Dynamic simulation of serial robots under force control

Arun Dayal Udai*

Department of Mechanical Engineering,
Birla Institute of Technology,
Mesra, Ranchi, India
Email: arun_udai@bitmesra.ac.in
*Corresponding author

Subir Kumar Saha

Department of Mechanical Engineering,
Indian Institute of Technology Delhi,
New Delhi, India
Email: saha@mech.iitd.ac.in

Abstract: The advantages of using force control in industrial robots are well known. Study of such systems in virtual environments in the form of simulation is of great help as most of the force controlled task works in close contact with the environment. In this paper, we show how to simulate different force control algorithms of a typical serial robot used in industries before deciding to choose a suitable one for real implementation. Hence, a proper dynamic model of the robot is essential which should be able to emulate the real robot, particularly if the robot moves at relatively higher speeds. This is done here using the concept of the decoupled natural orthogonal complement (DeNOC) matrices which is known to provide a recursive forward dynamic algorithm that is not only efficient but also numerically stable. Such simulation of robots under force control will allow users to tune the control gains without stopping the real robot on the production floor. Besides, such simulation can be used as an education tool as well to help beginners to explore various types of control algorithms and their performances. In addition, the framework for simulation proposed in this paper can work as a good test bench to test the performances of either a new control law or a different dynamic algorithm. As an illustration, the DeNOC based dynamics was substituted with MATLAB's SimMechanics which can also perform dynamic simulation. The comparison of the results validated the concept and correctness of the numerical simulations.

Keywords: simulation; force control; decoupled natural orthogonal complement; DeNOC.

Reference to this paper should be made as follows: Udai, A.D. and Saha, S.K. (2018) 'Dynamic simulation of serial robots under force control', *Int. J. Intelligent Machines and Robotics*, Vol. 1, No. 1, pp.79–108.

Biographical notes: Arun Dayal Udai is an Assistant Professor in the Department of Mechanical Engineering, BIT Mesra, India. He is currently on study leave and pursuing his PhD at the IIT Delhi, India. His research topic is adaptive force control of an industrial robot equipped with force/torque sensor. He completed his BTech in Marine Engineering from the MERI, Kolkata, India

in 1999 and worked as a Mercantile Marine Engineer with Anglo Eastern Ship Management, Hong Kong for four years. He did his MTech from BIT Mesra in 2006. He has co-authored a text book on computer graphics, which is published by Tata McGraw Hill, New Delhi in 2008. His research interests include robotics, mechatronics and embedded systems.

Subir Kumar Saha is currently a Naren Gupta Chair Professor in the Department of Mechanical Engineering, at the Indian Institute of Technology (IIT) Delhi, India. He graduated in 1983 from the NIT, Durgapur, India, completed his MTech from IIT Kharagpur, India, and PhD from the McGill University, Canada in 1991. Upon completion of his PhD, he joined Toshiba Corporation's R&D Center in Japan. Since 1996, he has been with IIT Delhi. He established the Mechatronics Laboratory at the IIT Delhi in 2001. He has more than 150 research publications in reputed journals/conference proceedings, and delivered more than 125 invited/keynote lectures. His research interests include dynamics of multi-body systems, robotics, mechatronics, etc. He has authored a text-book on introduction to robotics published by McGraw Hill in India, and co-authored two monographs in the area of dynamics published by Springer.

This paper is a revised and expanded version of a paper entitled 'Simulation of force control algorithms for serial robots' presented at 2012 IEEE/SICE International Symposium on System Integration (SII), Kyushu University, Fukuoka, Japan, 16–18 December 2012.

1 Introduction

Force controlled robots seem to have outreached traditional robots with passive springs at its end-effector, when it comes to rapidly adapting to the changing environment for compliant manipulation, surface finishing tasks and assembly operations. These robots are controlled using closed-loop with active force/torque sensory feedback. As they interact closely with the environment, it becomes a necessity to precisely design and test an algorithm before it is actually made operational on real robots. Simulating such force/torque control algorithms with a virtual robot could be one of the possible answers to such problems. As force control algorithms have evolved over the last three decades (Zeng and Hemani, 1997) and different techniques are used for simulating and testing these algorithms, there is a need to have a single platform where one can evaluate these control algorithms, or a precise robot dynamics algorithm, before deciding to use them for a specific application. For that, dynamics is important, particularly when the robot has to move fast, e.g., for cooperative manipulation.

The methodology proposed here is based on the decoupled natural orthogonal complement (DeNOC) matrices. It has been earlier applied to simulate a variety of robotic systems ranging from serial robots (Saha, 1999, 2003), closed-loop structures (Koul et al., 2013), tree-type legged walking machines (Shah et al., 2012) to a large degrees-of-freedom (DOF) a rope (Agarwal et al., 2013). Owing to its compact, modular and recursive formulation, the DeNOC-based dynamics offers an easy integration of robot dynamics to any control algorithm. This enabled us to develop a unified simulation structure to be built that can do complex robot dynamics, demonstrate environment

interactions and force/position control. The current work explores the capability of the DeNOC-based dynamic formulation to simulate several force/torque control algorithms using various robot architectures, which is an important contribution of this paper.

Note that, SimMechanics (Wood and Kennedy, 2003) and Simulink modules of MATLAB also allow one to perform robotic simulations (Udai and Saha, 2012; Cetinkunt and Book, 1989; Zhang and Ivlev, 2010; Kumar and Pathak, 2013). A symbolic modelling and dynamic simulation of robotic manipulators with compliant links and joints was proposed by Cetinkunt and Book (1989), where commercial symbolic computation was done in tools like SMP, MACSYMA and REDUCE to simulate a two-link manipulator. A dynamic simulator for compliant humanoid robot (CoMan) was reported by Dallali et al. (2013), which generated symbolic dynamic equations that can be used in MATLAB or C to carry out analysis. Symbolic computations are inherently slow and restrict users to analyse only smaller models. A SimMechanics based simulation of pneumatically-actuated soft robot was discussed in Zhang and Ivlev (2010). Dynamic modelling and simulation of a four legged jumping robot with compliant legs was discussed in Kumar and Pathak (2013), where MATLAB's Simulink environment was used for simulations. Modelling a robot in SimMechanics environment involves connecting multiple blocks for different links, joints, actuators, sensors, etc., which is quite cumbersome, particularly, when a robot comprises of long serial chain structure.

This paper is organised as follows: Section 2 describes the dynamic modelling of a general serial manipulator, which is used for the simulation of different force control algorithms, i.e., stiffness, impedance, admittance, and hybrid, discussed in the subsequent Section 3. Section 4 presents the modelling of a revolute-prismatic (RP) manipulator with different controller architectures, whose results are reported in Section 5. Section 6 presents the same for an industrial robot KUKA KR5 Arc. Section 7 concludes the paper.

2 Dynamic modelling of a serial robot

A typical industrial robot is a serial-chain mechanical system with a fixed-base, as shown in Figure 1. Dynamics of such systems are traditionally being modelled using Euler-Lagrange principle (EL), Newton-Euler (NE) equations or other formulations (Angeles, 2003). Each of them has their own merits and demerits. A comprehensive comparison was shown in Saha et al. (2013). In this context note that the formulation based on the concept of the DeNOC matrices allows one to derive analytical recursive algorithms both for inverse and forward dynamics which have been proven advantageous over traditional EL or NE approaches. Even though the DeNOC formulation starts with the unconstrained NE equations of motion, it finally leads to a set of coupled EL equations. Besides, modelling of even hyper-degree-of-freedom (upto 100,000) serial systems could provide realistic simulation, both in terms of computer CPU time and numerical accuracy (Agarwal et al., 2013), which otherwise are known to yield ill-conditioned system (Featherstone and Fijany, 1999). In a way, the DeNOC concept blends the positive aspects of NE and EL formulations, i.e., simplicity and compactness, respectively. The following subsections outline the steps for dynamic modelling of a general serial-chain robot shown in Figure 1. The same can be found in Saha et al. (2013, 2014), but it is briefly being included here for completeness of the paper.

2.1 Uncoupled NE equations

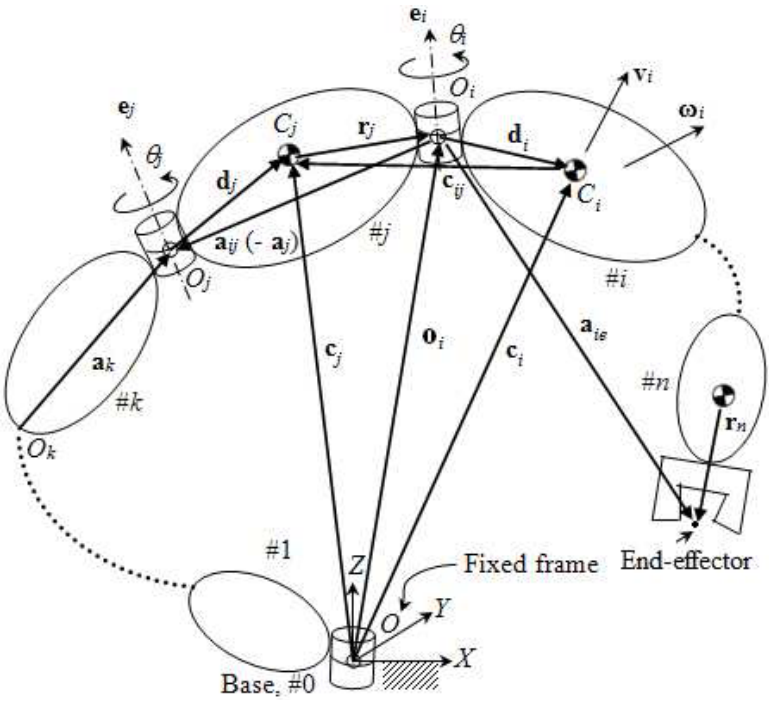
The uncoupled NE equations of motion for n links may be written in compact form as (Saha, 2014):

$$\mathbf{M}\dot{\mathbf{t}} + \mathbf{WMt} = \mathbf{w} \tag{1}$$

where $\mathbf{M}$ and $\mathbf{W}$ are respectively the $6n \times 6n$ generalised mass and angular-velocity matrix. They are defined as:

$$\mathbf{M} \equiv \text{diag.}\left[\mathbf{M}_1, \cdots, \mathbf{M}_n\right] \text{ and } \mathbf{W} \equiv \text{diag.}\left[\mathbf{W}_1, \cdots, \mathbf{W}_n\right] \tag{2}$$

Figure 1 A serial chain system



In (2), $\mathbf{M}_i$ and $\mathbf{W}_i$ are the 6×6 matrices of mass and angular velocity of the i^{th} link, which are defined as:

$$\mathbf{M}_i = \begin{bmatrix} \mathbf{I}_i & \mathbf{O} \\ \mathbf{O}m_i & \mathbf{1} \end{bmatrix} \text{ and } \mathbf{W}_i = \begin{bmatrix} \boldsymbol{\omega}_i \times \mathbf{1} & \mathbf{O} \\ \mathbf{O} & \mathbf{O} \end{bmatrix} \tag{3}$$

where $\boldsymbol{\omega}_i \times \mathbf{1}$ is the 3×3 cross-product tensor associated with the vector $\boldsymbol{\omega}_i$, i.e., $(\boldsymbol{\omega}_i \times \mathbf{1})\,\mathbf{x} = \boldsymbol{\omega}_i \times \mathbf{x}$, for any arbitrary three-dimensional Cartesian vector $\mathbf{x}$. Also, $\mathbf{1}$ and $\mathbf{O}$ are the 3×3 identity and null matrices, respectively, whereas, $\mathbf{I}_i$ and m_i are the 3×3 inertia tensor about the mass centre C_i, and mass of the i^{th} link, respectively.

Also, $\mathbf{t}$ and $\mathbf{w}$ are the $6n$-dimensional vectors of generalised twists and wrenches which are given by:

$$\begin{aligned}\mathbf{t} &\equiv \left[\mathbf{t}_1^T, \ldots, \mathbf{t}_n^T\right]^T, \text{ where } \mathbf{t}_i = \left[\boldsymbol{\omega}_i^T, \mathbf{v}_i^T\right]^T \\ \mathbf{w} &\equiv \left[\mathbf{w}_1^T, \ldots, \mathbf{w}_n^T\right]^T, \text{ where } \mathbf{w}_i = \left[\mathbf{n}_i^T, \mathbf{f}_i^T\right]^T\end{aligned} \tag{4}$$

In (4), $\boldsymbol{\omega}_i$ and $\mathbf{v}_i$ are the three-dimensional vectors of angular velocity and linear velocity of the mass centre C_i of the i^{th} link, whereas $\mathbf{n}_i$ and $\mathbf{f}_i$ are the three-dimensional vectors of moment and force applied about and at C_i, respectively.

