systems primarily use low-level assistive controllers to assist the movement of patients' arms. For example, MIT Manus uses an impedance controller, MIME uses a PID controller and GENTLE/s uses an admittance controller for movement assistance. In some cases, the rehabilitation system keeps track of the status of the task (e.g., AutoCITE (Taub et al., 2005)). However, to our knowledge, none of these systems has a dedicated high-level controller that can comprehensively monitor the task, provide assessment of the progress, and alter the task parameters to impart effective therapy based on patient's performance in an automated manner. Instead, in these existing robotic rehabilitation systems, a therapist administers the therapy where he/she monitors the progress of the tasks, patient's safety, and assess whether the task needs to be updated based on current condition of the therapy. As a result, it is likely to consume more time of the therapist, increase workload of the therapist, and consequently, increase the cost of treatment. In the current work, we present the design and development of a high-level controller that work in conjunction with the low-level controllers such that it can determine the task updates dynamically based on patients' performance; and monitor the safety related events in an automated manner and generate an accommodating plan of action.

In this chapter, we present a new control architecture which consists of a low-level controller and a high-level controller. The low-level controller is used to provide robotic assistance as and when needed to the participants to complete an upper arm rehabilitation task. This task is designed to impart movement training that requires cognitive processing. The high-level controller is used to monitor the progress of the rehabilitation task and make decisions on the modification of the task that might be needed for the therapy. In order to demonstrate the efficacy of the proposed control architecture, we needed to develop a rehabilitation robotic system, which is also presented in this chapter.

This chapter is organized as follows. It first presents the overall control architecture in Section 2. Then the rehabilitation robotic system is presented in Section 3. The low-level controller and the high-level controller have been described in Section 4 and Section 5, respectively. Results of the experiments are presented in Section 6 to demonstrate the efficacy of the low-level controller and the high-level controller on unimpaired participants. Section 7 discusses potential contributions of this work and possible directions for future work.

2. Control Architecture

The patients are asked to perform a rehabilitation task. However, the patients may not be able to complete the rehabilitation task because of their motor impairment. A low-level controller could be used to provide robotic assistance to participants' arm movement as and when needed to help them to complete the reaching task. Note that various robot, human and general task related information, called events, could affect the reaching task. For example, if the robot joint motor develops any fault; or if the patient feels uncomfortable he/she might want to stop the task; or the patient is more than capable of performing the current task and he/she needs more challenging task etc. These set of information may require some adjustments of the planning of the task. As a result, the low-level controller also needs to be aware of these adjustments of the task to accomplish the therapy requirements.

In order to provide therapy that can accommodate the above requirements, a high-level controller could be used in conjunction with the low-level controller that monitors the task and patient's safety and informs the low-level controller about the task updates. The high-level controller in here plays the role of a human supervisor (therapist) who would otherwise monitor the task and assess whether the task needs to be updated. However, in general, the high-level controller and the low-level controller cannot communicate directly because each may require different types of inputs and outputs. For example, a high-level controller may operate in the discrete domain whereas a low-level controller may operate in the continuous domain. Thus an interface is required which can convert continuous-time signals to sequences of discrete values and vice versa. Hybrid system theory provides mathematical tools that can accommodate both continuous and discrete system in a unified manner. As a result, in this work, we take the advantage of using a hybrid system model to design our control architecture. A hybrid system model has three parts, a "Plant", a "Controller" (supervisor) and an Interface (Koutsoukos et al., 2000, Antsaklis & Koutsoukos, 2003). In order to avoid confusion about terminology, we call the "Controller" in hybrid system model a high-level controller. The continuous part, identified as the "Plant" is the low-level controller. Fig. 1 presents the proposed control architecture. There has been no work to our knowledge on designing such a hybrid system for rehabilitation purposes. However, in this chapter, we argue that such a hybrid system framework could be useful in automating robotic rehabilitation and providing important aid to the therapist. Hybrid control framework has been effectively used in other fields, such as industrial robotics, medicine, and manufacturing (Antsaklis & Koutsoukos, 2003).

In this architecture (Fig. 1), the state information from the robot and the human is monitored by the process-monitoring module through the interface to trigger the relevant events. Each event is represented as a plant symbol so that the high-level controller can recognize the event. Once the high-level controller receives the event through a plant symbol, the decision

making module of the high-level controller generates sequences of control actions using its decision rules. The high-level controller is designed considering the need of the therapist and the patient and it can be easily modified and extended for new task requirements. The decision of the high-level controller is sent to the low-level controller through the interface using the control symbols. Interface converts the control symbols to the plant inputs which are used to update the task. The updated task is then executed by the low-level controller. This cycle continues to complete the therapy.

The proposed control architecture is flexible and extendible in the sense that new events can be included and detected by simply monitoring the additional state information from the human and the robot, and accommodated by introducing new decision rules and new low-level controllers.

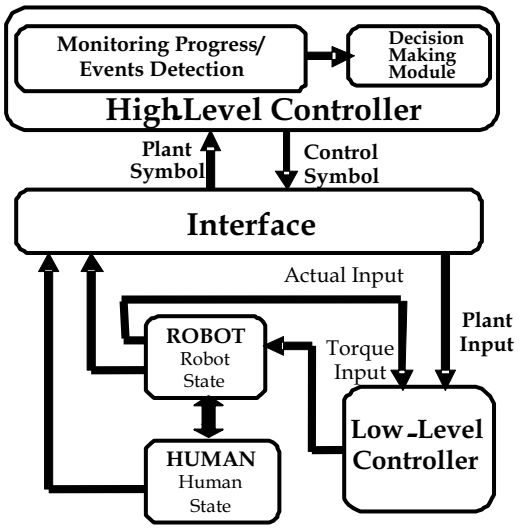


Fig. 1. Control Architecture.

3. The Rehabilitation Robotic System

A PUMA 560 robotic manipulator is used as the main hardware platform in this work. The manipulator is augmented with a force-torque sensor and a hand attachment device (Fig. 2).

3.1 Hardware

The PUMA 560 is a 6 degrees-of-freedom (DOF) device consisting of six revolute joints (PUMA web site). In order to record the force and torque applied by the human, an ATI Gamma force/torque sensor is used. The robot is interfaced with Matlab/Real-time Workshop to allow fast and easy system development. The force values recorded from the force/torque sensor are obtained using a National Instruments PCI-6031E data acquisition card with a sampling time of 0.001 seconds. The joint angles of the robot are measured using encoder with a sample time of 0.001 seconds from a Measurement Computing PCI-QUAD04 card. The torque output to the robot is provided by a Measurement Computing PCIM-DDA06/16 card with the same sample time. A computer monitor is placed in front of the

participant to provide visual feedback about his/her motion trajectory during the execution of the task.

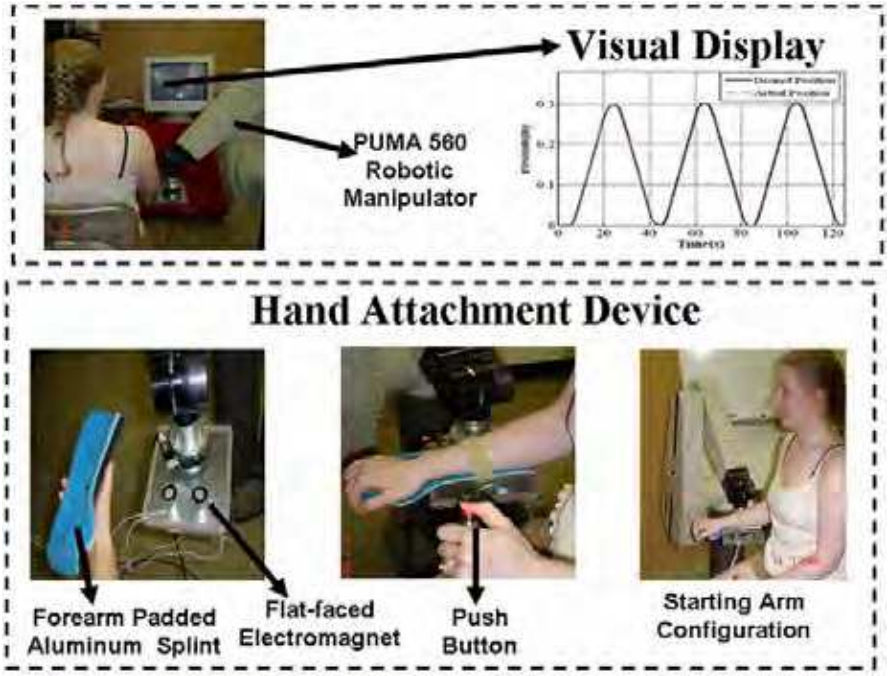


Fig. 2. Participant Arm Attached to Robot.

