Motion Control Strategies based on PD Control for a Four Degree-of-Freedom Serial Robotic Manipulator to Mimic Human Index Finger Articulations

Abhijit Mahapatra
CSIR-CMERI,
Durgapur, India.

Kaustav Biswas
NIT Durgapur
Durgapur, India.

Amit Kumar
CSIR-CMERI,
Durgapur, India.

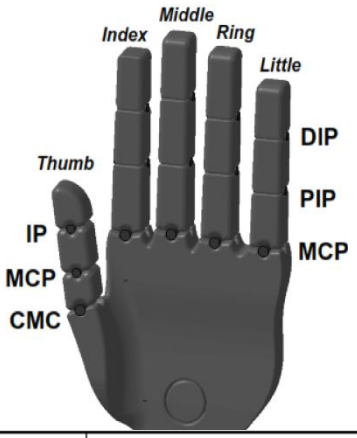
Avik Chatterjee
CSIR-CMERI,
Durgapur, India.

ABSTRACT
The present study is based on motion control of an articulated model of a human index finger that has been modeled as a four-degree-of-freedom serial robotic manipulator. Preliminary studies on position and tracking control have been carried out by testing various control strategies based on proportional-derivative (PD) control in a simulation environment wherein the manipulator has been modeled and simulated with a certain input signal and responses to various controllers have been shown. Model free and model based controllers have been designed simulated using MATLAB®/ Simulink®. Strategies like model free PD control have been improved upon by introducing auto-tuning and learning capabilities. A virtual plant modeled using Simmechanics® toolbox has been set up and used for simulation purposes.

General Terms
Degree of freedom, human hand articulations, serial manipulators

Keywords
PD Control, Feed-forward Control, Computed Torque Control, Human Index Finger, Simmechanics

1. INTRODUCTION
Since the beginning of humanity's quest with robotics and artificial intelligence, robotic manipulators have been widely studied, developed and implemented in various fields. Combined with structural diversity, the different types of control systems developed over time have enabled the use of manipulators for a wide variety of tasks ranging from industrial to domestic. Recently, robotic manipulators are being tested in the field of rehabilitation of patients who have partially lost the functionality of limb(s). The human hand is a highly articulated manipulator with a large number of degrees of freedom. Mimicking the functionality of the human hand using a machine is an interesting problem of robotic and control systems research. In the present study, an attempt has been made to develop an articulated hand model and subsequently implement various motion control algorithms to mimic actions like palmer grasping (Fig. 1). The methods developed can be used to construct a functional wearable exoskeleton for the human hand or a remotely controlled robotic hand which may communicate with this exoskeleton, worn by a human operator, facilitating its use in a variety of tasks like remote manipulation, physiotherapy etc., thus broadening the avenue of Human Computer Interaction.



Joints	Fingers					Total
	Thumb	Index	Middle	Ring	Little	
Carpometacarpal (CMC)	2					20 DOF
Metacarpophalangeal (MCP)	1	2	2	2	2	
Interphalangeal (IP)	1					
Proximalinter-phalangeal (PIP)		1	1	1	1	
Distalinterphalangeal (DIP)		1	1	1	1	

Fig. 1: CAD model of the human hand showing various joints and associated degrees of freedom

The study has been aided by important concepts presented in [1], [2] and [3]. Modeling of model free PD controllers has been very well explained in [4] and [5]. Innovative control architectures have been presented in [6]. Studies on model based control techniques like feed-forward (FFD) control and computed torque (CTC) control have been presented in [7], [8] and [9]. Further, we have referred to [10], [11] and [12] for motivation. Commercially available products similar to the one in this study are the Festo® 'Exohand', the 'Shadow Dexterous Hand™' etc.

The present paper has been categorized into five sections. Section 2 describes the mechanical and mathematical model setup for the problem using the Lagrange-Euler formulation for modeling dynamics of the manipulator. Section 3 presents the development of various controller strategies for controlling a single finger modeled as a four degree of freedom manipulator, with some controllers using the dynamic model developed in the previous section. Section 4 presents modeling and simulation of the finger manipulator, which show the response of various control systems to a specified input signal. Section 5 concludes the paper.

2. MODEL OF THE MANIPULATOR

2.1 Mathematical Model

The inverse dynamic formulation according to Euler-Lagrange formulation for a general *n*-link manipulator is given by the following equation, as described in [1]:

$$\mathbf{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \mathbf{H}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \mathbf{G}(\boldsymbol{q}) + \mathbf{F}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{\tau} \quad (1)$$

For the present case, ***'q'*** is the generalized co-ordinate, here, the joint angle; ***'M'*** is the 4×4 matrix of link inertias; ***'H'*** is the matrix of Coriolis/centripetal forces and ***'G'*** represents the gravity vector. ***'F'*** is the matrix of joint stiffness and damping terms. ***'τ'*** may represent the vector of generalized forces or torques applied by an actuator.

$$\boldsymbol{q} = [q_1 \;\; q_2 \;\; q_3 \;\; q_4]^T \quad (2)$$

$$\boldsymbol{\tau} = [\tau_1 \;\; \tau_2 \;\; \tau_3 \;\; \tau_4]^T \quad (3)$$

$$\mathbf{M} = \begin{bmatrix} M_{11}(q) & M_{12}(q) & M_{13}(q) & M_{14}(q) \\ M_{21}(q) & M_{22}(q) & M_{23}(q) & M_{24}(q) \\ M_{31}(q) & M_{32}(q) & M_{33}(q) & M_{34}(q) \\ M_{41}(q) & M_{42}(q) & M_{43}(q) & M_{44}(q) \end{bmatrix} \quad (4)$$

$$\mathbf{H} = [H_1(\boldsymbol{q},\dot{\boldsymbol{q}}) \;\; H_2(\boldsymbol{q},\dot{\boldsymbol{q}}) \;\; H_3(\boldsymbol{q},\dot{\boldsymbol{q}}) \;\; H_4(\boldsymbol{q},\dot{\boldsymbol{q}})]^T \quad (5)$$

$$\mathbf{G} = [G_1(\boldsymbol{q}) \;\; G_2(\boldsymbol{q}) \;\; G_3(\boldsymbol{q}) \;\; G_4(\boldsymbol{q})]^T \quad (6)$$

$$\mathbf{F} = [F_1(\boldsymbol{q},\dot{\boldsymbol{q}}) \;\; F_2(\boldsymbol{q},\dot{\boldsymbol{q}}) \;\; F_3(\boldsymbol{q},\dot{\boldsymbol{q}}) \;\; F_4(\boldsymbol{q},\dot{\boldsymbol{q}})]^T \quad (7)$$

The elements of the '**M**' matrix are of the form:

$$M_{ik} = \sum_{j=\max(i,k)}^{4} Tr\left[\mathbf{d}_{jk}\mathbf{I}_j\mathbf{d}_{ji}^T\right] \quad (8)$$

where, $\mathbf{I}_j$ represents the Identity matrix of order 4x4, and i,k = 1,2,3,4. '*Tr*' represents the trace of the matrix.