2.2 Constrained kinematic equations

The constrained kinematic equations of motion in terms of the generalised twist $\mathbf{t}$ may be given as:

$$\mathbf{t} = \mathbf{N}\dot{\boldsymbol{\theta}}, \text{ where } \mathbf{N} \equiv \mathbf{N}_l\mathbf{N}_d \tag{5}$$

in which $\dot{\boldsymbol{\theta}} \equiv [\dot{\theta}_1, \cdots, \dot{\theta}_n]^T$ is the n-dimensional vector of joint-rates. The $6n \times 6n$ matrix $\mathbf{N}_l$ and $6n \times n$ matrix $\mathbf{N}_d$ are the decoupled form of the $6n \times n$ natural orthogonal complement (NOC) (Angeles and Lee, 1988) matrix $\mathbf{N}$, which together are referred to as the DeNOC matrices (Saha, 1999). These are given by:

$$\mathbf{N}_l = \begin{bmatrix} \mathbf{1} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{B}_{21} & \mathbf{1} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{B}_{n1} & \mathbf{B}_{n2} & \cdots & \mathbf{1} \end{bmatrix}, \mathbf{B}_{ij} \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ \mathbf{c}_{ij} \times \mathbf{1} & \mathbf{1} \end{bmatrix} \tag{6}$$

and

$$\mathbf{N}_d = \begin{bmatrix} \mathbf{p}_1 & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \mathbf{p}_2 & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{p}_n \end{bmatrix}, \mathbf{p}_{ij} \equiv \begin{bmatrix} \mathbf{e}_i \\ \mathbf{e}_i \times \mathbf{d}_i \end{bmatrix} \tag{7}$$

In (7), the six-dimensional vector $\mathbf{p}_i$ is for a revolute joint. In the presence of a prismatic joint, $\mathbf{p}_i \equiv [\mathbf{0}^T, \mathbf{e}_i^T]^T$. The vector $\mathbf{e}_i$ is the unit vector parallel to the axis of the i^{th} revolute joint or to the axis of linear motion of the i^{th} prismatic joint. Moreover, the vector $\mathbf{c}_{ij}$ is the vector which joins the mass centres of the two successive links, i.e., C_i and C_j, as in Figure 1. Vector $\mathbf{c}_{ij}$ is given by $\mathbf{c}_{ij} = -\mathbf{d}_i - \mathbf{r}_j$. The expression $\mathbf{c}_{ij} \times \mathbf{1}$ denotes the 3×3 cross-product tensor associated with the vector $\mathbf{c}_{ij}$ which is defined similar to $\boldsymbol{\omega}_i \times \mathbf{1}$ of (3). In (6) and (7), $\mathbf{0}$, $\mathbf{O}$ and $\mathbf{1}$ are respectively the vector of zeros, zero and identity matrices of compatible sizes. For example, in $\mathbf{N}_l$ of (6) $\mathbf{O}$ is the 6×6 matrix, whereas in $\mathbf{B}_{ij}$ it is the 3×3 matrix of zeros.

2.3 Coupled equations of motion

By pre-multiplying the $6n$-uncoupled NE equations of motion (1) by $\mathbf{N}^T$, and substituting the constrained kinematic equations in velocity (5), one obtains the minimal order dynamic equations of motion analogous to that of EL formulation, i.e.,

$$\mathbf{I}\ddot{\boldsymbol{\theta}} + \mathbf{h} = \boldsymbol{\tau}, \text{ where } \mathbf{h} \equiv \mathbf{C}\dot{\boldsymbol{\theta}} \tag{8}$$

and $\mathbf{I} \equiv \mathbf{N}^T\mathbf{M}\mathbf{N}$. Note that the $n \times n$ generalised inertia matrix (GIM) $\mathbf{I}$ is symmetric positive definite, whereas the $n \times n$ matrix of convective inertia (MCI) terms is given by $\mathbf{C} \equiv \mathbf{N}^T\left(\mathbf{M}\dot{\mathbf{N}} + \mathbf{W}\mathbf{M}\mathbf{N}\right)$. Correspondingly, the vector $\mathbf{h}$ comprises of Coriolis, centrifugal and centripetal forces. The vector $\boldsymbol{\tau} \equiv \mathbf{N}^T\mathbf{w}_e$ is the n-dimensional vector of driving torques and forces due to the joint actuators, gravity, environmental interaction, joint friction, etc.

2.4 Dynamic model simplification and simulation

For dynamic simulation, one is required to compute the elements of the GIM and MCI as obtained in (8) for multiple numbers of times for each successive robot position during the dynamic simulation of a robot. As computation of the GIM and MCI involves products with null matrices, it can be made computationally efficient by eliminating such computations and obtaining an analytical closed form expression. The expression for the $n \times n$ GIM is $\mathbf{I} \equiv \mathbf{N}^T\mathbf{M}\mathbf{N}$, which can be expressed using the block elements of the DeNOC matrices as:

$$\mathbf{I} \equiv \begin{bmatrix} i_{11} & \cdots & sym \\ \vdots & \ddots & \vdots \\ i_{n1} & \cdots & i_{nn} \end{bmatrix} \tag{9}$$

where $i_{ij} \equiv \mathbf{p}_i^T\tilde{\mathbf{M}}_i\mathbf{B}_{ij}\mathbf{p}_j$, for $i = n, \ldots, 1$ and $j = i - 1, \ldots, 1$. The 6×6 matrix $\tilde{\mathbf{M}}_i$ is evaluated from $i = n$ to 1 recursively as:

$$\tilde{\mathbf{M}}_i = \mathbf{M}_i + \mathbf{B}_{i+1,i}^T\tilde{\mathbf{M}}_{i+1}\mathbf{B}_{i+1,i} \tag{10}$$

In which $\tilde{\mathbf{M}}_{n+1} \equiv \mathbf{O}$, as there is no $(n + 1)^{st}$ link and hence, $\tilde{\mathbf{M}}_n \equiv \mathbf{M}_n$. This recursive formulation and symmetric form of the GIM reduces the computational complexity significantly. A recursive formulation for the MCI terms is discussed at length in Saha (1999), which may not be required to compute explicitly as one requires to compute vector $\mathbf{h}$ as a multiplication of matrix $\mathbf{C}$ and $\dot{\boldsymbol{\theta}}$. Vector $\mathbf{h}$ can be efficiently computed using any recursive inverse dynamics algorithm, as suggested by Saha (1999, 2003) and Luh et al. (1980), and others when $\ddot{\boldsymbol{\theta}} = \mathbf{0}$.

Note that for a robot simulation, one needs to perform forward dynamics, where the joint accelerations are calculated from (8) for a given set of forces/torques applied at its joints. It is assumed that the dynamic model of the robot and the initial joint angles and rates defining the initial state of the robot are fully known. One can then easily obtain the successive joint rates and angles by numerically integrating the joint accelerations $\ddot{\boldsymbol{\theta}}$ obtained from the dynamic equation (8) as:

$$\ddot{\boldsymbol{\theta}} = \mathbf{I}^{-1}\phi \text{ where } \phi \equiv \boldsymbol{\tau} - \mathbf{h} \tag{11}$$

In (11), no explicit inversion of the GIM is required to be performed. Instead, the $\mathbf{U}\mathbf{D}\mathbf{U}^T$ – based approach proposed in Saha (2003, 1997) is adopted here. In this approach, the matrix $\mathbf{I}$ is decomposed into upper and lower triangular matrices $\mathbf{U}$ and $\mathbf{L}$ using the reverse Gaussian elimination (RGE) (Saha, 1997), i.e., $\mathbf{I} = \mathbf{U}\mathbf{L}$. Note that for the

symmetric matrix **I**, it can be shown $\mathbf{L} = \mathbf{D}\mathbf{U}^T$, where **D** is a diagonal matrix. Hence, $\mathbf{I} = \mathbf{U}\mathbf{D}\mathbf{U}^T$. The decomposition leads to recursive order n, i.e. $O(n)$, forward dynamics computational algorithm to compute $\ddot{\boldsymbol{\theta}}$ from (11), which is very efficient (Shah et al., 2012).

3 Force control algorithms

The basic aim of any force/torque control algorithm is to efficiently synthesise the information of force/torque senses at the joints or at the proximal end, i.e., the end-effector or even at the base in some cases, to actively control the forces exerted by the robot to the environment with which it interacts. Other primary variables involved for such control are position, velocity, and acceleration. The approaches, however, vary by the way it relates the information to achieve certain specific requirements. Stiffness, impedance, admittance, and hybrid control are four basic force-control algorithms adopted in various industrial applications and reported in the literature (Zeng and Hemani, 1997). They are summarised next along with their implementation with the dynamic model developed in Section 2. It is pointed out here that stiffness and impedance controls do not have explicit force inputs, but they are still referred to as 'force control' algorithms in the literature (Zeng and Hemani, 1997) because they are used for force compliant tasks performed in the industries.

Note here that no comprehensive reporting of the above typical force control algorithms was traced by the authors in the open literature. Hence, the present paper is relevant and contributes in providing a framework to study four force control algorithms. Besides one can tune the controllers for practical use, test any new force control algorithm or test even a new dynamic simulation algorithm, etc. by just appropriately substituting the corresponding module provided in the MATLAB programs available from website (http://www.web.iitd.ac.in/saha/todownload/ieee-tro-force-control.zip). Such a development is extremely useful for robots operating at relatively high speed whose dynamical effects cannot be ignored while designing for an appropriate controller or its simulation.

3.1 Stiffness control

Quite a good number of research activities had been done in the past dealing with passive stiffness control which has a physical spring-like structure attached to the end-effector of a robot. In this paper, however, an active stiffness control like the one used by Roberts et al. (1985) and Salisbury (1980) is used, where the forces are sensed at the end-effector and a closed-loop is made to create a programmable spring like effect. Considering the robot's behaviour be governed by (8), the closed-loop dynamics may then be given by:

$$\mathbf{I}\ddot{\boldsymbol{\theta}} + \mathbf{h} = \boldsymbol{\tau} + \mathbf{J}^T\mathbf{w}^E \tag{12}$$

where **I**, **h** and $\boldsymbol{\tau}$ are defined earlier in Section 2.3, whereas the $6 \times n$ matrix $\mathbf{J} \equiv [\mathbf{J}_\omega^T \mathbf{J}_v^T]^T$ and the six-dimensional vector $\mathbf{w}^E$ are the robot's Jacobian and the external wrench due to interaction forces between the environment and the end-effector, respectively. The matrix **J** relates the six-dimensional twist vector of the end-effector $\mathbf{t}_e$ with the

n-dimensional joint rates $\dot{\boldsymbol{\theta}}$, as derived in Appendix. Note that the active positional stiffness control-loop comprises of a proportional feedback of reaction force and the end-effector position error that relates to the joint torque $\boldsymbol{\tau}_p$ (Salisbury, 1980) as:

$$\boldsymbol{\tau}_p = \mathbf{J}_v^T \mathbf{K}_{pp} \Delta\mathbf{p} \tag{13}$$

where $\Delta\mathbf{p}$ is the three-dimensional vector of positional errors of the end-effector with respect to its desired input trajectory $\mathbf{p}_D$, whereas $\mathbf{K}_{pp}$ is the 3 × 3 diagonal matrix containing constant scalar stiffness values along x, y and z axes of the end-effector. Similarly, the active orientation control loop, if desired, may be formed that relates the end-effector orientation error to the joint torque $\boldsymbol{\tau}_\psi$ as:

$$\boldsymbol{\tau}_\psi = \mathbf{J}_\omega^T \mathbf{K}_{p\psi} \Delta\boldsymbol{\psi} \tag{14}$$

where $\Delta\boldsymbol{\psi}$ is the three-dimensional vector of orientation errors of the end-effector with respect to its desired orientation $\boldsymbol{\psi}_D$, where as $\mathbf{K}_{p\psi}$ is the 3 × 3 matrix that takes into account for the orientation stiffness values corresponding to say, the roll-pitch-yaw (RPY) Euler angles of the robot's end-effector frame with respect to the inertial frame attached to the robot's base. The orientation representation could be using any other 3-parameters representation (Angeles, 2003) as well which will accordingly change the expression of $\mathbf{J}_\omega$ and $\mathbf{K}_{p\psi}$. Combining (13) and (14), one can write the following:

$$\boldsymbol{\tau}_q = \mathbf{J}^T \mathbf{K}_p \Delta\mathbf{q} \tag{15}$$

where for a 6-DOF robot $\mathbf{J}$ and $\mathbf{K}_p$ are the 6 × 6 matrices, whereas $\Delta\mathbf{q}$ is defined as the six-dimensional vector. They are defined as:

$$\mathbf{J} \equiv \begin{bmatrix} \mathbf{J}_\omega \\ \mathbf{J}_v \end{bmatrix}, \mathbf{K}_p \equiv \begin{bmatrix} \mathbf{K}_{p\psi} & \mathbf{O} \\ \mathbf{O} & \mathbf{K}_{pp} \end{bmatrix} \text{ and } \Delta\mathbf{q} \equiv \begin{bmatrix} \Delta\boldsymbol{\psi} \\ \Delta\mathbf{p} \end{bmatrix} \tag{16}$$

