Arab J Sci Eng (2017) 42:1841–1852
DOI 10.1007/s13369-016-2324-y

RESEARCH ARTICLE - MECHANICAL ENGINEERING

Dynamics and Control of a Robotic Arm Having Four Links

H. M. Al-Qahtani[1] · Amin A. Mohammed[1] · M. Sunar[1]

Received: 10 March 2016 / Accepted: 29 September 2016 / Published online: 28 October 2016

Abstract The manipulator control is an important problem in robotics. To work out this problem, a correct dynamic model for the robot manipulator must be in hand. Hence, this work first presents the dynamic model of an existing 4-DOF robot manipulator based on the Euler–Lagrange principle, utilizing the body Jacobian of each link and the generalized inertia matrix. Furthermore, essential properties of the dynamic model are analyzed for the purpose of control. Then, a PID controller is designed to control the position of the robot by decoupling the dynamic model. To achieve a good performance, the differential evolution algorithm is used for the selection of parameters of the PID controller. Feedback linearization scheme is also utilized for the position and trajectory tracking control of the manipulator. The obtained results reveal that the PID control coupled with the differential evolution algorithm and the feedback linearization control enhance the performance of the robotic manipulator. It is also found out that increasing masses of manipulator links do not affect the performance of the PID position control, but higher control torques are required in these cases.

Keywords Robot control · PID · Differential evolution · Feedback linearization

1 Introduction

Obtaining the correct dynamic model of a robot is a necessary step for its control [1]. In other words, to achieve a proper control, a valid dynamic model must be obtained first. In their approach for modeling and identification of a high-performance robot control, Kostic et al. [2] highlighted a procedure for obtaining kinematic and dynamic models of a robot for the control design. The procedure involves deduction of the robot's kinematic and dynamic models, estimation of the model parameters experimentally, validation of the model and identification of the remaining robot dynamics, which should not be ignored if robustness and high-performance robot operation are required. Finally, a straightforward and efficient estimation of parameters of the rigid-body dynamics model that includes friction effects is suggested. With these additional dynamics available, more advanced feedback control designs become possible.

In [3], Long et al. showed the model and simulation of a test system for an articulated robotic arm. They designed the test system of smaller volume for the articulated robotic arm and then developed the mathematical model of the system. The mathematical model of the robot allowed for the MATLAB numerical simulation and PID optimization. According to the results, the system carries advantages of strong driving capability and high positioning accuracy for the hydraulic and motor drive systems.

Beside their uses in various industries [4–6], robotics has also applications in biologic and medical fields, where it is used to help disabled people to perform necessary living activities. Methods for kinematic modeling of robotics and biological systems were given in [7]. Strategies were developed to automatically produce kinematic models for the movements of biological and robotic systems. Particular attention was given to build a wheelchair-mounted robotic arm to be utilized by children with quadriplegia. The movement of the system was approximated with an open kinematic chain. Two identification methods were adopted to automatically generate kinematic models and tested with a planar RR robotic arm. While the first method requires the elimination

✉ H. M. Al-Qahtani
qahtanih@kfupm.edu.sa

[1] Mechanical Engineering Department, King Fahd University of Petroleum & Minerals, Dhahran 31261, Kingdom of Saudi Arabia

of measurable displacement variables, the second one tries to estimate changes in these variables.

After modeling the robotics system, a controller is needed to control the motion of the system. The control of robotic manipulators is unlike that of other industrial equipment, because it is accompanied with a big number of separately controllable mechanical axes [8]. Robust control of a two-link manipulator was introduced by Yadav and Singh [9], where the Lagrange–Euler method was used to drive the manipulator dynamics with uncertainty. Two different robust control schemes, namely H_∞ and μ-synthesis, were used to control the robotic manipulator. Although both controllers were sufficient for stabilizing the manipulator in an effective manner, the μ-synthesis controller was noted to have superior robust performance. In [10], a fuzzy controller for the under-actuated manipulator was presented. The complex mathematics usually encountered with the nonlinear dynamic model of the manipulator was avoided, making the proposed method rather simple and computationally fast. For verification purposes, numerical simulation for the position control of a planar RRR robot was carried out.