3.2 Hand Attachment Device

Since in this work we are primarily interested in effecting assistance to the upper arm, we design a hand attachment device where the participant's arm is strapped into a splint that restricts wrist and hand movement. The PUMA 560 is attached to that splint to provide assistance to the upper arm movement using the low-level controller (Fig. 2). Forearm padded aluminum splint (from MooreMedical), which ensures the participant's comfort, is used as a splint in this device. We further design a steel plate with proper grooves that hold two small flat-faced electromagnets (from Magnetool Inc.) that are screwed on it. This plate is also screwed with the force-torque sensor, which provides a rigid connection with the robot. We attach a light-weight steel plate under the splint, which is then attached to the electromagnets of the plate. These electromagnets are rated for continuous duty cycle (100% duty cycle), i.e., they can run continuously at normal room temperature. Pull ratings of these magnets are 40lb. We have used two electromagnets to have a larger pulling force to keep the splint attached to the hand attachment device. An automatic release (AU) rectifier controller (Magnetool Inc.) has been used to provide a quick, clean release of these electromagnets. A push button, which has been connected to the AU Rectifier Controller, is used to magnetize and demagnetize the electromagnets when the participant wants to remove the hand attachment device from the robotic manipulator in a safe and quick manner.

3.3 Discussion on Safety of the Rehabilitation Robotic System

Ensuring safety of the participant is a very important issue when designing a rehabilitation robotic system. Thus, in case of emergency situations, therapist can press emergency button. The patient and/or the therapist can quickly release the patient's arm from the PUMA 560 by using the quick-release hand attachment device (as described above) to deal with any physical safety related events. In order to release the participant's arm from the robot, the push button is used. When the push button is pressed electromagnets are demagnetized instantaneously and the participant is free to remove the splint from the robot. This push button can also be operated by a therapist. Additionally, we have covered the corner of the arm device with a foam self stick tape in order to avoid sharp surface.

4. Low-Level Controller

The objectives of the current work is to: i) design an upper arm movement rehabilitation task that requires cognitive processing as well as could contribute to a variety of functional daily living activities, and ii) design a controller to provide robotic assistance to help participants to perform the above movement rehabilitation task. In what follows we present the basic design of the task and the low-level controller.

4.1. Task Design

Let us first briefly review the task design of some well-known robotic rehabilitation systems. MIT Manus uses impedance controller to provide assistance to move patient's arm to the target position in an active assisted mode, where patients can visually see their movement and target location (Krebs et al., 2003, et al., 2004). MIME provides assistance to move the participant's arm with a pre-programmed position trajectory using proportional-integral-derivative (PID) controller (Lum et al., 2006). The participant is asked to maintain a specified off-axis force while they are trying to reach toward a goal position using ARM Guide (Kahn et al., 2006a, Kahn et al., 2006b). The GENTLE/s provides assistance to patients to move to the target positions along with a predefined path using admittance control. The participant's movement trajectory is represented in the virtual environment in (Loureiro et al., 2003). The therapy tasks designed for the rehabilitation robotic devices require predominantly shoulder motion or elbow motion, or some of them require the combination of both shoulder and elbow motion.

We choose a reaching task that is commonly used for rehabilitation of upper extremity after stroke. In this task, the participants are asked to move their arms in the forward direction to reach a desired point in space and then bring it back to the starting position repeatedly within a specified time. In other words, they have to follow a desired position trajectory. The reaching task designed in here requires combination of the shoulder and elbow movement which could increase the active range of motion (AROM) in shoulder and elbow in preparation of later functional reaching activities in rehabilitation. The allowable motion is restricted only to the direction of the task. For example, if the task requires the participants to move their arms in the Y-direction, then they will not be able to move their arms in X or Z directions. However, they can move their arms in the Y-direction at a velocity that could be the same, higher or lower than the desired velocity. The idea here is to improve the ability of participant's arm movement in one direction at a time by helping them to improve their speed of movement. Improving the speed of movement for such tasks

is an important criterion to measure the success of a therapy. For example, in Constraint Induced Movement Therapy (CIMT) (Taub et al., 1999) during the performance of the wipe table task, participants are required to complete as many back and forth motion as possible in a certain amount of time across the table and back between the two targets. The number of times of the completed movement in a certain amount of time is used as a metric to evaluate the participants' progress. If participants can improve their speed of movement, the metric described above will capture this progress. In this work, we constrain the motion of the arm in the horizontal plane and in one direction (along the Y-axis). Although, in this work the motion of the arm is constrained in the horizontal plane in one direction (along the Y-axis), it could also be designed for other directions (e.g., X-axis) or combination of directions (e.g., XY-axes) based on task requirements (only shoulder or elbow motion or the combination of shoulder and elbow motion).

In order to include cognitive processing within this reaching task, we ask the participants to follow a visually presented desired motion trajectory that is likely to command their concentration. The participants receive visual feedback of both their actual position and the desired position trajectories on a computer screen, which is placed in front of them. They are asked to pay attention to tracking the desired position trajectory as accurately as possible, which keeps them focused on the task. The visual feedback is used not only to inform the participants of how closely they are tracking the desired motion but also as a motivational factor to keep them focused on the task. The tip of the position trajectory that the participant is required to follow represents the velocity of the task trajectory.

The task presented here incorporates cognitive processing by asking the participants to follow the tip of the visually presented trajectory. The tip of the trajectory represents the current desired velocity. By asking the participant to follow the tip makes him/her focused on the task. This task is different from other tasks that have been used in the context of robotic rehabilitation in that here we are interested in improving the speed of motion in one direction at a time using visual feedback, which could be useful in a number of therapy tasks.

4.2. Controller Design

The controller designed in this work is responsible for providing robotic assistance to a participant to complete the movement tracking task in an accurate manner. The existing robotic rehabilitation systems operate in robot task-space to provide robotic assistance to the patients to follow a desired trajectory to complete a rehabilitation task (Krebs et al., 2004, Lum et al., 2006, Kahn et al., 2006a, Kahn et al., 2006b). Recently, a human-arm joint impedance controller is proposed, which operates in joint-space, to provide assistance to participants to follow desired joint angle trajectory (Culmer et al., 2005) specified for each individual joint (e.g., elbow joint). It is still not clear, however, whether the assistance in the task-space or in the joint-space will likely to have the best results for rehabilitation purposes. In this work, we design a controller that is responsible for providing the robotic assistance to participants to complete a rehabilitation task in task-space (Erol & Sarkar, 2007). In this controller, an outer force feedback loop is designed around an inner position loop (Fig. 3). The tracking of the reference trajectory is guaranteed by the inner motion control (Sciavicco & Siciliano, 1996). The desired force, which is given as a force reference to the controller, is computed by a planner. The proposed controller is similar to an impedance controller; however it allows specifying the reference time varying force directly. The equations of motion for the robot are given by:

$$\Gamma = M(q)\ddot{q} + Co(q)(q,\dot{q}) + Ce(q)\left|\dot{q}^2\right| + G(q) \qquad (1)$$
$$u - J^T(q)F = M(q)\ddot{q} + V(q,\dot{q}) + G(q)$$

where $M(q)$ represents the inertia matrix, $V(q,\dot{q})$ is the summation of the matrix of coriolis torques $Co(q)(q,\dot{q})$ and centrifugal torques $Ce(q)\left|\dot{q}^2\right|$, $G(q)$ is the vector of gravity torques. Γ is the generalized joint force torque which is calculated using $u - J^T(q)F$, where u is the input to the manipulator, $J(q)$ is the Jacobian matrix and F is the contact force exerted by the manipulator. Using inverse dynamics control, manipulator dynamics are linearized and decoupled via a feedback. The dynamic equation of the robotic manipulator was given in (1). Control input u to the manipulator is designed as follows:

$$u = M(q)y + V(q,\dot{q}) + G(q) + J^T F \qquad (2)$$

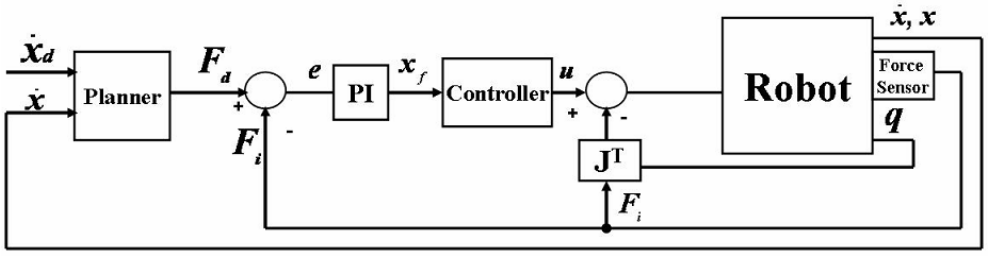


Fig. 3. Low-Level Controller.