Control laws to accommodate the damping properties inherent to the robot joints can also be accounted for as:

$$\boldsymbol{\tau}_v = \mathbf{K}_{vq} \mathbf{J}^{-1} \mathbf{t}_e \tag{17}$$

in which $\mathbf{K}_{vq}$ is the $n \times n$ diagonal matrix of joint damping, $\mathbf{t}_e \equiv [\boldsymbol{\omega}_e^T \mathbf{v}_e^T]^T$ is the end-effector twist, where $\boldsymbol{\omega}_e$ and $\mathbf{v}_e$ respectively are the three-dimensional vectors containing angular and Cartesian velocities of the end-effector. Equation (17) plays an important role to stabilise the motion, and maintain a constant contact when the end-effector encounters an impact with its environment.

Next, the interaction forces and moments, i.e., the wrench $\mathbf{w}^E$, is modelled as a plane moving against the end-effector (Whitney, 1987). As a result, the six-dimensional external wrench $\mathbf{w}^E$ is expressed as:

$$\mathbf{w}^E = \begin{cases} \mathbf{K}_E \left(\mathbf{q} - \mathbf{q}^E \right) & \forall \mathbf{q} > \mathbf{q}^E \\ \mathbf{0} & \forall \mathbf{q} > \mathbf{q}^E \end{cases} \tag{18}$$

In (18), $\mathbf{q} \equiv [\boldsymbol{\psi}_e \ \mathbf{p}_e]$T is the end-effector configuration corresponding to the end-effector orientation $\boldsymbol{\psi}_e$ and position $\mathbf{p}_e$, which can be calculated from the robot's forward kinematics algorithm provided the joint angles are known. Moreover, the vector

$\mathbf{q}^E$ represents the configuration of the moving plane comprising of its position of any point on it and the orientation. Accordingly, $\mathbf{K}_E \equiv \text{diag.}[\mathbf{K}_{E\psi}, \mathbf{K}_{Ep}]$ is the 6 × 6 diagonal matrix containing stiffness about and along three orthogonal directions corresponding to the rotation and translation. Note that the 6 × 6 diagonal compliance matrix $\mathbf{K}_F \equiv \text{diag.}[\mathbf{K}_{F\psi}, \mathbf{K}_{Fp}]$ is used here to transform the actual force felt by the end-effector due to the environment, (18), to the corresponding end-effector's orientation and position, i.e., $\mathbf{q} = \mathbf{K}_F\mathbf{w}^E$. The joint torques corresponding to interaction forces and moments are then given by:

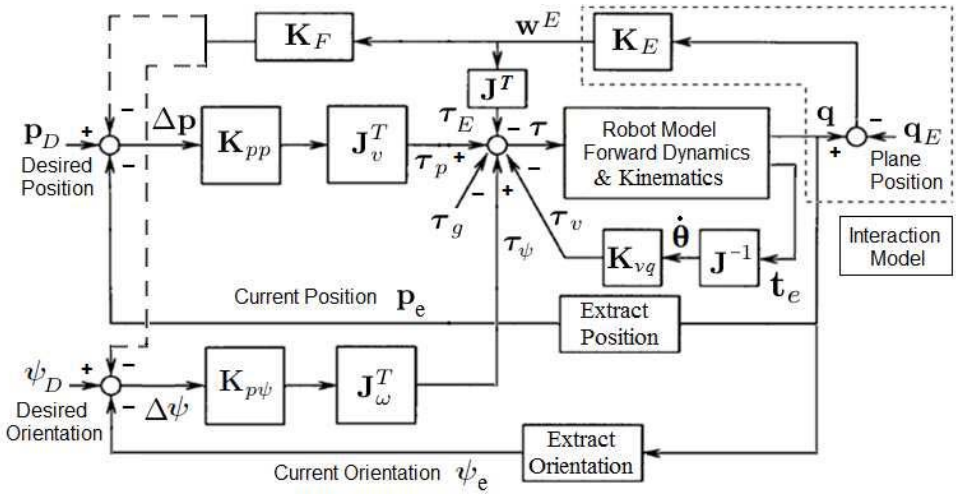
$$\boldsymbol{\tau}_E = \mathbf{J}^T\mathbf{w}^E \tag{19}$$

where $\mathbf{w}^E$ is given in (18). As per (12), the net controlling torque acting at the robot's joints is then given by:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_q - \boldsymbol{\tau}_v - \boldsymbol{\tau}_E - \boldsymbol{\tau}_g \tag{20}$$

where $\boldsymbol{\tau}_q$, $\boldsymbol{\tau}_v$ and $\boldsymbol{\tau}_E$ are given in (15), (17) and (19), respectively, whereas, $\boldsymbol{\tau}_g$ is the torque due to gravity which is derived in Appendix. The overall control scheme is shown in Figure 2.

Figure 2 Stiffness control model



3.2 Impedance control

In impedance control, a system is so designed that the end-effector applies a wrench $\mathbf{w}_e$ which is proportional to the end-effector's twist $\mathbf{t}_e$, i.e., $\mathbf{w}_e = \mathbf{K}_v\mathbf{t}_e$. The constant of proportionality is known as mechanical impedance of the manipulator. A robot with an impedance controller is desirable for an application where the robot is demanded not to just track the trajectory alone but also to regulate the mechanical impedance of the manipulator (Zeng and Hemani, 1997). Such systems are able to resist any end-effector motion due to sudden load on the end-effector due to impact. In contrast to basic stiffness control law of (15), the commanded torque for error correction is given by Whitney (1987) as:

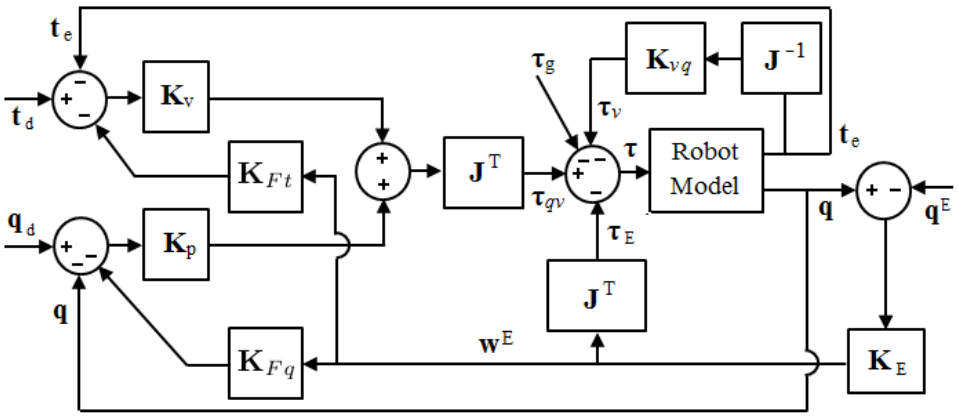
$$\boldsymbol{\tau}_{qv} = \mathbf{J}^T \left(\mathbf{K}_p \Delta\mathbf{q} + \mathbf{K}_v \Delta\mathbf{t}_e \right) \tag{21}$$

where $\mathbf{K}_p$ and $\mathbf{K}_v \equiv \text{diag.}[\mathbf{K}_{v\psi}, \mathbf{K}_{vp}]$ are the control gain matrices for the pose and twist errors, respectively, of the end-effector. The system is fed with both pose and twist trajectories $\mathbf{q}_D$ and $\mathbf{t}_D$, and the net torque $\boldsymbol{\tau}_{qv}$ was calculated based on their corresponding errors $\Delta\mathbf{q}$ and $\Delta\mathbf{t}_e$. The 6 × 6 compliance matrices $\mathbf{K}_{Fq}$ and $\mathbf{K}_{Ft}$, similar to those $\mathbf{K}_F$ defined after (18), are used to transform the environment (i.e., moving plane) pose and twist to the corresponding end-effector pose and twist, respectively. Its implementation was done by rearranging the control blocks of stiffness controller of Figure 2, which is shown in Figure 3. The orientation of the end-effector was maintained constant by utilising the end-effector's orientation error to the existing model (Zeng and Hemani, 1997) by adding control loops analogous to (14). The overall driving torque of the robot is thus given by:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{qv} - \boldsymbol{\tau}_v - \boldsymbol{\tau}_E - \boldsymbol{\tau}_g \tag{22}$$

where $\boldsymbol{\tau}_{qv}$ is given by (21).

Figure 3 Impedance control model



3.3 Admittance control

Admittance **A** is defined as the inverse of impedance such that:

$$\mathbf{t}_e = \mathbf{A}\mathbf{w}^E \tag{23}$$

where $\mathbf{t}_e$ is the end-effector twist, matrix $\mathbf{A} \equiv \text{diag.}[\mathbf{A}_\psi, \mathbf{A}_p]$ is the 6 × 6 diagonal matrix of admittance and $\mathbf{w}^E$ is the sensed wrench vector at the end-effector. Note here that $\mathbf{A} = \mathbf{K}_v^{-1}$ for any non-redundant robot manipulator, say, a 6-DOF robot performing tasks in the 6-DOF Cartesian space. Admittance control allows a desired wrench input $\mathbf{w}_D$ to be commanded explicitly to a pose controlled robot. This is quite analogous to a position controlled robot being controlled for force using an external force control loop. The pose $\mathbf{q}_a$ is fed to the robot controller corresponding to the desired wrench input $\mathbf{w}_D$. This may be given as:

$$\mathbf{q}_a = \int \mathbf{A}\left(\mathbf{w}_D - \mathbf{w}^E\right)dt \tag{24}$$

For a robot accepting torque commands at its joints, the contribution of torque due to the deviation of pose from the value as calculated in (24) and the desired pose input $\mathbf{q}_D$ is given as (Zeng and Hemani, 1997):

$$\boldsymbol{\tau}_a = \mathbf{K}_{pa}\mathbf{J}^{-1}\Delta\boldsymbol{\chi} \tag{25}$$

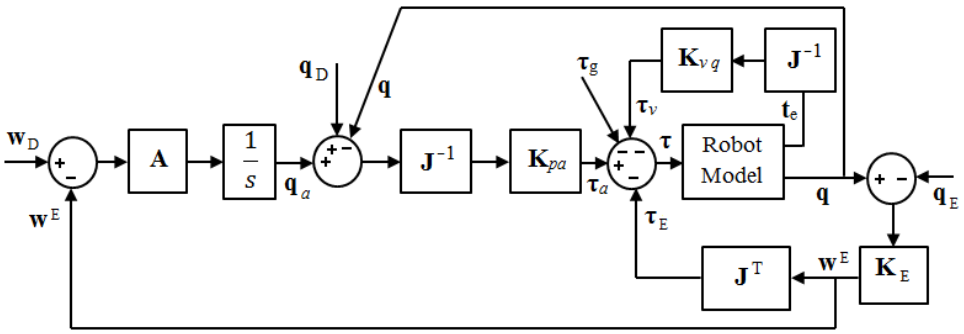
where $\Delta\boldsymbol{\chi} = \mathbf{q}_D + \mathbf{q}_a - \mathbf{q}$ and $\mathbf{K}_{pa}$ is the $n \times n$ diagonal matrix of proportional control gain. The remaining driving torques are $\boldsymbol{\tau}_v$, $\boldsymbol{\tau}_E$, and $\boldsymbol{\tau}_g$ which are given by (17), (19), and (46), respectively. Thus, the net driving torque is given by:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_a - \boldsymbol{\tau}_v - \boldsymbol{\tau}_E - \boldsymbol{\tau}_g \tag{26}$$

where $\boldsymbol{\tau}_a$ is given by (25).