Also,

$$\mathbf{d}_{ij} = \frac{\partial\left({}^0\mathbf{T}_i\right)}{\partial\theta_j} = \begin{cases} {}^0\mathbf{T}_{j-1}\mathbf{Q}_j\,{}^{j-1}\mathbf{T}_i & j \le i \\ 0 & j > i \end{cases}$$

and $\mathbf{Q}_j$=$\mathbf{Q}_k$=[0 -1 0 0;1 0 0 0;0 0 0 0;0 0 0 0] for a rotary joint

The elements of the '**H**' matrix are of the form:

$$H_i = \sum_{k=1}^{4}\sum_{m=1}^{4} h_{ikm}\dot{q}_k\dot{q}_m \quad (9)$$

where

$$h_{ikm} = \sum_{j=\max(i,k,m)}^{4} Tr\left[\mathbf{d}_{jkm}\mathbf{I}_j\mathbf{d}_{ji}^T\right], \; i, k, m = 1, 2, 3, 4$$

$$\mathbf{d}_{ijk} = \frac{\partial\mathbf{d}_{ij}}{\partial q_k} = \begin{cases} {}^0\mathbf{T}_{j-1}\mathbf{Q}_j\,{}^{j-1}\mathbf{T}_{k-1}\mathbf{Q}_k\,{}^{k-1}\mathbf{T}_i & i \ge k \ge j \\ {}^0\mathbf{T}_{k-1}\mathbf{Q}_k\,{}^{k-1}\mathbf{T}_{j-1}\mathbf{Q}_j\,{}^{j-1}\mathbf{T}_i & i \ge j \ge k \\ 0 & i < j, \quad i < k \end{cases}$$

where the '${}^{a}\mathrm{T}_{b}$' terms represent transformation matrices as obtained from Denavit-Hartenberg formulation as described in [1, 2]. The terms of the 'G' matrix are of the form:

$$G_i = \sum_{j=i}^{4}\left[-m_j\mathbf{g}^T.\mathbf{d}_{ji}\,{}^j\bar{\mathbf{r}}_j\right] \text{ where } i = 1 \text{ to } 4 \quad (10)$$

'm_j' represents the mass of the link *j*, 'r_j' represents its radius in case of a cylindrical link and '*g*' represents acceleration due to gravity = 9.81 m/s^2.

Thus the expression for torque applied by actuator at joint p (neglecting joint stiffness and damping) is obtained by substituting values of various terms in equation (1):

$$\tau_j = \sum_{j=i}^{4}\sum_{k=1}^{4} Tr\left[\mathbf{d}_{jk}\mathbf{I}_j\mathbf{d}_{ji}^T\right]\ddot{q}_k + \sum_{j=i}^{4}\sum_{k=1}^{4}\sum_{m=1}^{4} Tr\left[\mathbf{d}_{jkm}\mathbf{I}_j\mathbf{d}_{ji}^T\right]\dot{q}_k\dot{q}_m + \sum_{j=i}^{4}\left[-m_j\mathbf{g}^T.\mathbf{d}_{ji}\,{}^j\bar{\mathbf{r}}_j\right] \quad (11)$$

If we include joint stiffness and damping terms as terms of the 'F' matrix then,

$$F_i(q,\dot{q}) = d_i q_i + c_i\dot{q}_i \quad (12)$$

where 'd_i' and 'c_i' terms are stiffness and damping constants and the '**F**' matrix is added to the overall torque matrix obtained above. Generally, it is difficult to determine the exact nature of the terms of this matrix. Here it has been assumed that $c_i = q_i^2$ and $d_i = 0.25\, c_i$.

2.2 Mechanical Model

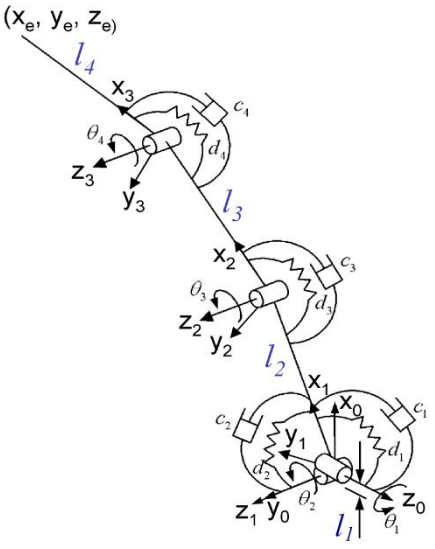
The virtual plant model for simulation that has been created in the Simmechanics® environment of Simulink® and shown in Section 4 has been constructed using the following inertia tensor for each link.

$$\mathbf{I}_j = [(-{}^jI_{xx}+{}^jI_{yy}+{}^jI_{zz})/2 \;\; {}^jI_{xy} \;\; {}^jI_{xz} \;\; m_j\bar{x}_j \;;\; {}^jI_{xy} \;\; ({}^jI_{xx}-{}^jI_{yy}+{}^jI_{zz})/2 \;\; {}^jI_{yz} \;\; m_j\bar{y}_j \;;\; {}^jI_{xz} \;\; {}^jI_{yz} \;\; ({}^jI_{xx}+{}^jI_{yy}-{}^jI_{zz})/2 \;\; m_j\bar{z}_j \;;\; m_j\bar{x}_j \;\; m_j\bar{y}_j \;\; m_j\bar{z}_j \;\; m_j] \quad (13)$$

where the ${}^jI_{xy}$, ${}^jI_{yz}$, ${}^jI_{xz}$ terms represent cross products of inertia and ${}^jI_{xx}$, ${}^jI_{yy}$, ${}^jI_{zz}$ terms represent moment of inertia with respect to the x, y and z axis respectively; m_j is the mass of the link *j*. In a right handed coordinate system {x y z}, $I_{xx}=\int(y^2+z^2)dm$; $I_{xy}=\int xy\, dm$ and so on.