which leads to the system of double integrators

$$\ddot{q} = y \qquad (3)$$

In (3), y represents a new input. The new control input y is designed so as to allow tracking of the desired force F_d. To this purpose, the control law is selected as follows:

$$y = J(q)^{-1}M_d^{-1}(-K_d\dot{x} + K_p(x_f - x) - M_d\dot{J}(q,\dot{q})\dot{q}) \qquad (4)$$

where x_f is a suitable reference to be related to force error. M_d(mass), K_d (damping) and K_p (stiffness) matrices specify the target impedance of the robot. x and $\dot{x}$ are the position and velocity of the end-effector in the Cartesian coordinates, respectively. The relationship between the joint space and the Cartesian space acceleration is used to determine position control equation.

$$\ddot{x} = J(q)\ddot{q} + \dot{J}(q,\dot{q})\dot{q} \quad and \quad \ddot{x} = J(q)y + \dot{J}(q,\dot{q})\dot{q} \qquad (5)$$

By substituting (4) into (5), we obtain

$$\ddot{x} = J(q)(J(q)^{-1}M_d^{-1}(-K_d\dot{x} + K_p(x_f - x) - M_d\dot{J}(q,\dot{q})\dot{q})) + \dot{J}(q,\dot{q})\dot{q}$$
$$\ddot{x} = -M_d^{-1}K_d\dot{x} + M_d^{-1}K_p(x_f - x) \qquad (6)$$
$$M_d\ddot{x} + K_d\dot{x} + K_p x = K_p x_f$$

Equation (6) shows the position control tracking of x with dynamics specified by the choices of K_d, K_p and M_d matrices. Impedance is attributed to a mechanical system characterized by these matrices that allows specifying the dynamic behavior. Let F_d be the desired force reference, which is computed using a PID velocity loop:

$$F_d = P_d(\dot{x}_d - \dot{x}) + I_d \int(\dot{x}_d - \dot{x})dt + D_d \frac{d(\dot{x}_d - \dot{x})}{dt} \tag{7}$$

where $\dot{x}_d$, $\dot{x}$, P_d, I_d and D_d are the desired velocity, actual velocity, the proportional, integral and derivative gains of the PID velocity loop, respectively. The relationship between x_f and the force error is expressed in (8) as:

$$x_f = P(F_d - F_i) + I\int(F_d - F_i)dt \tag{8}$$

where P and I are the proportional and integral gains, respectively, and F_i is the force applied by the human. Equations (6) and (8) are combined to obtain below equation:

$$M_d\ddot{x} + K_d\dot{x} + K_p x = K_p(P(F_d - F_i) + I\int(F_d - F_i)dt) \tag{9}$$

We can observe from (9) that the desired force response is achieved by controlling the position of the manipulator.

4.3. Decision of Robotic Assistance during Task Execution

During the tracking task, the activation of the low-level controller to provide robotic assistance is decided based on the participant's actual velocity ($\dot{x}$). If the actual velocity lies within an acceptable band, then it is understood that the participant is able to track the trajectory without robotic assistance. The acceptable band consists of upper and lower bounds on velocity, which are defined as:

$$\dot{x}_{upper} = \dot{x}_d + \left(\dot{x}_d * \frac{percentage}{100}\right), \quad \dot{x}_{lower} = \dot{x}_d - \left(\dot{x}_d * \frac{percentage}{100}\right) \tag{10}$$

where *percentage* is the value used to increment and decrement the desired velocity to define the upper and lower velocities for the selected $\dot{x}_d$. If the $\dot{x}$ is not between $\dot{x}_{upper}$ and $\dot{x}_{lower}$, then the low-level controller is activated to provide assistance to keep the participant's motion in the desired velocity range. However, note that any participant will require a finite amount of time to generate the desired motion. The controller should not be activated until it is determined that the participant is not able to generate the required motion by his/her own effort. Thus, initially a desired $\dot{x}_d$ is decided and it's upper ($\dot{x}_{upper}$) and lower ($\dot{x}_{lower}$) bound is calculated using (10). In order to determine the velocity trajectories $\dot{x}_d(t)$, $\dot{x}_{upper}(t)$ and $\dot{x}_{lower}(t)$, we use a generator block to generate smooth velocity trajectories within a specified distance using a skew-sine function. As a result, we define an algorithm to determine the average velocity of the participant $\dot{x}_{ave}$ (as opposed to instantaneous velocity) and average value of the upper $\dot{x}_{upperave}$ and lower $\dot{x}_{lowerave}$ velocity bounds for a given period of time, which are used to decide if the robotic assistance is needed. $\dot{x}_{ave}$, $\dot{x}_{upperave}$ and $\dot{x}_{lowerave}$ are calculated using the equations:

$$\dot{x}_{ave} = \frac{1}{\left(\frac{tf-ti}{ts}\right)} \sum_{t=ti}^{tf} (\dot{x}(t)), \dot{x}_{lowerave} = \frac{1}{\left(\frac{tf-ti}{ts}\right)} \sum_{t=ti}^{tf} (\dot{x}_{lower}(t)), \dot{x}_{upperave} = \frac{1}{\left(\frac{tf-ti}{ts}\right)} \sum_{t=ti}^{tf} (\dot{x}_{upper}(t)) \quad (11)$$

where tf, ti and ts are the final time, starting time and sampling time, respectively. $\dot{x}(t)$ is the participant's actual velocity at time t. If $\dot{x}_{lowerave} < \dot{x}_{ave} < \dot{x}_{upperave}$ is satisfied, then the low-level controller is not activated and participant continue tracking task without robotic assistance. If $\dot{x}_{lowerave} < \dot{x}_{ave} < \dot{x}_{upperave}$ is not satisfied then the controller is activated to provide robotic assistance to the participant to track the desired motion.

4.4. Switching Mechanism

Note that the controller will be switching in and out to provide robotic assistance. In order to ensure smooth switching, a switching mechanism that we have previously shown to guarantee bumpless switching for satisfactory force response (Mallapragada et al., 2006) is used in this work. This mechanism modifies the position reference, which is the input for the inner loop of the force controller, at the time of the switching in such a way that it is equal to the position reference at the time before switching occurred. The control action in (8) can be modified as below:

$$x_{fp}(t) = x(t) \quad and \quad x_{ff}(t) = Pe(t) + I(X_i(t) + X_{io})dt \quad (12)$$

Here $x_{fp}(t)$ is the position reference when the controller is not active, which is equal to the position of the human/robot $x(t)$. $x_{ff}(t)$ is the position reference determined using the *P* and *I* gains when the controller is active. $X_i(t)$ represents the integral action and X_{io} is the initial condition of the error integrator. $e(t)$ is defined as the $F_d - F_i$. If t_s is the time of switching, then equation (12) can be used to find the position reference just before the time of switching.

$$x_{fp}(t_s^-) = x(t_s^-) \quad (13)$$

where $x(t_s^-)$ represents the position of the human/robot right before the switching occurred. The position reference just after the switching is given as:

$$x_{ff}(t_s^+) = Pe(t_s^+) + I(X_i(t_s^+) + X_{io})dt \quad (14)$$

The integral action associated with the controller is reset during the switching so that:

$$X_i(t_s^+) = 0 \quad (15)$$

The force error defined as $F_d - F_i$ is set to zero just after the time of the switching for a small period of time. Hence:

$$Pe(t_s^+) = 0 \quad (16)$$

After the time of the switching F_d which is calculated using (7), and F_i, which is recorded from the force sensor are provided to the controller. The initial condition X_{io} is defined as:

$$X_{io} = x(t_s^-)/I \tag{17}$$

Then, substituting (15)-(17) into (14) we can observe that

$$x_{ff}(t_s^+) = x_{fp}(t_s^-) \tag{18}$$

This relation ensures that the position reference is indeed continuous during switching which guarantees bumpless activation and deactivation of the low-level controller.