Figure 4 shows the implementation of the basic admittance control depicted by (26). The admittance matrix **A** relates the wrench error vector to the end-effector twist (Whitney, 1987; Schutter and Brussel, 1988; Seraji, 1994). It can be designed to achieve a wrench response with low or no error, low overshoot, and rapid rise time. Admittance control focuses more on desired wrench tracking as compared to impedance control where pose and twist are tracked. Such systems are more suitable where the system demands maintaining a constant reaction wrench on making a contact with the object surface, e.g., grinding, polishing, buffing, etc.

Figure 4 Admittance control model



3.4 Hybrid control

Hybrid position/force control combines two independent control systems into a single controller based on two complimentary orthogonal spaces on displacement and force, as defined in Mason (1981). A hybrid control system was proposed in Raibert and Criag (1981) which was later extended to operational space by Khatib (1987). The same system is used here with a more generic representation of position and force, i.e., pose and wrench, respectively. The 6 × 6 diagonal compliant selection matrix **S** determines the subspace for which the wrench or the pose is to be controlled. Both cannot be simultaneously controlled (Saha, 2014; Craig, 2004). The diagonal element s_{ii} is set to 1, if the i^{th} degree-of-freedom of the end-effector is to be wrench controlled. The

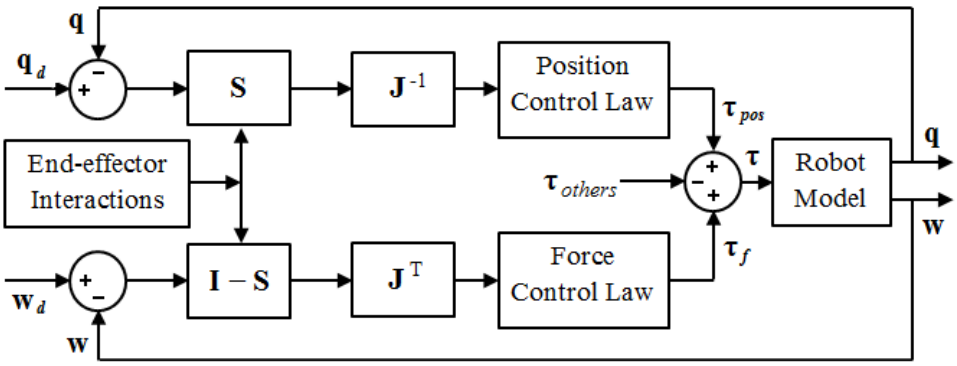
commanded torque acts simultaneously on the decoupled pose and wrench subspace due to corresponding errors. The algorithm takes input of both the desired pose and wrench trajectories. Hybrid control is more suited for robots that are designed to work for precise positioning as well as wrench tracking control in contact.

Control laws for the pose and wrench are typically chosen as proportional-integral-derivative (PID) controllers. The system takes torque as input from its actuators and gives the end-effector pose as output, where the wrench is sensed through a proximal force/torque sensor at its end-effector. The environment interaction model (18) generates the reaction wrench and also sets the wrench compliance selection matrix **S** elements to one for pose constrained directions. Thus, the model enters into wrench control mode along the pose constrained direction and pose control mode along remaining direction(s). The net torque is due to the combined contributions of torques due to pose and wrench tracking errors, namely, $\boldsymbol{\tau}_{pos}$ and $\boldsymbol{\tau}_f$ respectively. The total torque $\boldsymbol{\tau}$ is then given by:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{pos} + \boldsymbol{\tau}_f + \boldsymbol{\tau}_{others} \tag{27}$$

where $\boldsymbol{\tau}_{others}$ is the torque due to joint damping, external wrench and due to gravitational force, given by (17), (19) and (46), respectively. Figure 5 shows the implementation of the hybrid control.

Figure 5 Hybrid control model



4 Modelling of a RP robot

In order to test the force control algorithms discussed in Section 3 in the absence of a real robot, particularly when the robot moves at relatively higher speeds, its dynamic model described in Section 2 can be used. A RP planar robot model analogous to (Raibert and Criag, 1981) was considered, which is shown in Figure 6. Such consideration will allow us to validate the simulation results generated in our work. The Denavit-Hartenberg (DH) parameters of the RP robot is given in Table 1. Note here that the definitions of the DH parameters are adopted from Saha (2014), which are given in Appendix for easy reference to the readers.

Figure 6 RP robot model (see online version for colours)

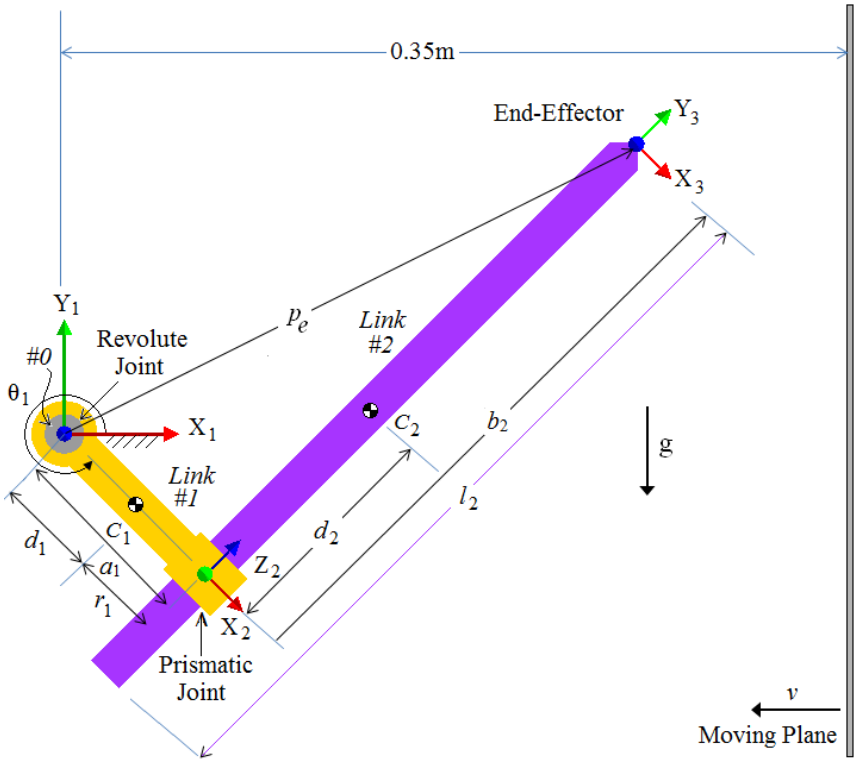


Table 1 DH parameters of the RP robot

Link	*Joint offset*	*Joint angle*	*Link length*	*Twist angle*
i	$b_i(m)$	$\theta_i(rad.)$	$a_i(m)$	$\alpha_i(rad.)$
1	0	θ_1 (*JV**)	$\alpha_1 = 0.079$	$-\pi/2$
2	b_2(*JV*)	0	0	$\pi/2$

Note: Length of link #2, $l_2 = 1.0$ *m*, and **JV* ≡ joint variable.

4.1 Derivation of the dynamic model

The two-link 2-DOF planar RP robot is shown in Figure 6. Its independent joint variables are θ_1 and b_2. Correspondingly, the vector of joint variables is given by $\boldsymbol{\theta} = [\theta_1, b_2]^T$. The joint rate vector is $\dot{\boldsymbol{\theta}} = [\dot{\theta}_1, \dot{b}_2]^T$. The physical link lengths are a_1 and l_2, and the masses are m_1 and m_2. The mass centres are assumed to be located at the middle of the links. The kinematic constraints, as defined in (6) and (7), are derived as:

$$\mathbf{p}_1 = \begin{bmatrix} \mathbf{e}_1 \\ \mathbf{e}_1 \times \mathbf{d}_1 \end{bmatrix}; \mathbf{p}_2 = \begin{bmatrix} 0 \\ \mathbf{e}_2 \end{bmatrix}$$
$$\mathbf{B}_{21} = \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ -(\mathbf{r}_1 + \mathbf{d}_2) \times 1 & \mathbf{1} \end{bmatrix}; \mathbf{B}_{22} = \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ \mathbf{O} & \mathbf{1} \end{bmatrix} \tag{28}$$

where $[\mathbf{e}_1]_1 \equiv [0\ 0\ 1]^T$, $[\mathbf{e}_2]_1 \equiv [-\sin\theta_0 \cos\theta_1\ 0]^T$, $[\mathbf{d}_1]_1 \equiv [\mathbf{r}_1]_1 \equiv [d_1 \cos\theta_1\ d_1 \sin\theta_1\ 0]^T$, and $[\mathbf{d}_2]_1 \equiv [-d_2 \sin\theta_1\ d_2 \cos\theta_1\ 0]^T$, where $d_1 = a_1 / 2$ and $d_2 = b_2 - l_2 / 2$. Moreover, the symbol $[\cdot]_k$ is used to represent a vector/matrix quantity in frame k. The elements of the 2×2 GIM $\mathbf{I}$ are then evaluated from (9) as follows: for $i, j = 2$, the calculation of i_{22} is given by:

$$i_{22} \equiv \mathbf{p}_2^T \tilde{\mathbf{M}}_2 \mathbf{B}_{22} \mathbf{p}_2 = m_2 \tag{29}$$

where $\tilde{\mathbf{M}}_2 = \mathbf{M}_2 \equiv \begin{bmatrix} \mathbf{I}_2 & \mathbf{O} \\ \mathbf{O} m_2 & \mathbf{1} \end{bmatrix}$. Similarly, the element of the GIM, $i_{21} = i_{12}$ is expressed as:

$$i_{21} \equiv \mathbf{p}_2^T \tilde{\mathbf{M}}_2 \mathbf{B}_{21} \mathbf{p}_1 = m_2 a_1 \tag{30}$$

Finally, the element i_{11} is given by:

$$i_{11} = \mathbf{p}_1^T \mathbf{M}_1 \mathbf{p}_1 = m_1 a_1^2 / 3 + m_2 l_2^2 / 12 + m_2 \left[a_1^2 + (b_2 - l_2 / 2)^2 \right] \tag{31}$$

where $\tilde{\mathbf{M}}_1 = \mathbf{M}_1 + \mathbf{B}_{21}^T \tilde{\mathbf{M}}_2 \mathbf{B}_{21}$ and $\mathbf{M}_1 \equiv \begin{bmatrix} \mathbf{I}_1 & \mathbf{O} \\ \mathbf{O} m_1 & \mathbf{1} \end{bmatrix}$. Assuming the links to be slender solid bars, $\mathbf{I}_1$ and $\mathbf{I}_2$ are given by:

$$[\mathbf{I}_1]_2 = \frac{1}{12} m_1 a_1^2 \text{diag.}[0 \quad 1 \quad 1] \text{ and } [\mathbf{I}_2]_3 = \frac{1}{12} m_2 l_2^2 \text{diag.}[1 \quad 0 \quad 1] \tag{32}$$

Next, $\mathbf{h}$ can be derived using (8) as the two-dimensional vector given below:

$$\mathbf{h} = \mathbf{C}\dot{\boldsymbol{\theta}} = \begin{bmatrix} 2m_2 (b_2 - l_2 / 2) \dot{b}_2 \dot{\theta}_1 \\ -m_2 (b_2 - l_2 / 2) \dot{\theta}_1^2 \end{bmatrix} \tag{33}$$

For calculating $\dot{\mathbf{N}}$, $\dot{\mathbf{r}}_i = \boldsymbol{\omega}_i \times \mathbf{r}_i$ and $\dot{\mathbf{d}}_i = \boldsymbol{\omega}_i \times \mathbf{d}_i$ were used. In case of the prismatic joint these vectors are also the functions of the joint displacement b_2.