Position control of a single-link robotic arm using a multi-loop PI controller was presented in [11]. In addition, the computed torque control (CTC) of the PUMA 560 robotic manipulator was investigated by Piltan et al. [12], based on the Simulink realization. The PD-CTC and PID-CTC control schemes were tested through step and ramp responses. A similar approach using the sliding mode control (SMC) was taken in [13]. A neural network (NN) self-tuning PID controller was proposed in [14]. It was shown via simulations that using the NN together with the conventional PID control improved the performance and reduced tracking errors significantly. Modeling and simulation of the nonlinear CTC in Simulink for an industrial robot to fully compensate the Coriolis and centripetal forces was introduced by Receanu [15]. A PD control strategy was employed to control the motion of the robot, whose dynamic model was based on the Euler–Lagrange method.

Dynamics modeling of an existing 4-DOF robotic arm [16] by the Euler–Lagrange method is carried out in this work. After the modeling, two control approaches are implemented for the control of joint angles (position control) of the arm, namely PID and feedback linearization (FBL) schemes. The FBL is also applied to the arm for the trajectory tracking control. Furthermore, link masses of the robotic arm are increased to investigate its effect on the PID position control performance. Results are reported and discussed.

2 Dynamics Model

In this section, dynamics model of the robotic arm [16] shown in Fig. 1 is derived based on the Euler–Lagrange approach.

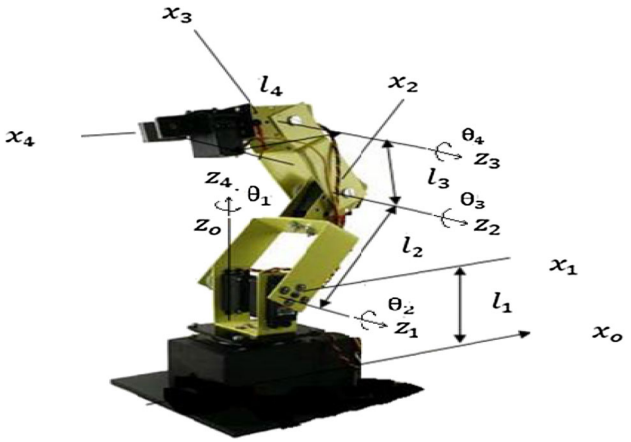

Fig. 1 Four-link robotic manipulator

Links of the robot are made up of aluminum. The robot dimensions shown in this figure are: $l_1 = 11.5\,\text{cm}, l_2 = 12\,\text{cm}$ and $l_3 = l_4 = 9\,\text{cm}$. Widths and thicknesses of links are taken as 5 cm and 1 mm, respectively. Mass of each servo at the joints is given as 55.2 g. By defining the potential V and kinetic K energies of the system, and the Lagrangian L as the difference between the kinetic and potential energies, the Euler–Lagrange equation (1) below can be used to determine the dynamics model of the robotic manipulator

$$\frac{\mathrm{d}}{\mathrm{d}t}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = Q \tag{1}$$

where Q stands for the generalized force. Since there are 4 links ($n = 4$) and hence 4 degrees of freedom, there are 4 generalized coordinates (q_i) in (1) leading to 4 equations. To calculate the kinetic energy of the robot, we sum the kinetic energy of each link. Thus, the total kinetic energy becomes

$$K(\theta, \dot{\theta}) \sum_{i=1}^{n} K_i(\theta, \dot{\theta}) = \frac{1}{2}\dot{\theta}^{\mathrm{T}} D(\theta)\dot{\theta} \tag{2}$$

where $D(\theta) \in R^{n\times n}$ is the manipulator inertia matrix defined as

$$D(\theta) = \sum_{i=1}^{n} J_i^T M_i J_i \tag{3}$$

where J_i and M_i refer to Jacobian and generalized mass matrices for the ith link. Next, we need to know the potential energy of the robot, for which we introduce h_i as the height for the mass center of the ith link. The potential energy is then given by

$$V_i(\theta) = m_i g h_i(\theta). \tag{4}$$

Thus, the Lagrangian becomes

$$L(\theta,\dot{\theta}) = \sum_{i=1}^{n} (K_i(\theta,\dot{\theta}) - V_i(\theta)) = \frac{1}{2}\dot{\theta}^T D(\theta)\dot{\theta} - V(\theta). \tag{5}$$

For control purposes, it is common and more practical to rewrite the Euler–Lagrangian dynamic model of the robot in compact or matrix form as follows

$$D(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) = \tau \tag{6}$$

where τ is the vector of actuator torques, $g(\theta)$ is the vector of gravity forces and the matrix $C(\theta, \dot{\theta})$ is called the Coriolis term for the manipulator given by

$$C(\theta,\dot{\theta}) = \Gamma_{ijk}\dot{\theta}_k = \frac{1}{2}\left(\frac{\partial D_{ij}}{\partial \theta_k} + \frac{\partial D_{ik}}{\partial \theta_j} - \frac{\partial D_{kj}}{\partial \theta_i}\right)\dot{\theta}_k. \tag{7}$$