5. The High-Level Controller

The high-level controller monitors the progress of the task, the status of the plant, and makes decision on the modification of the task that might be needed for the therapy. The high-level controller decisions are executed by the low-level controller to accomplish the task requirements. In this section, we first present the theory of the high-level controller, followed by the design rationale and details of the high-level controller.

5.1. Model

The high-level controller is a discrete-event system (DES) deterministic finite automaton, which is specified by $D = (\tilde{P}, \tilde{X}, \tilde{R}, \psi, \lambda)$ (Koutsoukos et al., 2000, Antsaklis & Koutsoukos, 2003). Here $\tilde{P}$ is the set of discrete states. Each event is represented as a plant symbol, where $\tilde{X}$ is the set of such symbols, for all discrete states. The next discrete state is activated based on the current discrete state and the associated plant symbol using the following transition function: $\psi : \tilde{P} \times \tilde{X} \to \tilde{P}$. In order to notify the low-level controller the next course of action in the new discrete state, the controller generates a set of symbols, called control symbols, denoted by $\tilde{R}$, using an output function: $\lambda : \tilde{P} \to \tilde{R}$. The action of the high-level controller is described by the following equations:

$$\tilde{p}_j[n] = \psi(\tilde{p}_i[n-1], \tilde{x}_k[n]) \tag{19}$$

$$\tilde{r}_c[n] = \lambda(\tilde{p}_j[n]) \tag{20}$$

where $\tilde{p}_i, \tilde{p}_j \in \tilde{P}$, $\tilde{x}_k \in \tilde{X}$ and $\tilde{r}_c \in \tilde{R}$. i and j represent the index of discrete states. k and c represent the index of plant symbols and control symbols, respectively. n is the time index that specifies the order of the symbols in the sequence.

5.2. Design Rationale for the High-Level Controller

Let us explain the role of each element of the automaton $D = (\tilde{P}, \tilde{X}, \tilde{R}, \psi, \lambda)$ in the context of rehabilitation tasks. $\tilde{P}$ is the set of discrete states. A rehabilitation therapy may consist of several actions and each discrete state may capture one of these actions. The action that takes place in each discrete state could be used to update the rehabilitation task. For example, if improving the speed of motion is the objective, then each category of speed (e.g., slow, medium, fast etc.) could be chosen as discrete states. When new actions are required for a rehabilitation task, new states can easily be included in the set of the states, $\tilde{P}$. Once the set $\tilde{P}$ is chosen, the next design parameters are what are called "events" that could affect the rehabilitation task. Events are various robot, human and general task related information that provide the current status of the task.

The set of events are not unique and are decided considering the need of the therapy, and the capabilities of the rehabilitation robotic systems. Generally the available sensory information from the robotic systems and the input from the therapist and the participant provide the core of the set of the events. When these events occur it may require some adjustments of the planning of the rehabilitation task. As discussed earlier, this sensory information may not be directly interpreted by the high-level controller. As a result, each event is represented as a plant symbol so that the high-level controller can recognize the events. $\tilde{X}$ is the set of the plant symbols, which is designed based on the set of events. The transition function $\psi : \tilde{P} \times \tilde{X} \rightarrow \tilde{P}$ uses the current state and the plant symbol to determine the next action that is required to update the rehabilitation task. For example, when the participant is performing the rehabilitation task and an event that requires the task to be stopped occurs, then the transition function is used to transit from one active state, which executes the task as required, to another one, which stops the task execution, based on the event. The high-level controller generates a control symbol, which is unique for each state, using the output function λ. $\tilde{R}$ is the set of the control symbols. The output of the control symbols are plant inputs which is in charge of the modification of the rehabilitation task. The control symbols and its outputs are decided based on the task requirements and the abilities of the low-level controller. For example, if the objective of the rehabilitation task is to increase the participant's range of motion, then the control symbol generates plant inputs to the low-level controller to change the desired goal position of the task in order to make the task more/less challenging for the participant. It is clear from the above discussion that the design of the various elements of the automaton $D = (\tilde{P}, \tilde{X}, \tilde{R}, \psi, \lambda)$ is not unique and is dependent on the task at hand, and sensory information available from the robotic system. In what follows we present the design of these elements with regard to the objective of the rehabilitation task we present in this chapter.

The design of the elements of $D = (\tilde{P}, \tilde{X}, \tilde{R}, \psi, \lambda)$ for the reaching task that has been described in Section 4.1 is motivated by the specific objective of the task. In here, the objective of the reaching task is to improve the participant's speed of movement while considering the current movement ability of the participant and the safety of the task. The participant is required to complete the movement in a certain amount of time, which represents the velocity of the task trajectory. The desired velocity trajectory could be updated to improve the participant's speed of movement and to ensure the safety of the participant. Thus the discrete states could be the level of speed at which the therapy is imparted to the participant. In order to decide the set of events, all sensory information that the current rehabilitation robotic systems can generate is analyzed. The rehabilitation robotic system used in this work has a force sensor to record the applied force of the participant, a PCI card to record the robot joint angles, and pause, stop and restart buttons for task execution. A counter is also used to record the number of times participant needed robotic assistance to determine the improvement of participant's movement ability. This set of information is used to define several events in our work. Once the discrete states and the events are determined, the necessary plant and control symbols are designed based on the structure of the high-level and low-level controllers, and the objectives of the task (e.g., when should discrete states be changed, how to increase or decrease speed etc.). The design details of the high-level controller for the reaching task are given in the next section.

5.3. Design Details of the High-Level Controller

We initially define the following discrete states $\tilde{p}$: stay, difficult, easy, stop and pause. Stay ($\tilde{p}_1$) implies the participant needs to continue the task at the same difficulty level by keeping the desired velocity same. Difficult ($\tilde{p}_2$) means the participant has improved his/her task performance and task need to be more challenging by increasing the desired velocity. Similarly, easy ($\tilde{p}_3$) implies changing the task parameters to make the task easier by decreasing the desired velocity. Stop ($\tilde{p}_4$) and pause ($\tilde{p}_5$) are defined in their usual ways. New discrete states can easily be included in the design of the high-level controller when new control actions are needed to modify the task parameters.

The state information from the robot and the human is detected to define the events. The state information from the robot and the human can be a continuous signal or a discrete value. Let S_{Rn} and S_{Hn} represent the sets of robot and human state information, respectively. In this research, the continuous signals that are detected from the robot are: i) robot's joint angles (S_{R1}), ii) the force reference calculated using (7) (S_{R2}), iii) the participant's velocity, which is measured from the tool frame velocity (S_{R3}). The discrete value detected from the robot is the participant's progress during the tracking task (S_{R4}). In order to find S_{R4}, the number of times participant needed robotic assistance at 10th trial (n_{10}) and at 50th trial (n_{50}) were recorded. Decision logic is defined to determine the value of S_{R4} using (21). Δp is the percentage value that is used to detect the improvement of the participant's movement ability in terms of the number of times he/she needs assistance from the robotic device, which is likely to be specified by the therapist based on individual progress.

$$\begin{aligned} &if \quad n_{50} < \left(n_{10} - \left(n_{10} * \frac{\Delta p}{100} \right) \right) \quad then \quad \{S_{R4}=1\} \\ &elseif \quad n_{50} > \left(n_{10} + \left(n_{10} * \frac{\Delta p}{100} \right) \right) \quad then \quad \{S_{R4}=-1\} \\ &else \quad \{S_{R4}=0\} \end{aligned} \tag{21}$$