4.2 Jacobian and gravity torques

Here, the 2×2 Jacobian matrix $\mathbf{J}_v$ of the RP robot is derived using the knowledge of Appendix as:

$$\mathbf{J}_v = \begin{bmatrix} -a_1 \sin\theta_1 & -b_2 \cos\theta_1 & -\sin\theta_1 \\ a_1 \cos\theta_1 & -b_2 \sin\theta_1 & \cos\theta_1 \end{bmatrix} \tag{34}$$

The torque due to gravity $\boldsymbol{\tau}_g$ was then derived from the knowledge of Appendix as:

$$\boldsymbol{\tau}_g = \begin{bmatrix} m_2 g\left[a_1 \cos\theta_1 - (b_2 - l_2/2)\sin\theta_1\right] + m_1 g a_1 \cos\theta_1 / 2 \\ m_2 g \cos\theta_1 \end{bmatrix} \tag{35}$$

This completes the dynamic equations of motion for the RP robot, which is given by $\mathbf{I}\ddot{\boldsymbol{\theta}} + \mathbf{h} = \boldsymbol{\tau}$, where the other contributions to $\boldsymbol{\tau}$, namely, $\boldsymbol{\tau}_q$, $\boldsymbol{\tau}_v$, $\boldsymbol{\tau}_E$, $\boldsymbol{\tau}_{qv}$, $\boldsymbol{\tau}_a$, $\boldsymbol{\tau}_{pos}$ and $\boldsymbol{\tau}_f$, depend on the type of force control law desired, as derived in Section 3.

4.3 Simulation of force controllers

For the simulation of stiffness, impedance, admittance and hybrid controllers, the robot was commanded to stay in its initial position and a moving plane was made to collide against the end-effector from the right as shown in Figure 6. This allowed an easy testing of the control behaviours. The reaction force should rise proportionally to the end-effector displacement if the robot is to behave like a spring in stiffness control. However, in case of hybrid control the robot may be commanded for both position and force. The simulation was done similarly with the other controllers so as to have an easy comparison. Once the controlling torque was calculated as discussed in Section 3, using (20), (22), (26) and (27), the system may be expressed in the state-space form as:

$$\dot{\mathbf{y}}(t) \equiv \begin{bmatrix} \dot{\mathbf{y}}_1(t) \\ \dot{\mathbf{y}}_2(t) \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{y}}_2(t) \\ \mathbf{I}^{-1}(\boldsymbol{\tau} - \mathbf{h}) \end{bmatrix} \tag{36}$$

where $\mathbf{y}_1(t) = \boldsymbol{\theta}$ and $\mathbf{y}_2(t) = y_2(t)\dot{\boldsymbol{\theta}}$. This was integrated numerically using in-built function ODE45 of MATLAB®, which is based on the explicit Runge-Kutta (4, 5) formula or the Dormand-Prince pair (Dormand and Prince, 1980). The controller models were built using MATLAB/Simulink blocks as available in Udai and Saha (2012). Due to space constraints they are not shown here, but it can be downloaded from website (http://www.web.iitd.ac.in/saha/todownload/ieee-tro-force-control.zip).

5 Control simulation of RP robot

This section reports the simulation results of different controllers and their performances. For simulation, a typical value for the stiffness of the moving plane along its normal was taken as 9×10^5 N/m for all the controllers and no stiffness was considered along other directions.

5.1 Stiffness controller

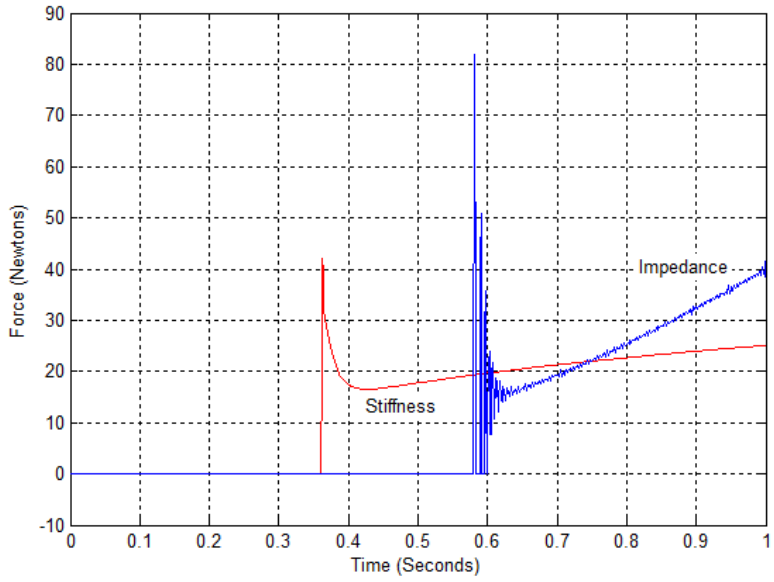
The end-effector of the robot was fixed to a position at (0.332 m, 0.3842 m, 0), i.e., $[\mathbf{p}_e]_1 \equiv [0.332\ 0.3842\ 0]^T$ with the initial joint positions as $\boldsymbol{\theta} \equiv [-32°, 0\text{ m}]$. Since a planar 2-DOF robot is used only for positioning purposes, no orientation aspect was considered during simulation. In order to understand the behaviour of the end-effector as a spring (Raibert and Criag, 1981), the plane situated at a distance of 0.35 m from the base frame of the robot, as shown in Figure 6, was moved with a constant velocity of $v_x = -0.05$ m/s for 1 s. The simulation results are shown in Figure 7. The plane makes contact with the

end-effector after 0.3606 s, as evident from Figure 7(a), and moves the end-effector to the left of X by 0:032 m at 1 s, which is shown in Figure 7(b). As clear from Figure 7(a), the end-effector encounters an impact where the reaction force is surged. Thereafter, the reaction force increases linearly like a spring with time to 25 N at time equal to 1 s. The various constants used during the simulation are:

$$\mathbf{K}_{pp} = \begin{bmatrix} 50 & 0 & 0 \\ 0 & 1000 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\mathrm{N}}{\mathrm{m}}, \mathbf{K}_{Fp} = \begin{bmatrix} 0.01 & 0 & 0 \\ 0 & 0.01 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\mathrm{m}}{\mathrm{N}}, \quad \mathbf{K}_{vq} = \begin{bmatrix} 200\frac{\mathrm{Nms}}{\mathrm{rad}} & 0 \\ 0 & 200\frac{\mathrm{Ns}}{\mathrm{m}} \end{bmatrix} \tag{37}$$

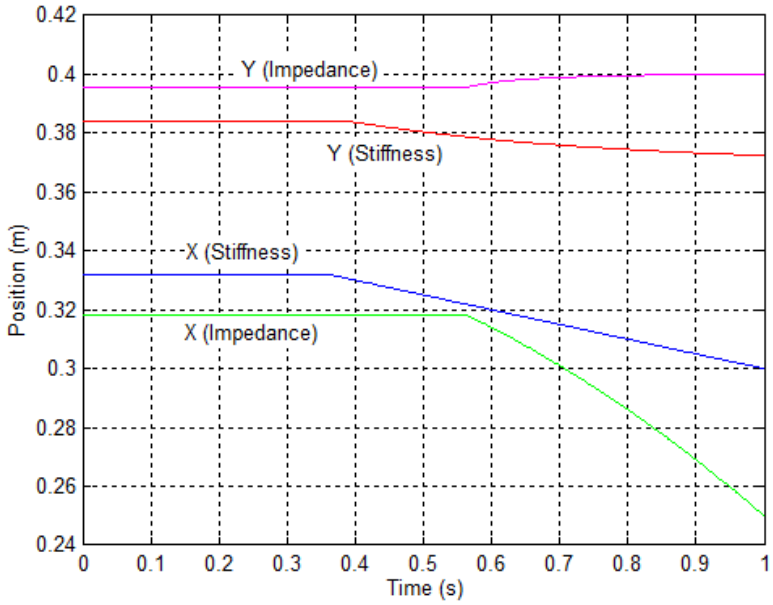
The linear variation of the force with the end-effector displacement shown in Figure 7(a) demonstrates that the robot end-effector is equivalent to a physical spring. It may be observed in Figure 7(b) that the end-effector moves with the plane once the contact is established at 0.3606 s. Figure 7(c) shows the variation of the joint torque and force during the robot-plane interaction with the stiffness controller.

Figure 7 Performance of the RP robot with stiffness and impedance controllers, (a) end-effector force with stiffness and impedance controllers (b) end-effector position with stiffness and impedance controllers (c) joint torque and force with stiffness and impedance controller (see online version for colours)

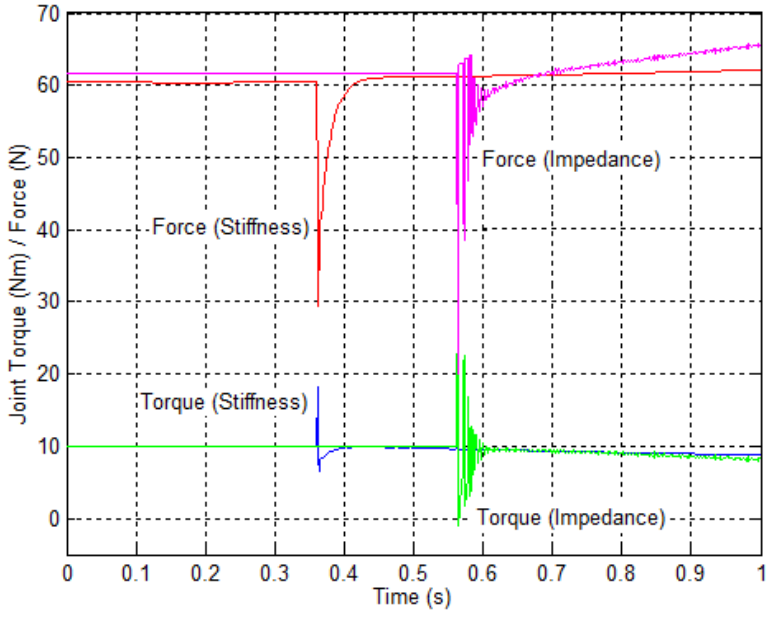


(a)

Figure 7 Performance of the RP robot with stiffness and impedance controllers, (a) end-effector force with stiffness and impedance controllers (b) end-effector position with stiffness and impedance controllers (c) joint torque and force with stiffness and impedance controller (continued) (see online version for colours)



(b)



(c)

5.2 Impedance controller

As the model now requires to be tested with the variation of velocity, the vertical plane was moved with a constant acceleration of –0.2 m/s^2 so that the velocity changes linearly from 0 to –0.2 m/s during 1 s. The end-effector's desired position was taken as $[\mathbf{p}_e]_1 \equiv [0.3184, 0.3955, 0]^T$ with initial joint positions as $\boldsymbol{\theta} \equiv [-30°, 0]$ and zero velocity, i.e., $\dot{\boldsymbol{\theta}} = [0, 0]$. Note that the end-effector and joint positions were taken different from that of the stiffness control to avoid any overlapping of the plots. It may be observed in Figure 7(a) that the reaction force surges to 81.84 N on impact at 0.5796 s and then damps down before increasing again linearly with the velocity of the end-effector to 40 N at 1 s. This is in agreement with the impedance controller characteristics. The initial surge and jitter during impact may be reduced by slowly approaching the end-effector and by optimising the control gains. The gains for the simulation were taken as:

$$\mathbf{K}_{pp} = \begin{bmatrix} 300 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\text{N}}{\text{m}} \text{ and } \mathbf{K}_{vp} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{\text{Ns}}{\text{m}} \tag{38}$$

The end-effector follows the moving plane once the contact is established at 0.5796 s, which is also depicted in Figure 7(b). The joint torque and force required to generate the robot-plane interaction under impedance control are shown in Figure 7(c).