A simplified model taking into account stiffness and damping effects at each joint has been shown in Fig. 2(a) and free body diagrams for the links have been shown in Fig. 2(b). Here the joint angles have been represented by 'θ_i' terms and actuator torques as 'τ_{Mi}' terms.

(a)



(b)

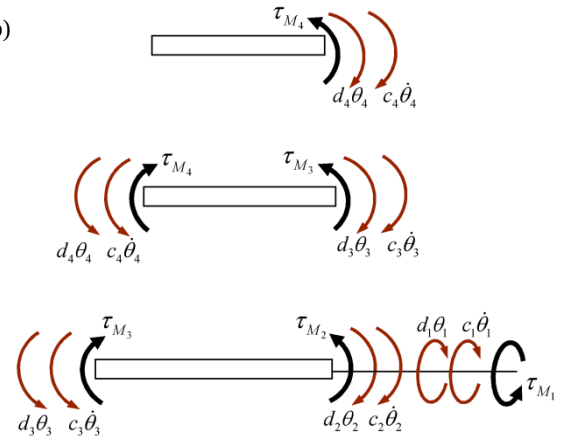


Fig.2: (a) Simplified physical model showing each link as a line and joints as cylinders with stiffness and damping terms; (b) Free body diagrams of all the links with actuator torques indicated

3. CONTROL SYSTEM FOR THE MANIPULATOR

Various effective control strategies for robotic manipulators have been reported. These include the conventional PID based linear controllers described in [4-5], FFD controllers, CTC controllers described in [7-9, 11-12], robust, adaptive, sliding mode control etc. With PD controllers at each joint, global asymptotic stability about a given joint configuration can be achieved. Present investigation deals with performance of PD based controllers to control a single manipulator representing the human index finger with four revolute joints as shown in Fig. 2. Here, independent joint control scheme has been implemented in several different ways in an attempt to improve performance and include effects of model nonlinearities and dynamics as FFD terms. This also helps us linearize the system and decouple the link dynamics. Multiple PD control loops are set up to treat this Multi input Multi Output (MIMO) system as a set of four Single Input Single Output (SISO) systems. Robotic manipulators used for motion control applications can be of 'point to point' type or 'continuous path' type as described in [1]. The former requires position control, while in the latter requires tracking control. These terms are explained (as defined in [6]):

Position control: Given a desired joint configuration qd, find a control law such that the manipulator state $[\,q^T\ \dot{q}^T\,]^T$, converges to $[\,q_d^T\ \ 0^T]^T$. Thus the manipulator end point arrives at the desired location with the desired joint configuration and zero velocity.

Tracking control: Given a bounded desired position trajectory $q_d(t)$, which is twice continuously differentiable and has bounded first and second derivatives, find a control law such that the manipulator state $[\,q^T(t)\ \dot{q}^T(t)\,]^T$, converges to $[\,q_d^T(t)\ \dot{q}_d^T(t)\,]^T$ for any initial condition. Thus the manipulator configuration must always match with the desired configuration under the influence of a tracking control law.

Present application demands the development of a tracking control algorithm due to the varying nature of configurations this articulated manipulator may need to achieve at different times. Ideal tracking could be achieved if all dynamic effects of the plant could be compensated for by a controller. But it is never possible to obtain a fully accurate dynamic model of the plant or know about the nature of disturbances that the plant might face during operation. Only an approximate description of the plant can be developed using prior knowledge of the plant's mechanics and models can be proposed for describing effects such as stiffness and damping. This calls for inclusion of FFD loops with additional controllers such as PD or PID controllers along with the compensating terms. However, acceptable tracking performance can be achieved without using a dynamic model incorporated into the controller, by suitably choosing a feedback structure and carefully tuning the controller. Both these approaches have been used for designing the following controllers.

3.1 PD Controller

Proportional and derivative control law requires angular position and velocity feedback. Here a FFD term containing model stiffness and damping terms to compensate for the effects of those factors, has been included.

The PD control can be described as:

$$\tau = K_p e + K_D \dot{e} + F(q_d, \dot{q}_d) \qquad (14)$$

where K_p>0 and K_D>0 are 4x4 diagonal matrices of controller gains, 'e' and '$\dot{e}$' represent the joint angle and velocity errors respectively. $F(q_d, \dot{q}_d)$ is a feedforward term with 'q_d' and '$\dot{q}_d$' representing desired joint angles and velocities as in equation(12). 'τ' represents the actual control torque applied to the plant (manipulator). The structure of this controller is as shown in Fig.3.

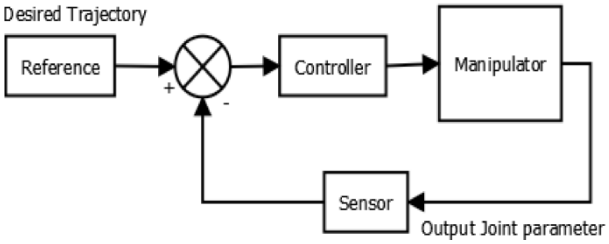


Fig. 3: General scheme for PD/PDNL controller showing position of each major component in the scheme

An improvement over the previous control strategy can be achieved by applying varying control efforts according to the magnitude of the error as shown in [4, 5]. This can be achieved by automatically tuning the PD gains by making them nonlinear functions of the error. Such a control law (PDNL) can be stated as:

$$\tau = K_p(t)e(t) + K_D(t)\dot{e}(t) + F(q_d, \dot{q}_d) \qquad (15)$$

where '$Kp(t)$' and '$K_D(t)$' represent the time varying diagonal gain matrices and 'e(t)' represents current value of error.