Robot and human state information is monitored to trigger relevant events to modify the task. When these events are triggered, the interface provides the necessary plant symbol ($\tilde{x}$) to the high-level controller. Currently we have defined nine events for the proposed high-level controller. However, the number of events can be easily extended. Five of these (E1, E2, E3, E4 and E5) are robot generated, and three of these (E6, E7 and E8) are human generated events. The other event, which is a secondary event, is called SE1. This is used to detect the previous state when the participant wants to continue with the task after he/she stops. The high-level controller needs to know which state was active before the pause or stop button was pressed in order to provide the same task parameters to the participant when he/she resumes the task. For example, when the participant presses pause button, a value is assigned to SE1. This value is retrieved when the participant resumes the task so that he or she can continue the therapy with the same task requirements. Events are reset at the beginning of task execution. Additionally, the triggered event is reset when a new event occurs. When the participant requires less, more or same level of robotic assistance to track the desired trajectory, E1, E2 and E3 is triggered, respectively. E4 occurs when the robot's joint angles are out of range. If the force reference (calculated by (7)) provided to the low-level controller to assist the participant and the participant's velocity ($\dot{x}$) are above predefined threshold values, then E5 and E6 are triggered, respectively. E7 occurs when the

participant presses the pause or the stop button. In order to continue with the task, the participant resets the pause button and E8 event is triggered. Plant symbols ($\tilde{x}$) are designed based on the events as shown in Table 1. The *joint_limits* are known from the robot's specifications. $F_{dthreshold}$ and $\dot{x}_{threshold}$ are determined by the therapist at the beginning of the task execution. Note that if any of E4, E5, E6, and E7 or their combinations occurs then the state stop ($\tilde{p}_4$) is activated. Thus we assign the same plant symbol, $\tilde{x}_4$ for these events.

The secondary event, SE1, is defined as follows: if the state is difficult and E7=1, then SE1=1. We assign a corresponding plant symbol $\tilde{x}_6$. Similarly, if the state is easy and E7=1, then SE1=2, and the plant symbol $\tilde{x}_7$ is assigned. If the state is stay and E7=1, then SE1=3. We assign a corresponding plant symbol $\tilde{x}_8$. SE1 releases state information when E7=0 and E8=1.

Table 1. Plant Symbols for the High-Level Controller.

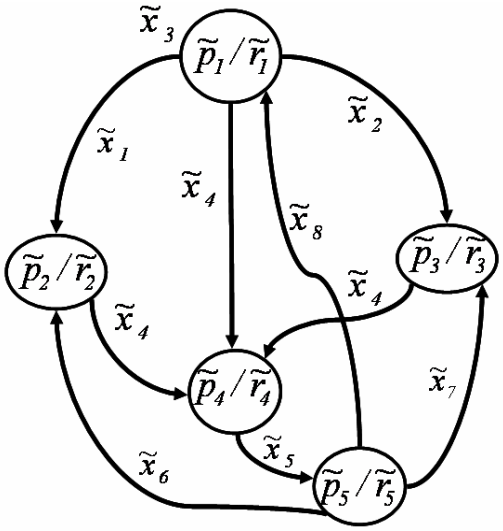
Signals from Human and Robot	Event Triggered	Plant Symbol
S_{R4}=1	E1=1	$\tilde{x}_1$
S_{R4}=-1	E2=1	$\tilde{x}_2$
S_{R4}=0	E3=1	$\tilde{x}_3$
S_{R1}>joint_limits or S_{R2}> $F_{dthreshold}$ or S_{R3}> $\dot{x}_{threshold}$ or S_{H1}=1	E4=1 E5=1 E6=1 E7=1	$\tilde{x}_4$
S_{H2}=1	E8=1	$\tilde{x}_5$

When any of these events is triggered, the high-level controller decides the next plan of action to modify the task. When an event is triggered, the corresponding plant symbol ($\tilde{x}$) is generated by the interface. The current state ($\tilde{p}$) and the plant symbol ($\tilde{x}$) are used by the high-level controller to determine the next state. Then the high-level controller generates the corresponding control symbol ($\tilde{r}$) for this new state and provides it to the interface. The add feasible paths in the proposed high-level controller is shown in Fig. 4 (left). In this figure, $\tilde{r}_c$ s are corresponding control symbols for each plant symbol $\tilde{x}_k$, where *c=1,2,…5* and *k=1,2,…8*. Any event that generates corresponding plant symbols $\tilde{x}_k$ along with the current state information $\tilde{p}_i$ determines the next $\tilde{p}_j$ and as a result, $\tilde{r}_c$, where *i=1,2,…5* and *j=1,2,…5*. In our application only one state is active at any given time, and therefore we

uniquely assign a control symbol $\tilde{r}_i$ for each discrete state $\tilde{p}_i$. Since the low-level controller cannot interpret the control symbols, the interface converts them to the appropriate values for α and β for (22) to execute the task. The available control symbols $\tilde{r}_i$ and their corresponding α and β values for the plant input are defined in a table in Fig. 4 (right).
The plant equation which determines the desired velocity for the low-level controller is defined as:

$$\dot{x}_{dm} = \beta(\dot{x}_d + (\alpha * delta)) \tag{22}$$

where $delta$ is selected as a constant value to increase and decrease the $\dot{x}_d$, which makes the task more or less challenging. $\dot{x}_{dm}$ is the new desired velocity value used to determine the new $\dot{x}_{upper}$ and $\dot{x}_{lower}$. Then the reference generator block is used to determine velocity trajectories $\dot{x}_d(t)$, $\dot{x}_{upper}(t)$ and $\dot{x}_{lower}(t)$ using new $\dot{x}_{dm}$, $\dot{x}_{upper}$ and $\dot{x}_{lower}$. The $\dot{x}_{ave}$, $\dot{x}_{upperave}$ and $\dot{x}_{lowerave}$ are calculated using (11). If $\dot{x}_{lowerave} < \dot{x}_{ave} < \dot{x}_{upperave}$ is not satisfied then the low-level controller is activated to provide assistance to complement participant's effort to complete the task in a precise manner. The Matlab/Simulink/Stateflow software is used to implement the proposed high-level controller (Stateflow/Matlab).



Control Symbol	Constant Value
$\tilde{r}_1$	α=0,β=1
$\tilde{r}_2$	α=1,β=1
$\tilde{r}_3$	α=-1,β=1
$\tilde{r}_4$	β=0
$\tilde{r}_5$	β=0

Fig. 4. Feasible Paths in the High-Level Controller.

6. Results

In this section we present the experimental results on unimpaired participants to demonstrate the efficacy of both the low-level and high-level controller.

6.1. Experiment Procedure

Participants are seated in a height adjusted chair as shown in Fig. 2 (top left). The height of the PUMA 560 robotic manipulator has been adjusted for each participant to start the tracking task in the same arm configuration. The starting arm configuration is selected as shoulder at neutral $0°$ position and elbow at $90°$ flexion position. The task requires moving the arm in forward flexion to approximately $60°$ in conjunction with elbow extension to approximately $0°$. Participants are asked to place their forearm on the hand attachment device as shown in Fig. 2 (bottom left) when the starting arm configuration is fixed. The push button has been given to the participants that can be used during the task execution in case of emergency situations (Fig. 2- bottom middle). The participants receive visual feedback of their position on a computer monitor on top of the desired position trajectory (Fig. 2-top right). Participants were asked to execute the tracking task 50 times.

6.2. Low-Level Controller Evaluation

We had conducted two experiments to evaluate the proposed low-level controller. In the first experiment, the participants were required to perform the tracking task without any external resistance applied to his/her upper arm. Participants were asked to track the position trajectory displayed on the computer screen. The participant's $\dot{x}_{ave}$ was calculated using (11) and if it was in between $\dot{x}_{upperave}$ and $\dot{x}_{lowerave}$ then robot did not need to provide any assistance. However, friction and gravity compensation were always activated in order for the participant to move the robot along with his/her arm in an effortless way. If the $\dot{x}_{ave}$ was not between $\dot{x}_{upperave}$ and $\dot{x}_{lowerave}$, then low-level controller was activated to provide robotic assistance to complement participant's effort to complete the task in a precise manner. During these two experiments, the number of trials and the number of times participant needed robotic assistance were recorded to observe the improvement of participant's movement ability.

In the second experiment, we asked the participant to perform the same task as in Experiment 1; however, in this case, the participant's arm movement ability was constrained with a resistive band (Thera-bands). This was done to simulate the movement of a stroke patient who may experience variable stiffness during the course of motion. In order to apply resistance to participant's upperarm, a mechanism was designed as shown in Fig. 5. Thera-bands are color-coded into many levels of resistance, thus different color resistive bands can be selected in order to simulate different stiffness of the stroke patient's arm. We selected the green (heavy) color resistive band for our experiment, because it provided sufficient resistance to participant's movement while not inhibiting their ability to complete the task. The mechanism has a rod which can slide right or left to change the position of the attachment and can be used for both right-handed and left-handed participants. The rod has holes on it to adjust the location of the resistive band on the upper arm that may vary among participants. The resistive band is connected to the participant's upper arm through a soft strapped attachment to prevent the participant's arm from the irritation that may be caused when the band is stretched. A seat-belt mechanism that connects the rod to the resistive band attachment can be used to release the rod from the resistive band quickly.