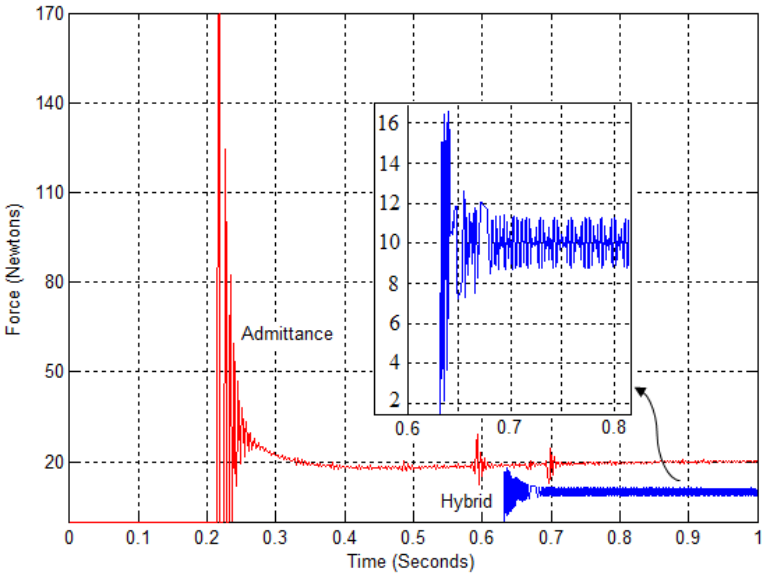
5.3 Admittance controller

For admittance control, a desired reaction force and the end-effector position $\mathbf{p}_e$ were set at 20 N and $[\mathbf{p}_e]_1 = [0.3184\ 0.3955\ 0]^T$. The moving plane was initially at 0.35 m from the end-effector, which was made to approach with a constant velocity of $v_x = -0.01$ m/s. The plane makes the initial contact at 0.215 s, as seen in Figure 8(a). It initially shoots up due to impact and finally maintained a constant reaction force of 20 N, as desired, within 0.5 s. The values of various constants like joint control gains $\mathbf{K}_{vp}$ and $\mathbf{K}_{vq}$ for the position error and to accommodate joint damping, respectively and the admittance $\mathbf{A}_p$ were taken as:

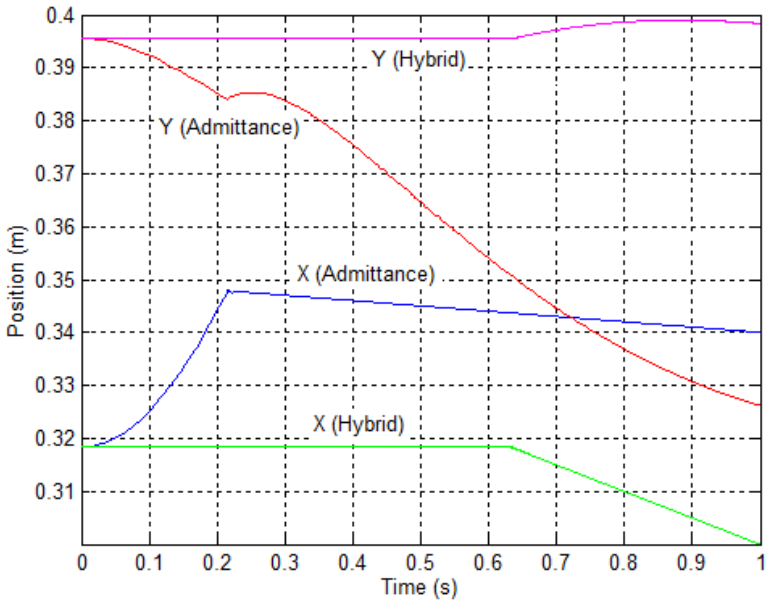
$$\mathbf{K}_{vp} = \begin{bmatrix} 100\frac{\text{Nms}}{\text{rad}} & 0 \\ 0 & 100\frac{\text{Ns}}{\text{m}} \end{bmatrix} \frac{N}{m}, \mathbf{K}_{vq} = \begin{bmatrix} 30\frac{\text{Nms}}{\text{rad}} & 0 \\ 0 & 30\frac{\text{Ns}}{\text{m}} \end{bmatrix}, \mathbf{A}_p = \begin{bmatrix} 0.03\frac{\text{m}}{\text{Ns}} & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{39}$$

The end-effector seeks to attain the desired force of 20 N even if it was desired to maintain the initial position. Hence, the movement occurs even before the end-effector makes contact with the plane as depicted in Figure 8(b). Once the contact was established, the plane moves with the end-effector so as to maintain the desired contact force. The end-effector position is not clearly maintained while the plane keeps moving. Corresponding joint torque and force required to attain the required control behaviour are shown in Figure 8(c).

Figure 8 Performance of the RP robot with admittance and hybrid controllers, (a) end-effector force with admittance and hybrid controllers (b) end-effector position with admittance and hybrid controllers (c) joint torque and force with admittance controller (see online version for colours)

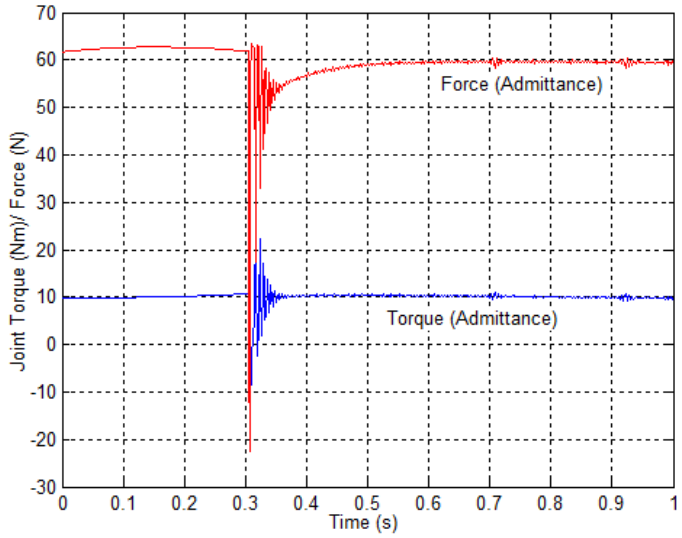


(a)



(b)

Figure 8 Performance of the RP robot with admittance and hybrid controllers, (a) end-effector force with admittance and hybrid controllers (b) end-effector position with admittance and hybrid controllers (c) joint torque and force with admittance controller (continued) (see online version for colours)



(c)

5.4 Hybrid controller

The plane, which initially was at 0.35 m from the robot base, was made to move with a uniform velocity of 0.05 m/s towards the robot's end-effector. This is about eight times higher velocity than that reported in Raibert and Criag (1981), which was 0.0065 m/s. The robot was commanded to remain stationed at $[\mathbf{p}_e]_1 = [0.3184 \text{ m } 0.3955 \text{ m } 0]^T$ with the joint angles $\boldsymbol{\theta} = [-30°, 0 \text{ m}]$ and zero velocities, i.e., $\boldsymbol{\theta} = [0, 0]$. As seen in Figure 8(a), the plane collides with the end-effector at 0.6321 *s* where the reaction force surges to 16.3 N. The system gradually converges to the desired force of 10 N at 0.68 s, i.e., 0.05 s after the contact was established.

The system successfully entered into the force control mode along the perpendicular to the plane upon making contact at 0.6321 s with the plane while maintaining position control mode along the remaining orthogonal directions. The force and the position behaviour are well comparable to the one obtained in Raibert and Criag (1981). Note that the jitters in the force values of Figure 8(a) are probably due to non-optimised control gains, as pointed out for stiffness control. Figure 8(b) shows the end-effector position variation with time. The behaviour is different from admittance controller because the hybrid controller simultaneously servos for desired force as well as position. The joint torque and force varied so as to maintain the required controller behaviour. As can be seen from Figure 8(a), the end-effector maintains the desired reaction force with jitters, the same is reflected on the joint torque and force, and, hence is not shown here to avoid clumsy plots. However, one can quickly generate it using the MATLAB model of website (http://www.web.iitd.ac.in/saha/todownload/ieee-troforce-control.zip).

5.5 Comparison with SimMechanics models

In order to compare and demonstrate the use of different dynamic model, namely, the one based on MATLAB's SimMechanics, and also to validate the results obtained in Section 5, the RP robot model was also created in SimMechanics. While the simulation time shown in Figures 7 and 8 are exactly same for the DeNOC and SimMechanics models, the CPU times taken by them were different. Table 2 shows the CPU times taken by different controllers while DeNOC and SimMechanics models were used. The comparison shows a clear advantage of the customised DeNOC-based force controller models over the generic SimMechanics-based block diagrams.

Table 2 CPU times for force control simulations of the RP and KUKA KR5 Arc robot

	Controller model	*Dynamics model*	*RP robot time (s)*	*KUKA robot time (s)*
A	Stiffness	DeNOC	1.63	36.39
	Control	SimMechanics	2.58	62.44
B	Impedance	DeNOC	1.09	18.14
	Control	SimMechanics	1.80	20.09
C	Admittance	DeNOC	2.84	18.52
	Control	SimMechanics	4.03	17.84
D	Hybrid	DeNOC	34.47	208.23
	Control	SimMechanics	39.00	314.23

Note: CPU platform: Intel i7 Processor, 2.667 GHz, MATLAB 2013a/Simulink v8.1/SimMechanics v4.2.

6 Control simulation of an industrial robot

Inspired by the better performance of the DeNOC-based dynamic modelling for control simulation of the RP robot, as explained in Section 5.5, control simulation of a 6-DOF industrial robot, namely, KUKA KR5 Arc was carried out to test how a control simulation, namely, stiffness controller performs in terms of both CPU time and accuracy of the results. The CAD model of the KUKA KR5 Arc was downloaded from KUKA Robotics Website (http://www.kuka-robotics.com/usa/en) and imported to Autodesk® Inventor. Mass properties of the robot were obtained from the CAD model with their density approximated so as to match the total mass of the real robot, i.e., KUKA KR5 Arc. Figure 9 shows the architecture of the KUKA KR5 Arc robot whose DH parameters and the inertia parameters are given in Tables 3 and 4, respectively. For the control simulation of KUKA KR5 Arc, a generic version of the DeNOC-based formulation, namely, acronym for recursive dynamic simulator (ReDySim) (Saha et al., 2013) was used. Moreover, both position and orientation controllers were used. The end-effector reaction force variation with the moving plane is shown in Figure 10(a), which is similar to Figure 7(a), as expected for the stiffness controller. Figure 10(b) shows the joint torques for the first three joints, whereas joint torques for the last three joints range within ±1.0 Nm with initial surge of ±8.0 Nm. Due to almost similar pattern of the plots they are not shown. The mean value of joint torque was 0.142 Nm, –0.534 Nm, and 0.0 Nm for

joints 4, 5 and 6, respectively. Figure 11(a) shows the end-effector position variation for 1 s of simulation, which shows that the end-effector moves in the sagittal plane along with the plane once the contact is established. The orientation of the end-effector is maintained precisely, as can be seen in Figure 11(b). Various matrices associated to the stiffness controller were taken as:

$$\begin{aligned} \mathbf{K}_{p\psi} &= \text{diag.}[1, 1, 1]\tfrac{\text{kNm}}{\text{rad}}, \mathbf{K}_{pp} = \text{diag.}[30, 1, 1]\tfrac{\text{kN}}{\text{m}}, \\ \mathbf{K}_{v} &= \text{diag.}[111111]\tfrac{\text{kNms}}{\text{rad}}, \\ \textit{and } \mathbf{K}_{F\psi} &= \text{diag.}[000]\tfrac{\text{rad.}}{\text{Nm}}, \mathbf{K}_{Fp} = \text{diag.}[0.0111]\tfrac{\text{m}}{\text{N}} \end{aligned} \tag{40}$$

In order to validate and compare the results with another dynamic model, namely, the SimMechanics, the Inventor model was also imported to SimMechanics environment using SimMechanics CAD translator tool CAD translator website (http://www.mathworks.com/products/simmechanics/download cad.html), where the joint constraints, sensors and actuators blocks were attached. The completed robot model in SimMechanics was then integrated to the controllers and plots for end-effector forces were obtained. Figure 12 shows the simulation model of KUKA KR5 Arc and the moving plane in the MATLAB/SimMechanics environment. The results showed a close match with those obtained using the DeNOC-based ReDySim shown in Figure 10 and Figure 11. The CPU time taken by the DeNOC-based ReDySim and SimMechanics for 0.5 s of simulation are given in Table 2. Hence, there is a clear advantage of using customised dynamic model based on say, the DeNOC-based modelling. Note here that other control simulations, i.e., impedance, admittance, and hybrid, were also performed for the KUKA KR5 Arc robot. They are not discussed here mainly to avoid the repetitions of similar steps as done for the stiffness controller. Control models for KUKA KR5 Arc robot can be downloaded from website (http://www.web.iitd.ac.in/saha/todownload/ieee-tro-force-control.zip).