Also, $K_p(t) = K_{p0} * K^p(t)$ (16)

and $K_D(t) = K_{d0} * K^d(t)$ (17)

where 'K_{p0}' and 'K_{d0}' are initial diagonal gain matrices.

It is desired that, when the error is high the gains should be large to achieve quicker convergence and also avoid overshoot. Similarly when the error is low it is desired to keep the gains low so that possibilities of overshoot or oscillations are less. Also the control effort must not be too large in the presence of large unexpected errors, which may cause actuator saturation or damage to the plant. Thus saturation of control effort after a certain limit is required.

A function satisfying these conditions is the negative hyperbolic secant function (Fig.4). The gains can be tuned online by making them nonlinear functions of the error variable.

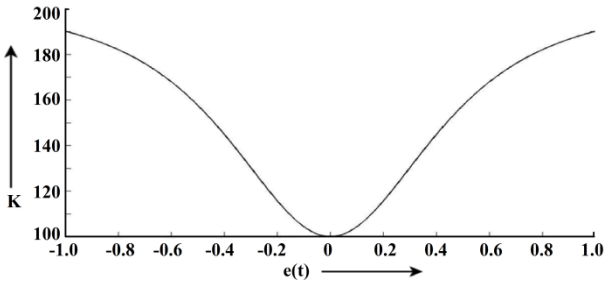


Fig. 4: Variation of gain with error for NPD controller (a particular case)

Let gain, $K=K_{max}–K_{min}*sech(a*e(t))$ (18)

where *'K_{max}'* , *'K_{min}'* and *'a'* are user defined positive constants and *'e(t)'* represents current error The generalized Proportional and Derivative gains can thus be expressed as:

$$K^{p_i}(t) = K_{\max}^{p_i} - K_{\min}^{p_i} sec\, h\left(a_i . e_i(t)\right) \quad (19)$$

$$K^{d_i}(t) = K_{\max}^{d_i} - K_{\min}^{d_i} sec\, h\left(a_i . e_i(t)\right) \quad (20)$$

where i = 1,2,3,4

3.2 PD + Iterative Learning Control

As the dynamic model of the manipulator has not been incorporated in the controllers described above, the controller might find it difficult to make the system track inputs effectively, due to lack of knowledge of the system's probable behavior which is governed by the dynamics. Some knowledge of the dynamics can be included by incorporating an FFD term to the control law as shown in [4, 5]. An iterative learning algorithm feeds forward the control torque obtained from the previous iteration such that the controller can 'learn' about the system's dynamics from history of it's 'behavior'. A controller can thus be obtained, that has feedback PD terms and FFD learning term for each controlled joint. The PD Learning Controller (PDLC) can be expressed as:

$$\tau = K_p e + K_D \dot{e} + F(q_d, \dot{q}_d) + z(q, \dot{q}, \ddot{q}) \quad (21)$$

where the structure obtained in (14) has been retained and last term 'z' has been added to account for torque from the previous iteration. Thus the law can be simplified (for the i^{th} iteration) as:

$$\tau_i = K_p e_i + K_D \dot{e}_i + \tau_{i-1} \quad (22)$$

Fig.5 shows a schematic of the PD+ILC law:

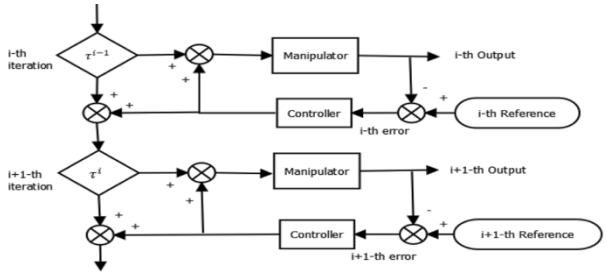


Fig. 5: The PD+ILC scheme showing only two generations or iterations

During the first iteration the FFD term zero and the control is simple PD control. This controller also insures good tracking performance without detailed knowledge about the plant dynamics which gives this type of controller a distinct advantage over CTC type of controllers with regard to computational ease.

A better version of this controller can be obtained by combining learning control with a PD controller having nonlinear gains as developed in equation(15). The control law for a PDNLLC controller can thus be written as:

$$\tau = K_p e(t) + K_D \dot{e}(t) + F(q_d, \dot{q}_d) + z(q, \dot{q}, \ddot{q}) \quad (23)$$

where the terms carry meanings as defined in previous sections.

3.3 Model based Controllers

Next, an attempt to design controllers that directly consider a dynamic model of the actual system to be available for computation, is considered. A very popular class of controllers for robotic manipulators is the CTC controller which is a special case of feedback linearization of nonlinear systems. Here a nonlinear inner loop and a linear (PD) outer loop can effectively decouple and linearize the system as stated in [1].

This controller is also known as 'inverse dynamics' controller [2] because of its reliance on an inverse dynamic model for the plant described in equation(1). The accuracy of this controller in solving the tracking problem also depends upon how accurate the dynamic model of the manipulator is. Thus this method is computationally intensive and not suitable for cases where an accurate description of a system's dynamics is absent. A FFD control law can also be stated, that does not use the online data from outputs, and rather it uses the reference data to compute an approximate plant model for the controller. These control schemes are discussed as follows:

3.3.1 PD + FFD control

A FFD controller gives prior knowledge to the controller regarding dynamics of the plant. The FFD data is provided by an approximate model of the plant dynamics and thus the accuracy of the controller depends on the accuracy of the dynamic model. Feeding forward dynamic data decouples the link dynamics to achieve better control of the nonlinear system. By including a feedback loop employing classical PD controller, linearization can be achieved. Thus by having a PD outer loop and a FFD signal based on the desired trajectory, better tracking can be obtained, while dealing with complicated dynamics such as the present case. Thus the PD + FFD control law can be described as [2, 7-9]:

$$\tau = K_p e + K_D \dot{e} + M(q_d)\ddot{q}_d + H(q_d, \dot{q}_d) + G(q_d) + F(q_d, \dot{q}_d) \quad (24)$$

where the first two terms represents the P and D control terms and the next four terms represent the FFD terms as functions of desired/reference terms indicated with subscript 'd'. These have been modeled according to equations (1)-(12). The FFD scheme is shown in Fig.6.