Fig. 5a. Initially No Resistance is Applied to the Participant's Upper Arm, Fig. 5b. Resistance is Applied to the Participant's Upper Arm in the Direction of Motion as the Task Begins.

Three female and one male participants within the age range of 25-30 years took part in the experiments that were described in above. All participants were right-handed. In these experiments (i.e., Experiments 1 and 2 as described in above), the participant tried to track the desired position trajectory by visually looking at the computer screen. Each participant performed the task 50 times for each experiment. $\dot{x}$ was selected as *0.02m/s*, which was chosen in consultation with a physical therapist who works with stroke patients at the Vanderbilt Stallworth Rehabilitation Hospital. The $\dot{x}_{upper}$ and $\dot{x}_{lower}$ were selected as *25%* more and less of $\dot{x}$, which were *0.025m/s* and *0.015m/s*, respectively. The range could be increased or decreased based on participant's movement ability. Then, $\dot{x}_d(t)$, $\dot{x}_{upper}(t)$ and $\dot{x}_{lower}(t)$ velocity trajectories were generated using the reference block. The $\dot{x}_{ave}$, $\dot{x}_{upperave}$ and $\dot{x}_{lowerave}$ were calculated using (11) at every *5* seconds. *5* seconds were sufficient to estimate the progress of the participant. If $\dot{x}_{lowerave} < \dot{x}_{ave} < \dot{x}_{upperave}$ was not satisfied then the controller was activated for the next *5* seconds to provide robotic assistance to the participant to track the desired motion within the desired velocity range.

In the first experiment, each participant performed the tracking task without any external resistance applied to his/her upper arm. The idea was to assist the participants as and when they were out of the velocity band. It was noticed that the participants needed less assistance from the robot as they practiced more (Table 2). This result implies that the participants learned how to accomplish the task with practice.

Table 2. Number of Times Robot Assisted for Experiment 1.

Trial Range / Number of Assistance for	1-10	11-20	21-30	31-40	41-50
P1	8	6	5	4	3
P2	14	13	13	12	11
P3	13	11	9	8	7
P4	12	12	11	10	9

Now we present the detailed analysis of the data for one participant (P1) as an example to demonstrate the effectiveness of the low-level controller. This data represented P1's 50th trial. It could be observed from Fig. 6 that the participant's average velocity (stars), which was calculated every *5* seconds using (11), was out of range at A, B and C points. The controller was activated for the next 5 seconds to provide robotic assistance in order to take participant's velocity inside the velocity boundary, thus the controller was active between A-A′, B-B′ and C-C′ (Fig. 6). It could be seen that the participant's velocity was brought inside the desired range at A′, B′ and C′ points, which verified that the assistive ability of the proposed low-level controller.

We further analysed the amount of time taken by the low-level controller (t_s, in seconds) to take $\dot{x}$ into the desired velocity range. Here t_s is defined as the settling time, which is the time taken between the moment the low-level controller was activated and the actual velocity reached the boundary of the desired velocity range. The mean and standard deviation of t_s for all participants' data for Experiment 1 are presented in Table 3.

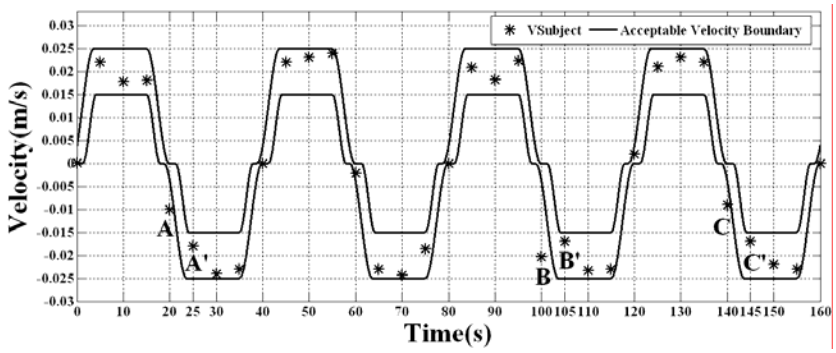


Fig. 6. Calculated Average Velocities for Experiment 1.

Table 3. Settling Time for Experiment 1.

Participant	Mean	Standard Deviation
P1	0.4723	0.1502
P2	0.5801	0.1937
P3	0.4929	0.1272
P4	0.545	0.232

Thus it can be observed from the above set of results that the proposed low-level controller could assist as and when needed and the provided robotic assistance could quickly (i.e., in approximately *0.55* seconds) bring the participant's velocity in the desired range.
In the second experiment, the participant's arm movement ability was constrained with a resistive band as shown in Fig. 5. The participants were asked to track the desired motion by visually looking at the screen as before. It was observed that the participants needed more robotic assistance when their motion was constrained. It could also be observed from Table 4, participants learned how to accomplish the task with practice.

Table 4. Number of Times Robot Assisted for Experiment 2.

Trial Range / Number of Assistance for	1-10	11-20	21-30	31-40	41-50
P1	13	12	12	11	9
P2	16	16	14	13	13
P3	14	13	13	12	12
P4	15	15	14	13	13

We present the mean and standard deviation of the settling time of the low-level controller in Table 5 for all participants' data when they performed Experiment 2. The second experiment was conducted to observe the performance of the controller in an artificially constrained motion scenario, which might provide insight about applying the system to stroke patients whose movement could be naturally constrained. It can be observed that the controller was able to assist as and when needed and could bring the actual velocity of the participant's arm within the desired range in about *0.65* seconds.

Table 5. Settling Time for Experiment 2.

Participant	Mean	Standard Deviation
P1	0.6317	0.232
P2	0.6274	0.2677
P3	0.6438	0.2674
P4	0.6985	0.248

6.3. High-Level Controller Evaluation

In order to demonstrate the efficacy of the proposed high-level controller, we had designed two experiments. In the first experiment, we had demonstrated the efficacy the proposed high-level controller to modify the task when the participant improved his/her movement ability to track the desired trajectory. In the second experiment, we had demonstrated the efficacy of the high-level controller to modify the task in order to ensure the safety of the participants.
In the first experiment, we had used P1's low-level controller results. Δp, was selected as *30*, which could be varied based on participant's progress and the therapist's choice. It was observed from Table 2 that n_{10}=8 and n_{50}=3 and the first criteria in (21) was satisfied, thus E1

was triggered and the plant symbol $\tilde{x}_1$ was generated from the interface difficult ($\tilde{p}_2$) state became active and the control symbol $\tilde{r}_2$ was generated. The interface converted this control symbol to α=*1* and β=*1*. Amount of the increment (*delta*) to increase the difficulty level of the task was an important issue that needed to be decided. In rehabilitation therapies, increasing $\dot{x}_d$ with a small increment would be more desirable especially for low-functioning stroke patients. In this experiment, we had incremented $\dot{x}_d$ by 20%, where $delta = 0.004$. New desired velocity was calculated using (22), which was *0.024m/s*. The velocity boundaries were calculated using (11) as *0.03m/s* and *0.018m/s* for $\dot{x}_{upper}$ and $\dot{x}_{lower}$, respectively. We had asked P1 to perform the tracking task 50 times with this new velocity boundary. Low-level controller provided robotic assistance to the participant as and when they were out of the new velocity band. It was observed that the P1 needed more robotic assistance when the desired velocity to complete the task was increased. It could be seen that P1 learned how to accomplish the task with practice (Table 6).

Table 6. Number of Times Robot Assisted for P1 with New Velocity Boundary.