Figure 9 Architecture of KUKA KR5 arc robot

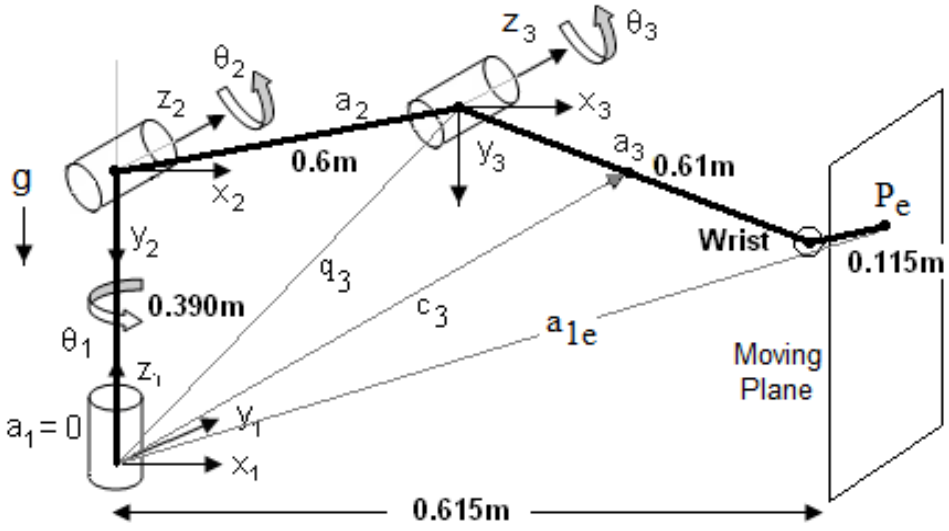


Table 3 DH Parameters of KUKA KR5 Arc robot

Link i	*Joint offset* $b_i(m)$	*Joint angle* $\theta_i(rad.)$	*Link length* $\alpha_i(m)$	*Twist angle* $\alpha_i(rad.)$
1	0.4	$\theta_1(JV^*)$	0.18	$\pi/2$
2	0	$\theta_2(JV)$	0/6	0
3	0	$\theta_3(JV)$	0.12	$\pi/2$
4	0.62	$\theta_4(JV)$	0	$\pi/2$
5	0	$\theta_5(JV)$	0	$-\pi/2$
6	0.115	$\theta_6(JV)$	0	0

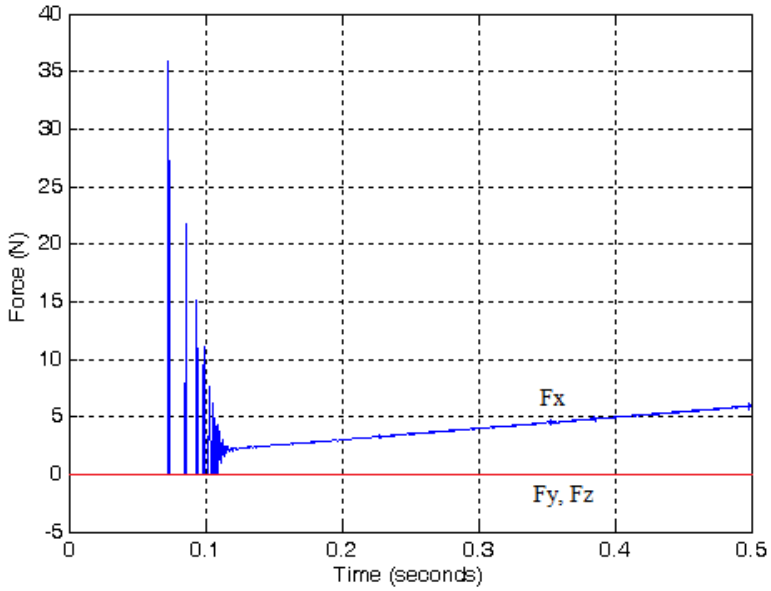
Note: *JV – joint variable.

Table 4 Assumed inertia parameters of KUKA KR5 Arc robot

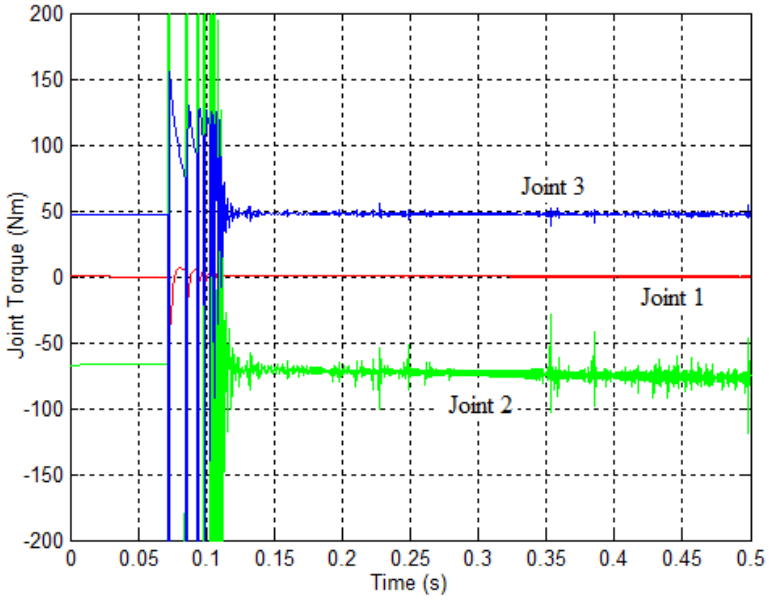
Link i	*Mass* $m_i(kg)$	*CG location* $\mathbf{c}_i(m)$	*Inertia tensor* $\mathbf{I}_{ci}(kgm^2)$
1	26.980	$\begin{bmatrix} 0.089 \\ 0.006 \\ 0.323 \end{bmatrix}$	$\begin{bmatrix} 0.322 & -0.018 & -0.145 \\ -0.018 & 0.467 & -0.014 \\ -0.145 & -0.014 & 0.478 \end{bmatrix}$
2	15.920	$\begin{bmatrix} 0.447 \\ -0.174 \\ 0.389 \end{bmatrix}$	$\begin{bmatrix} 0.541 & 0.0 & -0.005 \\ 0.0 & 0.552 & 0.017 \\ -0.005 & 0.017 & 0.044 \end{bmatrix}$
3	25.852	$\begin{bmatrix} 0.868 \\ -0.008 \\ 0.356 \end{bmatrix}$	$\begin{bmatrix} 0.775 & -0.009 & 0.025 \\ -0.009 & 0.750 & 0.007 \\ 0.025 & 0.007 & 0.208 \end{bmatrix}$
4	4.088	$\begin{bmatrix} 0.899 \\ 0.008 \\ -0.110 \end{bmatrix}$	$\begin{bmatrix} 0.010 & 0.002 & 0.0 \\ 0.002 & 0.020 & 0.0 \\ 0.0 & 0.0 & 0.024 \end{bmatrix}$
5	1.615	$\begin{bmatrix} 0.9 \\ 0.010 \\ -0.252 \end{bmatrix}$	$\begin{bmatrix} 0.002 & 0.0 & 0.0 \\ 0.0 & 0.004 & 0.0 \\ 0.0 & 0.0 & 0.004 \end{bmatrix}$
6	0.016	$\begin{bmatrix} 0.9 \\ 0.0 \\ -0.330 \end{bmatrix}$	$\begin{bmatrix} 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \end{bmatrix}$

Notes: Mass of the base: 52.402 kg, Total mass of the robot ≈127.0 kg. The vector $\mathbf{c}_i$ is for robot at $\boldsymbol{\theta} = [0, 0, 90, 0, 0, 0]^T$.

Figure 10 Performance of KUKA KR5 Arc robot with stiffness controller, (a) end-effector force (b) joint torques (see online version for colours)

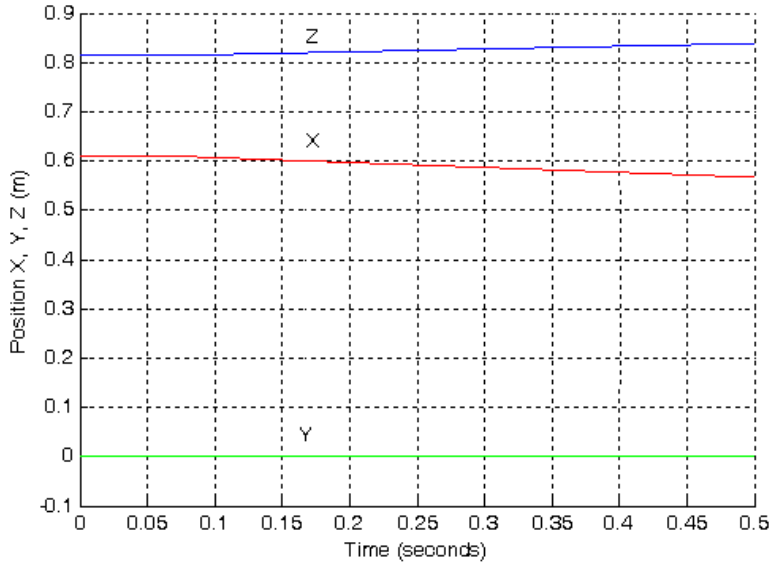


(a)

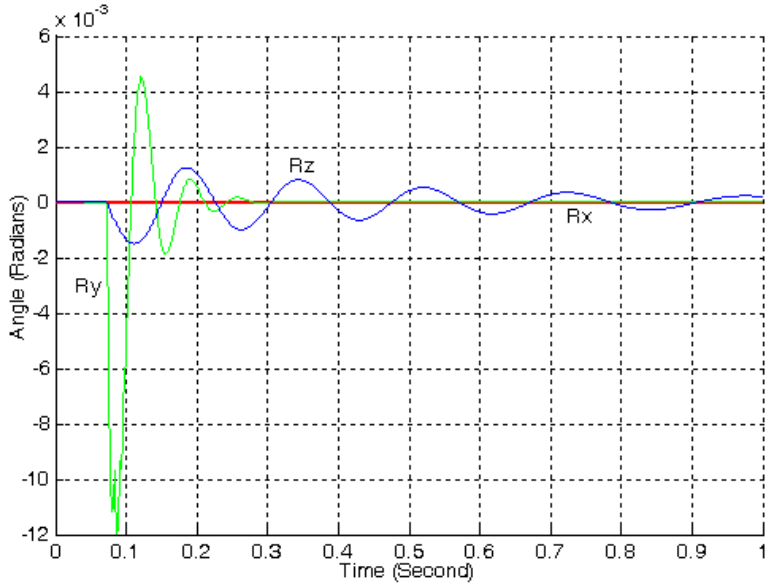


(b)

Figure 11 End-effector pose of KUKA KR5 Arc robot with stiffness controller, (a) end-effector position variation with time (b) end-effector orientation variation with time (see online version for colours)

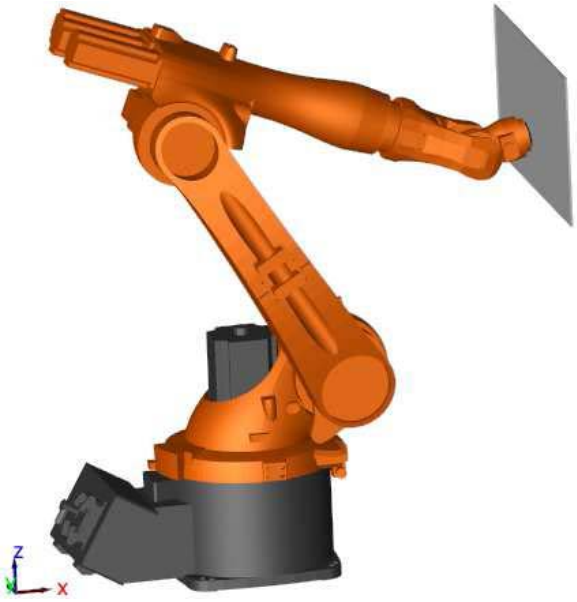


(a)



(b)

Figure 12 Simulation model of KUKA KR5 in the MATLAB/SimMechanics environment (see online version for colours)



7 Conclusions

This paper presents a framework for a test bench:

1 to test several force control algorithms say, to compare their performances

2 to tune their controller gains

3 to test a new control or dynamic algorithm by replacing the appropriate blocks

4 to use as a good teaching/learning tool for force control algorithms.