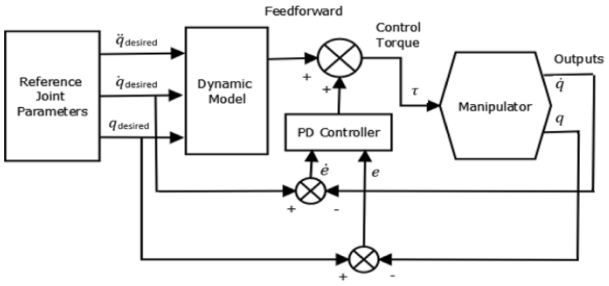


Fig. 6: PD + FFD control scheme

3.3.2 CTC Control

The principal idea behind CTC control is to arrive at a linear control law using the dynamics of a nonlinear system in the FFD and using an outer linear control loop in the feedback path like a PD loop as described in [2, 7, 8]. The dynamic model's terms are functions of the plant outputs and thus are updated online. The control law can be stated as:

$$\tau = M(q)\tau' + H(q,\dot{q}) + G(q) + F(q,\dot{q}) \quad (25)$$

where $\tau' = K_p e + K_D \dot{e} + \ddot{q}_d$ (26)

Thus a linear control law can be obtained, such as,

$$\ddot{q} = \tau', \quad (27)$$

Equation (26) represents the output of the outer PD loop control law and equation (25) gives the complete control law for this controller. The dynamic model's terms are outputs of the plant being controlled. The CTC control scheme is shown in Fig.7.

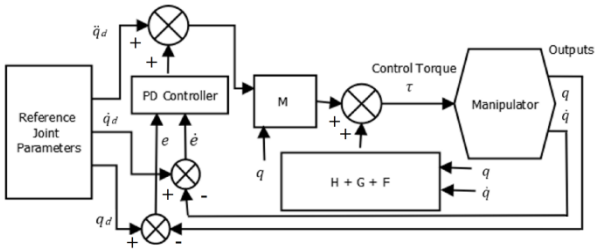


Figure 7: Computed Torque control scheme

4. SIMULATION STUDY AND RESULTS

As stated earlier, the designed control systems have been tested on a model of only one 'finger' having four revolute joints. All other fingers can be designed in a similar manner. Simulation has been carried out in MATLAB® using the Control Systems Toolbox® and the Simmechanics® toolbox [10, 13] of Simulink®. A virtual physical model of the plant has been constructed using the inertia tensor of each link and estimated masses and radii for the links. Every joint has been actuated with the control torques for each controller. Joint sensor blocks measure the angular positions and velocities for providing the feedback. 'MATLAB Function' blocks have been used to set up the PD controllers and dynamic FFD terms. All controllers have been tuned separately for best performance, for which this is not a true comparison study, though some expected results are obtained. The manipulator as modeled in Simmechanics® is shown in Fig.8. The model in Simmechanics® show the links, base link (in black) and others (in brown), joint actuator, joint sensor, joint initial condition blocks and other blocks to set environment effects like ground/ reference and gravity. Link parameters are as per Table 1.

Table 1. Link Parameters

Link	Mass (kg)	Length (m)	Radius (m)
1	68.86e-6	0.002	0.002
2	38.73e-3	0.045	0.01
3	19.79e-3	0.023	0.01
4	17.21e-3	0.020	0.01

In accordance with 3D right handed co-ordinate system {x y z}, gravity has been taken along the negative 'y' direction. Henceforth the various joints of the index finger, the MCP (2 DOF), the PIP and the DIP have been referred to as joints 1, 2, 3 and 4 respectively. While joint 1 revolves about the 'y' axis, all other joints revolve about the 'z' axis.

The input signal i.e., the reference joint angle signal for each joint is:

$$q_d = 1+0.5\sin(\pi t) \quad (28)$$

The joint angular velocity and acceleration data wherever required, are generated by differentiation of the above signal. Fig.9 shows a screenshot of the complete Simulink model of only one of the controllers, the PD plus Learning controller. The controller was designed in Simulink® where the subsystem (in blue) is the model shown in Fig.8 and the controller that has been developed using 'MATLAB function' blocks, is shown by another subsystem. The reference inputs are introduced as time series variables. Here 'Memory' blocks feed in the value of the signal during the previous iteration. We also introduce a FFD term for joint stiffness and damping effects by the 'Disturbance' subsystem. The 'Theta' blocks inject reference joint angle 'q_d' signals and they are differentiated to obtain the 'Omega' signals or reference joint angular velocity '$\dot{q}$' signals.

The PD and PDLC controllers have been set up with the following gains:

K_p = diag. {800, 1350, 2800, 2800} and K_D = diag. {15, 18, 10, 10}

The PDNL and PDNLLC controllers have been set up with the following:

$K_{\max}^{p_1} = 200;\ K_{\max}^{p_2} = 150;\ K_{\max}^{p_3} = 200;\ K_{\max}^{p_4} = 200;$

$K_{\min}^{p_1} = 50;\ K_{\min}^{p_2} = 100;\ K_{\min}^{p_3} = 500;\ K_{\min}^{p_4} = 500;$

$K_{p_{10}} = 1000;\ K_{p_{20}} = 1500;\ K_{p_{30}} = 3000;\ K_{p_{40}} = 3000.\ K_{\max}^{d_1} = 5;\ K_{\max}^{d_2} = 2;\ K_{\max}^{d_3} = 20;\ K_{\max}^{d_4} = 20;$

$K_{\min}^{d_1} = 2;\ K_{\min}^{d_2} = 1;\ K_{\min}^{d_3} = 5;\ K_{\min}^{d_4} = 5;$

$K_{d_{10}} = 20;\ K_{d_{20}} = 20;\ K_{d_{30}} = 30;\ K_{d_{40}} = 30;$

$a_1 = a_2 = a_3 = a_4 = 1.$

For the FFD and the CTC controllers we have: K_p = diag. {650, 900, 1100, 1100} and K_D = diag. {10, 15, 8, 8}

The following figures show the (angular position) tracking performance for the controllers while tracking the signal as shown in equation(28). Fig.10, shows the performance of the PD, PDNL, PDLC and PDNLLC controllers. The next four figures, Fig.11, show the performance of the CTC and FFD controllers. All angles are in radians.