Trial Range / Number of Assistance for	1-10	11-20	21-30	31-40	41-50
P1	11	10	9	8	7

In the second experiment, we had assumed a safety event had occurred when P1 was performing the task with new increased velocity band. In this experiment, at some point of time during the task P1 wanted to pause for a while and then reset the pause button when she was ready to complete the rest of the task. This scenario might represent when a stroke patient want to pause for a while due to some discomfort. When the task had initially started, E1 was triggered and the plant symbol $\tilde{x}_1$ was generated from the interface. difficult ($\tilde{p}_2$) state became active and the control symbol $\tilde{r}_2$ was given to the interface. The interface converted this control symbol to constant values α=*1* and β=*1*. The plant equation (22) was used to calculate $\dot{x}_{dm}$ (the desired velocity), which was *0.024m/s*. The reaching task required participant to move *0.3m*, thus, the initial position (*0*), desired position (*0.3*) and desired $\dot{x}_{dm}$ (*0.024m/s*) was provided to the reference block to generate the smooth desired velocity trajectory from A to B (Fig.7-left-solid line).

When P1 pressed the pause button at B, E7 was triggered. When E7 was triggered, plant symbol $\tilde{x}_4$ was generated from the interface and stop ($\tilde{p}_4$) state became active. When stop state was active, the high-level controller provided the control symbol $\tilde{r}_4$ and β=*0* was given to (22) and $\dot{x}_{dm}$ was determined as zero. The zero velocity could cause a sudden stop. In order to prevent P1 from suddenly stopping, the reference generator block was used to provide a smooth velocity trajectory to bring the motion to stop. In this case, the velocity was detected when E7 was triggered and the desired velocity was given

as zero and using the reference generator block, the smooth desired velocity was given to the low-level controller from B to C (Fig.7-left-solid line). It could be seen that P1's position (Fig. 7 - right) did not change after the velocity became zero until P1 reset the pause button. SE1 was set to 1 because the state was difficult and E7=1. When the participant reset the pause button, E8 was triggered and $\tilde{x}_5$ plant symbol was given to the interface, and pause ($\tilde{p}_5$) state became active and the high-level controller provided $\tilde{r}_5$. Then $\tilde{x}_6$ was given to the interface because SE=1. The corresponding control symbol $\tilde{r}_2$ was generated, and α=*1* and β=*1* values were given to (22) for calculation of $\dot{x}_{dm}$, which was *0.024m/s*. It could be seen that the high-level controller resumed the task in such a manner that the participant could continue with the therapy with the same task parameters. The participant's position at the time of the triggering of E8 was automatically detected and was given as an initial position to the reference generator block and the desired position was set to *0.3*. The velocity trajectory from C to D was generated and given to the low-level controller (Fig.7-left-solid line). On the other hand, if we did not use this high-level controller, the desired velocity trajectory would not have been automatically modified to register the intention of the participant to pause the task. As a result, the velocity trajectory would have followed the dashed line in Fig. 7-left. In such a case, when P1 wanted to start the task again, the desired velocity trajectory would start at point C′ with non-zero velocity (Fig. 7-left-dashed line). This could create unsafe operating condition. In addition, since the desired velocity computation would not have included the pause action, restarting the task at point C′ would not allow the completion of the task as desired. For example, in this case, if P1 had used the dashed velocity trajectory, she would start moving in the opposite direction at point C′. It could be possible to pre-program all types of desired velocity trajectories beforehand and retrieve them as needed. However, for non-trivial tasks such a mechanism might be too difficult to manage and extend as needed. The presented high-level controller provides a systematic mechanism to tackle such issues. It could also be seen that new velocity trajectories could be created dynamically using the generator block. In order to generate the required trajectories, the task parameters were needed. High-level controller monitored the progress of the task and made decision on the modification of the task parameters.
When the participant reached the desired position, which was *0.3m*, then the velocity trajectory from D to E was generated and given to the low-level controller (Fig.7-left-solid line) so that P1 moved back to the starting position (Fig.7-right).

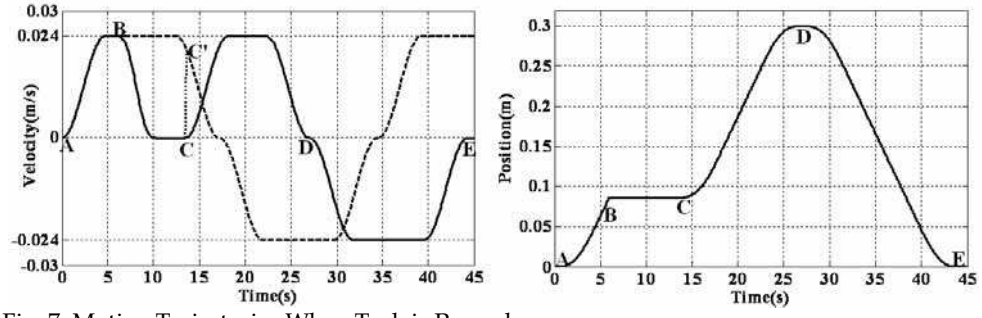


Fig. 7. Motion Trajectories When Task is Paused.

As could be seen from the results, the high-level controller determined the task parameters dynamically based on participant's performance and monitored the safety related events to generate the necessary motion trajectories at the required time.

7. Conclusions and Future Work

In this work we present a new control approach to offer robotic assistance for stroke patients that include the coordination between a high-level controller and a low-level controller. The control architecture presented here is an example of a hybrid control system. There has been no work to our knowledge on designing similar type of control architecture for rehabilitation purposes.

We have initially designed a movement tracking task where the participants not only make repetitive movement but also pay attention to the desired speed of motion from visual feedback. The task was designed in such a manner that it required cognitive processing. Including cognitive processing in the task design is an important criterion because it had been previously shown that the movement tracking task that requires cognitive processing achieved greater gains for brain reorganization of stroke patients than that of movement task that does not require cognitive processing (Carey et al., 2005, Carey et al., 2006).

We have presented a low-level controller to provide robotic assistance to participants to complete the movement tracking task. The high-level controller coordinates with the low-level controller to improve the robotic assistance with the following objectives: 1) to monitor the upper arm rehabilitation task; and ii) to make necessary decisions to address the status of the task. We present a systematic design procedure for the high-level controller to accomplish the above objectives. Note that the proposed high-level controller can be integrated with other low-level controllers with minor modifications. We have conducted experiments with unimpaired participants and demonstrated the usefulness of the high-level and low-level controllers. The results of the use of the low-level controller have demonstrated that the participants needed less assistance from the robot as they practiced more, which implies that the participant's ability to complete the desired motion within a defined velocity range have been improved. Improving the velocity of patient's movement could be an important criterion to measure the success of a rehabilitation therapy. We have also demonstrated that the low-level controller provides assistance to the participants as and when needed and quickly brought the participant's velocity in the desired range. The results of the use of the high-level controller have demonstrated that the task parameters could be determined dynamically based on participant's performance and monitored for safety related events to generate the necessary motion trajectories at the required time. The speed of motion is used as the task parameter in this work. However, the high-level controller can determine other task parameters such as desired reaching position. In some of the rehabilitation tasks, the reaching task is shaped by defining the target position closer to or away from the patient to change the difficulty level of the task. In such a case, for example, the high-level controller can determine the target position based on the participant's progress while monitoring the safety related events.

We are aware that a PUMA 560 robot might not be ideal for rehabilitation applications. However the use of the hand attachment device, which has been described in Section 3, provided a quick release mechanism to protect the participant's arm from injuries. Note that the presented control framework is not specific to the proposed rehabilitation robotic system but can be integrated with any previously proposed rehabilitation robotic system.

An important direction for future development involves testing the usability of the proposed control architecture with stroke patients. Functional magnetic resonance imaging (fMRI) procedure can be used to investigate whether the presented task that included cognitive processing result in long-term brain reorganization. New methods to detect human state information can be integrated into the control architecture such as ECG signals can be used to monitor patients' heart rate to detect their exhaustion and a voice recognition system can be integrated to examine the patient's verbal commands. The proposed control architecture is flexible and extendible in the sense that new events can be included and detected by simply monitoring the additional state information from the human and the robot. In this regard, we are currently working with Vanderbilt University's Stallworth Rehabilitation Hospital to include additional human and robot information.

8. Acknowledgments

We gratefully acknowledge the help of Dr. Thomas E. Groomes who is the Medical Director of Spinal Cord and Traumatic Brain Injury Program, and therapist Sheila Davy of Vanderbilt University's Stallworth Rehabilitation Hospital for their feedback about experimental setup and task design during this work. The work was supported by Vanderbilt University Discovery grant.