Such a comprehensive tool, which can be downloaded from website (http://www.web.iitd.ac.in/saha/todownload/ieee-tro-forcecontrol.zip), is generally not found in the literature. Hence, this is considered as the key contribution of this paper. In order to develop such a tool, the DeNOC-based dynamic modelling technique was used resulting in faster simulation compared to those based on MATLAB. This suggests that more and more complex controllers with complex robotic systems can be simulated using DeNOC matrices with relative ease.

8 The robot Jacobian

The Jacobian of a robot is a $6 \times n$ matrix that relates the Cartesian velocities of the end-effector, i.e., the twist denoted by $\mathbf{t}_e$, with the joint-rates $\dot{\boldsymbol{\theta}}$ defined after (5) as:

$$\mathbf{t}_e = \mathbf{J}\dot{\boldsymbol{\theta}} \tag{41}$$

where the matrix $\mathbf{J}$ is evaluated using (5), (6), and (7). Note that the end-effector being the n^{th} link of the robot its end-point velocity can be easily derived from the last block row of (5) as:

$$\mathbf{t}_e = \mathbf{R}_n \mathbf{t}_n \tag{42}$$

where $\mathbf{R}_n \equiv \begin{bmatrix} \mathbf{1} & \mathbf{O} \\ \mathbf{r}_n \times \mathbf{1} & \mathbf{1} \end{bmatrix}$ and $\mathbf{t}_n \equiv \mathbf{N}_{ln}\mathbf{N}_d\dot{\boldsymbol{\theta}}$.

In (42), $\mathbf{r}_n \times \mathbf{1}$ is the 3×3 cross-product matrix associated with the vector $\mathbf{r}_n$, as indicated in Figure 1. Moreover, the $6 \times 6n$ matrix $\mathbf{N}_{ln} \equiv [\mathbf{B}_{n1}\ \mathbf{B}_{n2}\cdots\mathbf{1}]$ is nothing but the last block row of the matrix $\mathbf{N}_l$ of (6). Comparing this to (42):

$$\mathbf{J} \equiv \mathbf{R}_n \mathbf{N}_{ln} \mathbf{N}_d \tag{43}$$

Expansion of (43) in terms of the three-dimensional vectors can then be expressed as:

$$\mathbf{J} = \begin{bmatrix} \mathbf{J}_\omega \\ \mathbf{J}_v \end{bmatrix} = \begin{bmatrix} \mathbf{e}_1 & \mathbf{e}_2 & \dots & \mathbf{e}_n \\ \mathbf{e}_1 \times \mathbf{a}_{1e} & \mathbf{e}_2 \times \mathbf{a}_{2e} & \dots & \mathbf{e}_n \times \mathbf{a}_{ne} \end{bmatrix} \tag{44}$$

which is available in any text book on robotics, e.g., in Saha (2014). Here, $\mathbf{a}_{ie}$ is the vector joining i^{th} joint to the end-effector, as indicated in Figure 1. The vectors $\mathbf{e}_i$ and $\mathbf{a}_{ie}$ can be easily obtained using forward kinematic transformation matrices and some vector manipulations.

Acknowledgements

The reported research was carried out under a project 'Setting up of Program for Autonomous Robotics' sponsored by BARC/BRNS, Mumbai, India. Financial support provided to the first author from the project is sincerely acknowledged. The authors wish to acknowledge the help of Mr. Majid Hameed Koul for integrating the ReDySim dynamics to the MATLAB's Simulink.

References

Agarwal, A., Shah, S.V., Bandyopadhyay, S. and Saha, S.K. (2013) 'Dynamics of serial kinematic chains with large number of degrees-of-freedom', *Int. J. of Multibody System Dynamics*, Vol. 32, No. 3, pp.273–298.

Angeles, J. (2003) *Fundamentals of Robotic Mechanical Systems: Theory, Methods, and Algorithms*, Springer-Verlag, New York.

Angeles, J. and Lee, S. (1988) 'The formulation of dynamical equations of holonomic mechanical systems using a natural orthogonal complement', *ASME J. of Applied Mechanics*, Vol. 55, No. 1, pp.243–244.

Cetinkunt, S. and Book, W.J. (1989) 'Symbolic modeling and dynamic simulation of robotic manipulators with compliant links and joints', *Robotics and Computer-Integrated Manufacturing, Special Issue on Simulation Software for Robotics*, Vol. 5, No. 4, pp.301–310.

Craig, J.J. (2004) *Introduction to Robotics: Mechanics and Control*, Prentice Hall, New Jersey.

Dallali, H., Mosadeghzad, M., Medrano-Cerda, G.A., Docquier, N., Kormushev, P., Tsagarakis, N., Li, Z. and Caldwell, D. (2013) 'Development of a dynamic simulator for a compliant humanoid robot based on a symbolic multibody approach', *Proc. of IEEE Int. Conf. on Mechatronics (ICM)*, pp.598–603.

Dormand, J.R. and Prince, P.J. (1980) 'A family of embedded Runge-Kutta formulae', *J. of Computational and Applied Mathematics*, Vol. 6, No. 1, pp.19–26.

Featherstone, R. and Fijany, A. (1999) 'A technique for analyzing constrained rigid-body systems and its application to the constraint force algorithm', *IEEE Transactions on Robotics Automation*, Vol. 15, No. 6, pp.1140–1144.

Khatib, O. (1987) 'A unified approach for motion and force control of robot manipulators: the operational space formulation', *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 1, pp.43–53.

Koul, M.H., Shah, S.V., Saha, S.K. and Manivannan, M. (2013) 'Reduced-order forward dynamics of multiclosed-loop systems', *Int. J. of Multibody System Dynamics*, Vol. 31, No. 4, pp.451–476.

KUKA Robotics Website [online] http://www.kukarobotics.com/usa/en (accessed 05 March2014).

Kumar, K.G. and Pathak, P.M. (2013) 'Dynamic modelling and simulation of a four legged jumping robot with compliant legs', *Robotics and Autonomous Systems*, Vol. 61, No. 3, pp.221–228.

Luh, J., Walker, M.W. and Paul, R.P.C. (1980) 'On-line computational scheme for mechanical manipulators', *Journal of Dynamic Systems, Measurement, and Control*, Vol. 102, No. 2, pp.69–76.

Mason, M. (1981) 'Compliance and force control for computer controlled manipulators', *IEEE Trans. on Sys. Man. Cyber.*, Vol. SMC-11, No. 6, pp.418–432.

Raibert, M.H. and Criag, J.J. (1981) 'Hybrid position/force control of manipulators', *ASME Journal of Dynamics System Measurement and Control*, Vol. 103, No. 2, pp.126–133.

Roberts, R.K., Paul, R.P. and Hillberg, B.M. (1985) 'The effect of wrist force sensor stiffness on the control of robot manipulators', *Proc. of IEEE Int. Conf. on Robotics and Automation, Louis*, pp.269–274.

Saha, S.K. (1997) 'A decomposition of the manipulator inertia matrix', *IEEE Transactions on Robotics and Automation*, Vol. 13, No. 2, pp.301–304.

Saha, S.K. (1999) 'Dynamics of serial multibody systems using the decoupled natural orthogonal complement matrices', *Int. J. of Applied Mechanics, ASME*, Vol. 66, No. 4, pp.986–996.

Saha, S.K. (2003) 'Simulation of industrial manipulators based on the UDU^T decomposition of inertia matrix', *Int. J. of Multibody System Dynamics*, Vol. 9, No. 1, pp.63–85.

Saha, S.K. (2014) *Introduction to Robotics*, 2nd ed., Mc-Graw Hill, New Delhi.

Saha, S.K., Shah, S.V. and Nandihal, P.V. (2013) 'Evolution of the DeNOC based dynamic modelling for multibody systems', *Open Access J. of Mechanical Sciences*, Vol. 4, No. 1, pp.1–20.

Salisbury, K.J. (1980) 'Active stiffness control of a manipulator in Cartesian coordinates', *Proc. of 19th IEEE Int. Conf. on Decision and Control, Albuquerque*, pp.95–100.

Schutter, J.D. and Brussel, H.V. (1988) 'Compliant robot motion II, a control approach based on external control loops', *Int. J. of Robotics Research*, Vol. 7, No. 4, pp.18–33.

Seraji, H. (1994) 'Adaptive admittance control: an approach to explicit force control in compliant motion', *Proc. of IEEE Int. Conf. on Robotics and Automation*, pp.2705–2712.

Shah, S.V., Saha, S.K. and Dutt, J.K. (2012) 'Modular framework for dynamic modeling and analyses of legged robots', *Mechanism and Machine Theory*, Vol. 49, pp.234–255.

SimMechanics CAD Translator Website [online] http://www.mathworks.com/products/simmechanics/downloadcad.html (accessed 05 March2014)

Udai, A.D. and Saha, S.K. (2012) 'Simulation of force control algorithms for serial robots', *Proc. of IEEE/SICE Int. Symp. on Syst. Integration (SII)*, Fukuoka, Japan, pp.481–486.

Website to download the MATLAB/Simulink models presented in this paper [online] http://http://www.dayalsoft.com/todownload/sim-force-control.zip (accessed 25 July 2017).

Whitney, D.E. (1987) 'Historical perspective and state of the art in robot force control', *Int. J. of Robotics Research*, Vol. 6, No. 1, pp.3–14.

Wood, G.D. and Kennedy, D.C. (2003) *Simulating Mechanical Systems in Simulink With Simmechanics*, Technical Report.

Zeng, G. and Hemani, A. (1997) 'An overview of robot force control', *Robotica*, Vol. 15, No. 5, pp.473–482.

Zhang, X. and Ivlev, O. (2010) 'Simulation of interaction tasks for pneumatic soft robots using simmechanics', *Proc. of 19th IEEE Int. Workshop on Robotics in Alpe-Adria-Danube Region (RAAD)*, pp.149–154.

Appendix

Gravity torque

Given the position vector of the i^{th} joint as $\mathbf{o}_i$ and that for the centre of mass be $\mathbf{c}_i$, the torque due to gravity at the i^{th} joint may be given by Udai et al. (2012) as:

$$\boldsymbol{\tau}_i = -\sum_{j=i}^{n} \left[\left(\mathbf{c}_j - \mathbf{o}_i \right) \times m_j \mathbf{g} \right]^T \mathbf{e}_i \tag{45}$$

where m_j is the mass of the link #j and $\mathbf{g}$ is the acceleration due to gravity. A gravity compensation vector of torque which is required to balance the robot against the gravitational forces may be formulated as:

$$\boldsymbol{\tau}_g \equiv \left[\tau_1, \tau_2, \cdots, \tau_n \right]^T \tag{46}$$

where $\boldsymbol{\tau}_i$ is given by (45), which is used in Section 3 for control simulation.

DH parameters

In the literature, there are many variations in the definition of DH parameters, e.g., in Craig (2004); Saha (2014), and others. The one used here is from Saha (2014). Referring to Figure 1, the manipulator consists of $n + 1$ links, namely, the base #0 and n moving links denoted as #1, #2, ..., #n, coupled by n pairs numbered as 1, 2, …., n, respectively. As shown in Figure 1, the i^{th} pair couples the $(i - 1)^{st}$ link with the i^{th} one. Moreover the coordinate system X_i, Y_i, Z_i is attached to the $(i - 1)^{st}$ link. The DH parameters are then defined as:

1 Z_i is the axis of the i^{th} pair. Its positive direction can be chosen arbitrarily.

2 X_i is defined as the common perpendicular to Z_{i-1} and Z_i, directed from the former to the latter. The origin of the i^{th} frame, O_i, is the point where X_i intersects Z_i. If these two axes intersect, the positive direction of X_i is chosen arbitrary. The origin, O_i, coincides with the origin of the $(i - 1)^{st}$ frame, O_{i-1}.

3 the distance between Z_i and Z_{i+1} is defined as a_i, which is non-negative.

4 the Z_i coordinate of the intersection of the X_{i+1} axis with Z_i is defined as b_i, which thus can be either positive or negative. For prismatic joint, b_i is variable.

5 the angle between Z_i and Z_{i+1} is defined as α_i, and is measured about the positive direction of X_{i+1}

6 the angle between X_i and X_{i+1} is defined as θ_i, and is measured about the positive direction of Z_i. For revolute joint, θ_i is variable.

Since no $(n + 1)^{st}$ link exists the above definition do not apply to the $(n + 1)^{st}$ frame and it can be chosen at will.