Fig. 12 shows snapshots of animation during the simulation from different views. The joint initial condition dictates the manipulator to maintain a straight configuration at t=0. After application of the control torque for 0.5 seconds, from one of the controllers we get the shown configuration, somewhat similar, to what we expect during the act of palmer grasping. The results obtained in Fig.10, indicate a superiority of the learning controllers over the non-learning types and of the 'online tunable' controllers over the fixed gain types. Armed with both tunable gains and learning capability, the PDNLLC controller is found to achieve faster convergence compared to the simple PD controller that lacks both these capabilities. As expected, the PDLC and PDNL controllers' performance lie in between the other two. In case of the model based controllers, Fig. 11, show that the CTC controller performs better than the FFD controller for the all the joints. That both these controllers show appreciable tracking performance, indicates the dynamic model developed for them is accurate enough for this application.

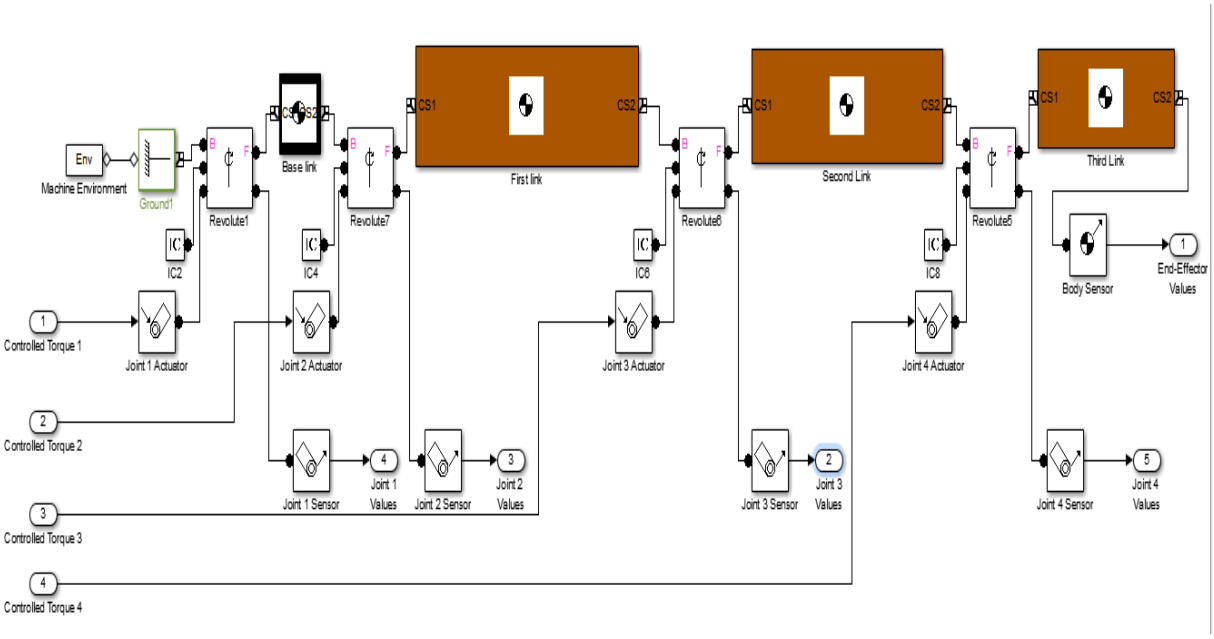


Fig. 8: Manipulator model in Simmechanics®

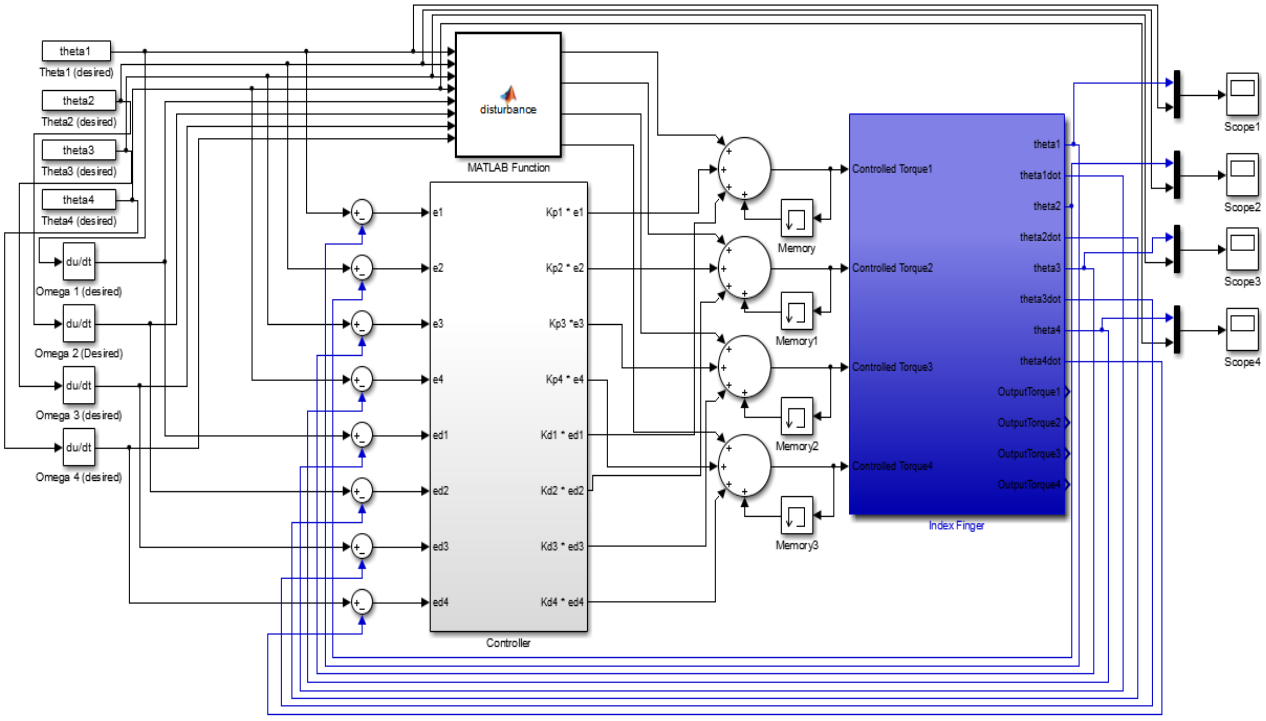


Fig. 9: PD plus Learning controller designed in Simulink®

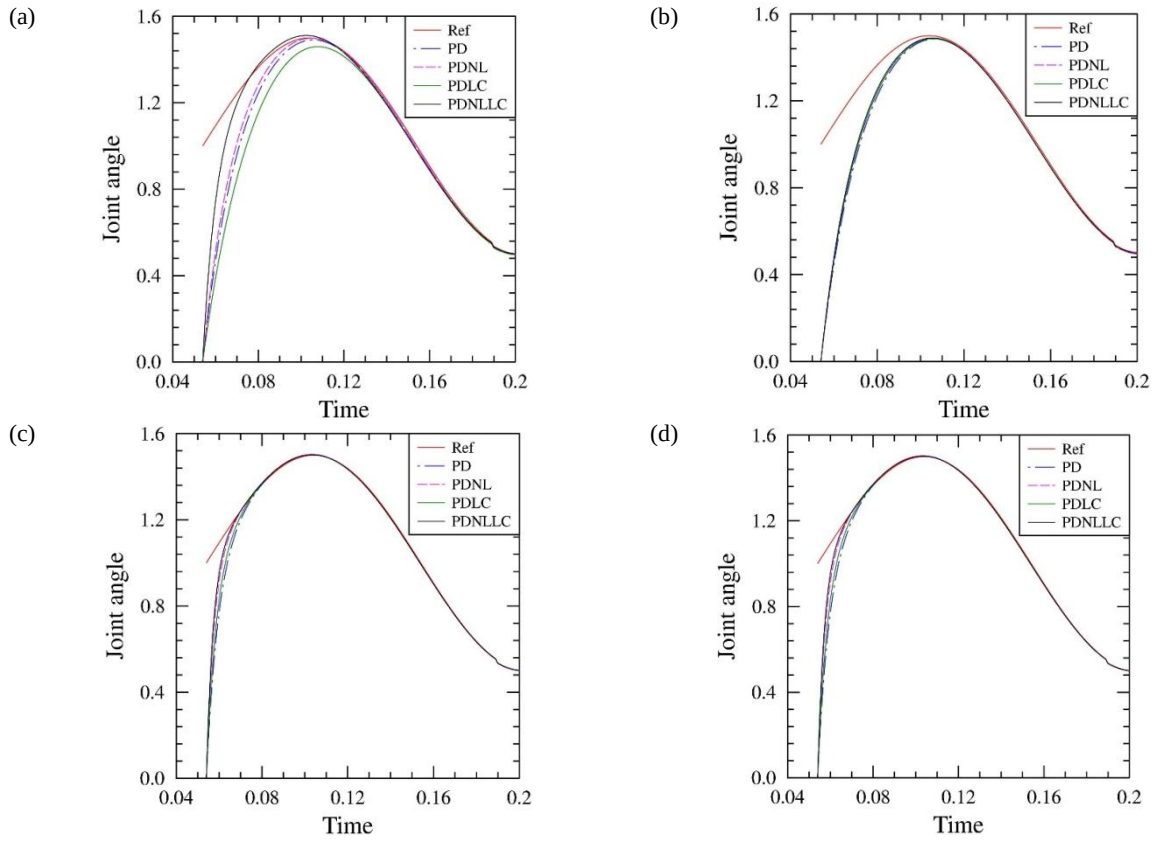


Fig. 10. Tracking performance of Model free controllers: PD, PDNL, PDLC and PDNLLC controllers. (a) joint 1, (b) joint 2, (c) joint 3, (d) joint 4.

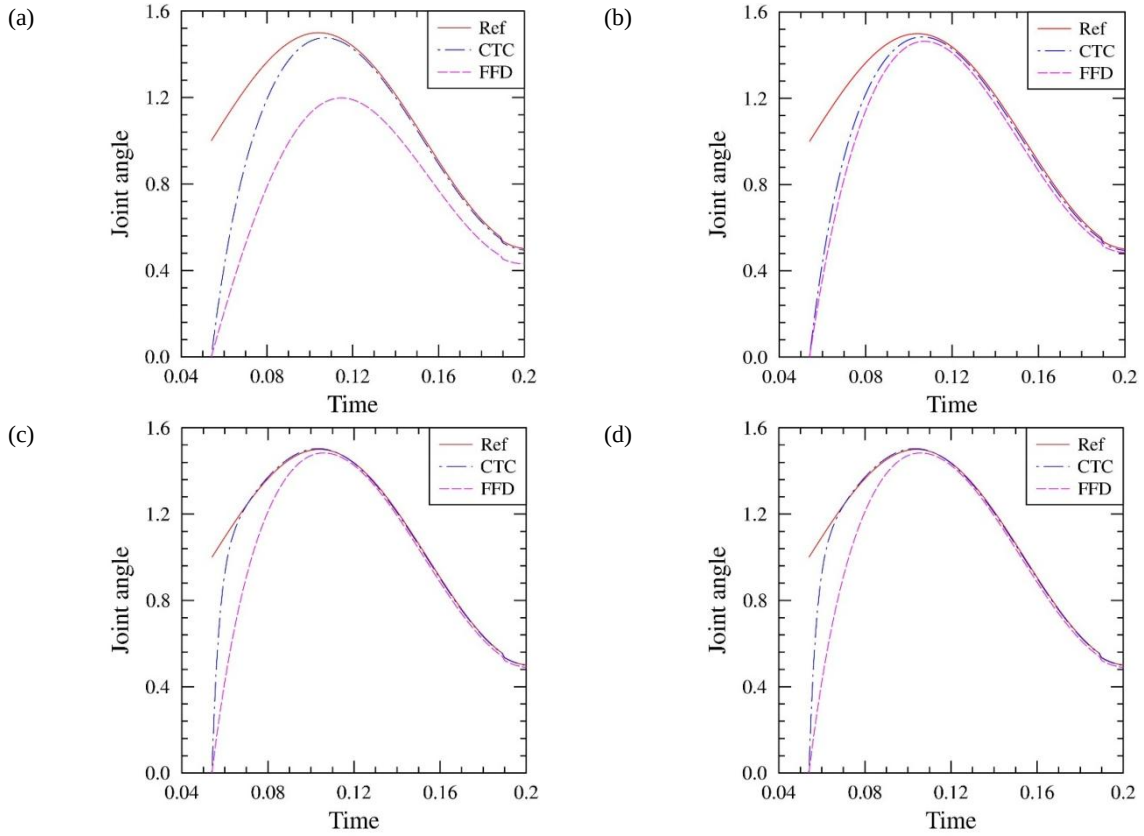


Fig. 11. Tracking performance of Model based controllers: CTC and FFD controllers (a) joint1, (b) joint 2, (c) joint 3, (d) joint 4.