9. References

Antsaklis, P.J. & Koutsoukos, X.D. (2003) Hybrid Systems: Review and Recent Progress. In: *Software-Enabled Control: Information Technologies for Dynamical Systems,* T. Samad and G. Balas, (Ed.), IEEE Press, pp. 1-29.

American Heart Association: Heart and Stroke Statistical Update (2006). Available from: http://www.Americanheart.org/statistics/stroke.htm.

Black, J.E.; Isaacs, K.R.; Anderson, B.J.; Alcantara, A.A. & Greenough, W. T. (1990). Learning causes synaptogenesis, whereas motor activity causes angiogenesis, in cerebellar cortex of adult rats. *Proceedings of National Academy of Science*, Vol. 87, No. 15, pp. 5568-5572.

Carey, J.R.; Bhatt, E. & Nagpal, A. (2005). Neuroplasticity Promoted by Task Complexity. *Exercise and Sport Science Review*, Vol. 33, pp. 24-31.

Carey J.; Durfee, W.; Bhatt, E. ; Nagpal, A.; Weinstein, S.;Anderson, K. & Lewis, S. (2006). Tracking vs. Movement Telerehabilitation Training to Change Hand Function and Brain Reorganization in Stroke. *Submitted to Neurorehabilitation and Neural Repair*.

Culmer, P. ; Jackson, A. ; Richardson, R. ; Bhakta, B. ; Levesley, M. & Cozens, A. (2005). An admittance control scheme for a robotic upper-limb stroke rehabilitation system. *International Conference on Engineering in Medicine and Biology Society*, pp. 5081 - 5084.

Erol, D. & Sarkar N. (2007). Design and Implementation of an Assistive Controller for Rehabilitation Robotic Systems. *International Journal of Advanced Robotics Systems*, Vol. 4, No. 3.

Kahn, L.E.; Zygman, M.L; Rymer, W.Z. & Reinkensmeyer, D.J. (2006a). Robot-assisted reaching exercise promotes arm movement recovery in chronic hemiparetic stroke: a randomized controlled pilot study. *Journal of NeuroEngineering and Rehabilitation*, Vol. 3, No.12, pp. 1-13.

Kahn L.E; Lum P.S.; Rymer W.Z. & Reinkensmeyer D.J. (2006b). Robot-assisted movement training for the stroke-impaired arm: Does it matter what the robot does? *Journal of Rehabilitation Research & Development*, Vo. 43, No. 5, pp. 619-630.

Kleim, J. A.; Barbay, S. & Cooper, N.R. (2002). Motor learning-dependent synaptogenesis is localized to functionally reorganized motor cortex. *Neurobiology Learning and Memory*, Vol. 77, pp. 63-77.

Koutsoukos, X.D.; Antsaklis, P.J.; Stiver, J.A. & Lemmon, M.D. (2000). Supervisory control of hybrid systems. *Proceedings of the IEEE on Special Issue on Hybrid Systems: Theory and Applications*, Vol. 88, No. 7, pp. 1026 – 1049.

Krebs, H.I.; Palazzolo, J.J.; Dipietro, L.; Ferraro, M.; Krol J.; Rannekleiv, K.; Volpe, B.T. & Hogan N. (2003). Rehabilitation Robotics: Performance-Based Progressive Robot-Assisted Therapy. *Autonomous Robots*, Vol. 15, No. 1, pp. 7-20.

Krebs, H. I.; Ferraro, M., Buerger, S.P., Newbery, M. J., Makiyama, A., Sandmann, M.; Lynch, D.; Volpe, B. T. & Hogan, N. (2004). Rehabilitation robotics: pilot trial of a spatial extension for MIT-Manus. *Journal of NeuroEngineering and Rehabilitation*, Vol. 1, No. 5, pp. 1-15.

Loureiro, R.; Amirabdollahian, F.; Topping, M.; Driessen, B. & Harwin, W. (2003). Upper limb mediated stroke therapy - GENTLE/s approach. *Autonomous Robots*, Vol.15, pp. 35-51.

Lum, P.S.; Burgar, C.G.; Van der Loos, H.F.M.; Shor, P.C.; Majmundar M. & Yap R. (2006). MIME robotic device for upper-limb neurorehabilitation in subacute stroke subjects: A follow-up study. *Journal of Rehabilitation Research & Development*, Vol. 43, No. 5, pp. 631-642.

Mallapragada, V.; Erol, D. & Sarkar, N. (2006). A New Method for Force Control for Unknown Environments. *Proceedings of the International Conference On Intelligent Robots and Systems*, pp. 4509 - 4514.

Matchar, D.B. & Duncan, P.W. (1994). Cost of stroke, *Stroke Clinical Updates*, Vol. 5, pp. 9-12.

Nudo, R.; Milliken, G.; Jenkins, W. & Merzenich M. (1996). Use-dependent alterations of movement representations in primary motor cortex of adult squirrel monkeys. *The Journal of Neuroscience*, Vol. 16, No. 2, pp. 785-807.

Pascual-Leone, A.; Nguyen, K.T.; Kohen, A.D.; Brasil-Neto, J.; Cammarota, A. & Hallett M. (1995). Modulation of muscle responses evoked by transcranial magnetic stimulation during the acquisition of new fine motor skills. *Journal of Neurophysiology*, Vol. 74, pp. 1037-1045.

Plautz, E.J.; Milliken, G. W. & Nudo, R J. (2000). Effects of repetitive motor training on movement representations in adult squirrel monkeys: role of use versus learning. *Neurobiology of Learning and Memory*, Vol. 74, pp. 27-55.

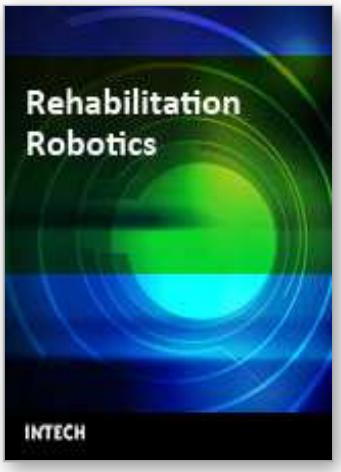
PUMA 560 Related Sites on the Internet, Available from: www.ee.ualberta.ca/~jasmith/puma/pumasites.html.

Sciavicco, L. & Siciliano, B. (1996). *Modeling and Control of Robot Manipulators*, McGrawHill, ISBN-1852332212, Great Britain.

Stateflow, Mathworks Inc, http://www.mathworks.com/products/stateflow/?BB=1

Taub, E.; Uswatte, G. & Pidikiti, R. (1999). Constraint-Induced Movement Therapy: A New family of techniques with broad application to physical rehabilitation – a clinical review. *Journal of Rehabilitation Research and Development,* Vol. 36, pp. 237-251.

Taub, E.; Lum, P.S.; Hardin, P.; Mark V.W. & Uswatte, G. (2005). AutoCITE: Automated Delivery of CI Therapy With Reduced Effort by Therapists. *Stroke*, Vol. 36, pp. 1301-1304.



Rehabilitation Robotics

Edited by Sashi S Kommu

ISBN 978-3-902613-04-2

Hard cover, 648 pages

Publisher I-Tech Education and Publishing

Published online 01, August, 2007

Published in print edition August, 2007

The coupling of several areas of the medical field with recent advances in robotic systems has seen a paradigm shift in our approach to selected sectors of medical care, especially over the last decade. Rehabilitation medicine is one such area. The development of advanced robotic systems has ushered with it an exponential number of trials and experiments aimed at optimising restoration of quality of life to those who are physically debilitated. Despite these developments, there remains a paucity in the presentation of these advances in the form of a comprehensive tool. This book was written to present the most recent advances in rehabilitation robotics known to date from the perspective of some of the leading experts in the field and presents an interesting array of developments put into 33 comprehensive chapters. The chapters are presented in a way that the reader will get a seamless impression of the current concepts of optimal modes of both experimental and ap- plicable roles of robotic devices.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Duygun Erol and Nilanjan Sarkar (2007). Design and Implementation of a Control Architecture for Rehabilitation Robotic Systems, Rehabilitation Robotics, Sashi S Kommu (Ed.), ISBN: 978-3-902613-04-2, InTech, Available from: http://www.intechopen.com/books/rehabilitation_robotics/design_and_implementation_of_a_control_architecture_for_rehabilitation_robotic_systems

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821