(a) (b)

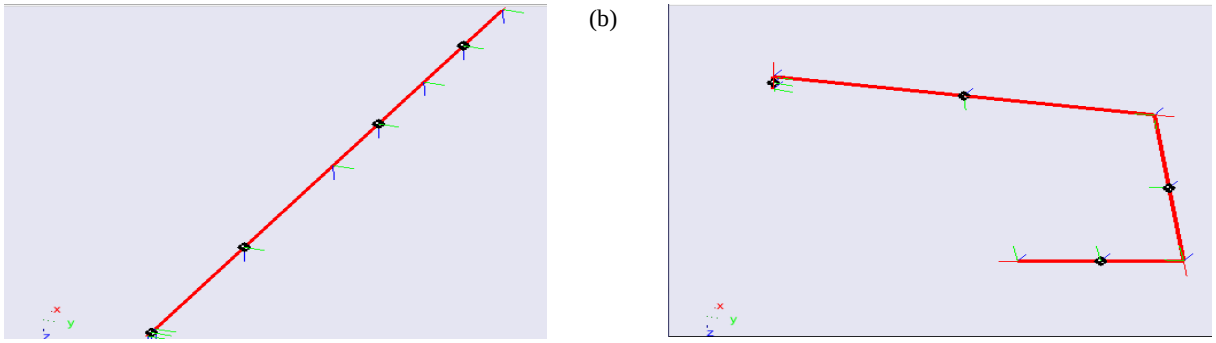


Fig. 12: Animation snapshots (a) t=0 (b) t=0.5s

5. CONCLUSION

The paper shows a preliminary study on position control methods for a robotic manipulator that is being developed to act as a human finger on a hand that can have multifarious uses within the scope of Human Computer Interaction like haptic rehabilitation applications. The aim of this manipulator would be to mimic the human finger's complicated motion during actions such as grasping an object. The controllers developed are based on PD independent joint control and have multiple PD loops in order to control a MIMO system as a set of SISO systems. Improvements on the conventional PD controller can be possible by introducing online gain tuning and learning capabilities. The simulation tests carried out confirm the superiority of these approaches over simple PD control. Model based approaches, that take an inverse dynamic plant model into consideration to reduce coupling of parameters, have also been developed, that has fared better than the model free approaches in the simulation tests performed. However, the complicated inverse dynamic model may be difficult to set up in software. Future work would include developing a force control method based on PD and model based strategies and subsequently a hybrid control method (motion and force control) for the manipulator, prior to manufacture and deployment of these strategies on hardware. Modern control strategies like Model Predictive Control are being studied to replace PD based controllers, if viable.

6. ACKNOWLEDGMENTS

This work was supported by the CSIR Supra Institutional Project under 12th plan. The authors would like to gratefully acknowledge the support and encouragement received from Director, CSIR-CMERI, Durgapur.

7. REFERENCES

[1] Mittal, R.K., Nagrath, I.J., 2003, Robotics and Control, Tata Mc-Graw Hill Publishing Company Limited.

[2] Lewis, F.L., Dawson, D.M., Abdullah, C.T., 1993, Robot Manipulator Control: Theory and Practice, Macmillan Pub. Co.

[3] Craig, J.J., 1988, Adaptive Control of Mechanical Manipulators, Addison-Wesley.

[4] Ouyang, P.R. and Zhang, W.J., 2004, "Comparison of PD-based Controllers for robotic Manipulators", Proceedings of the ASME Design Engineering Technical Conference.

[5] Ouyang, P.R, Zhang, W.J and M.M. Gupta, 2004, "Adaptive nonlinear PD Learning Control of Robotic Manipulators", Proceedings of the ASME Design Engineering Technical Conference.

[6] Paden, B. and Panja R., 1988, "Globally asymptotically stable PD+ Controller for robot manipulators", International Journal of Control, Vol. 47, No. 6, pg. 1697-1712.

[7] An, C.H., Atkeson, C.G., Griffiths, J.D., Hollerbach, J.M., 1989, "Experimental Evaluation of feedforward and computed torque control", IEEE Transactions on Robotics and Automation, Vol. 5, Issue 3.

[8] Khosla P.K. and Kanade, T., 1988, "Experimental Evaluation of Nonlinear Feedback and Feedforward Control Schemes for Manipulators", The International Journal of Robotics Research, Vol. 7, No. 1, pg. 18-28.

[9] Santibanez, V. and Kelly, R., 2001, "PD control with feedforward compensation for robot manipulators: analysis and experimentation", Robotica, Cambridge University Press, Vol. 19, Issue 1, pg. 11-19.

[10] Dung, L.T., Kang, H. and Ro, Y., 2010, "Robot Manipulator Modelling in Matlab-Simmechanics with PD control and online Gravity Compensation", IFOST 2010 Proceedings.

[11] De Luca, A., Siciliano, B. and Zollo, L., 2005, "PD control with on-line gravity compensation for robots with elastic joints: Theory and Experiments", Automatica 41, Elsevier, pg. 1809-1819.

[12] Middleton, R.H. and Goodwin, G.C., 1988, "Adaptive computed torque control for rigid link manipulators", Systems and control Letters, Elsevier, Vol. 10, Issue 1, pg. 9-16.

[13] MATLAB Simmechanics Documentation and User Guide. (http://www.mathworks.in/help/physmod/sm/).