In Eqs. (6) and (7), D is a square matrix of the size 4×4, C, g and τ are column vectors of the size 4×1. To find the inertia matrix D using Eq. (3), we need to know the body Jacobian for every link. By attaching a coordinate frame at the mass center of each link, the generalized inertia matrix M_i simplifies to the following diagonal matrix

$$M_i = \begin{bmatrix} m_i & 0 & 0 & 0 & 0 & 0 \\ 0 & m_i & 0 & 0 & 0 & 0 \\ 0 & 0 & m_i & 0 & 0 & 0 \\ 0 & 0 & 0 & I_{xi} & 0 & 0 \\ 0 & 0 & 0 & 0 & I_{yi} & 0 \\ 0 & 0 & 0 & 0 & 0 & I_{zi} \end{bmatrix}. \tag{8}$$

Expanding Eq. (3), we get

$$D(\theta) = J_1^T M_1 J_1 + J_2^T M_2 J_2 + J_3^T M_3 J_3 + J_4^T M_4 J_4 \tag{9}$$

where the Jacobian matrix J corresponding to each link is given by

$$J_1 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, J_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ r_2 c_2 & 0 & 0 & 0 \\ 0 & r_2 & 0 & 0 \\ s_2 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ c_2 & 0 & 0 & 0 \end{bmatrix},$$

$$J_3 = \begin{bmatrix} 0 & l_2 s_3 & 0 & 0 \\ r_3 c_{23} + l_2 c_2 & 0 & 0 & 0 \\ 0 & r_3 + l_2 c_3 & r_3 & 0 \\ s_{23} & 0 & 0 & 0 \\ 0 & -1 & -1 & 0 \\ c_{23} & 0 & 0 & 0 \end{bmatrix}$$

$$J_4 = \begin{bmatrix} 0 & l_2 s_{34} + l_3 s_4 & l_3 s_4 & 0 \\ l_3 c_{23} + l_2 c_2 + r_4 c_{234} & 0 & 0 & 0 \\ 0 & r_4 + l_2 c_{34} + l_3 c_4 & r_4 + l_3 c_4 & r_4 \\ s_{234} & 0 & 0 & 0 \\ 0 & -1 & -1 & -1 \\ c_{234} & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

where r_i stands for distance of the ith link to its mass center, $s_i = \sin\theta_i, c_i = \cos\theta_i, s_{ij} = \sin(\theta_i + \theta_j), s_{ijk} = \sin(\theta_i + \theta_j + \theta_k)$ and so on. Elements of the symmetric inertia matrix D are computed as follows

$$\begin{aligned}
D_{11} &= I_{x2}s_2^2 + I_{x3}s_{23}^2 + I_{x4}s_{234}^2 + I_{z1} + I_{z2}c_2^2 \\
&\quad + I_{z3}c_{23}^2 + I_{z4}c_{234}^2 + m_2 r_2^2 c_2^2 + m_3(r_3 c_{23} + l_2 c_2)^2 \\
&\quad + m_4(l_3 c_{23} + l_2 c_2 + r_4 c_{234})^2 \\
D_{12} &= 0,\ D_{13} = 0,\ D_{14} = 0 \\
D_{22} &= I_{y2} + I_{y3} + I_{y4} + m_2 r_2^2 + m_3 l_2^2 s_3^2 + m_3(r_3 + l_2 c_3)^2 \\
&\quad + m_4(l_2 s_{23} + l_3 s_4)^2 + m_4(r_4 + l_2 c_{34} + l_3 c_4)^2 \\
D_{23} &= I_{y3} + I_{y4} + m_3 r_3(r_3 + l_2 c_3) + m_4 l_3 s_4(l_2 s_{23} + l_3 s_4) \\
&\quad + m_4(r_4 + l_3 c_4)(r_4 + l_2 c_{34} + l_3 c_4) \\
D_{24} &= I_{y4} + m_4 r_4(r_4 + l_2 c_{34} + l_3 c_4) \\
D_{33} &= I_{y3} + I_{y4} + m_3 r_3^2 + m_4 l_3^2 s_4^2 + m_4(r_4 + l_3 c_4)^2 \\
D_{34} &= I_{y4} + m_4 r_4(r_4 + l_3 c_4) \\
D_{44} &= I_{y4} + m_4 r_4^2.
\end{aligned} \tag{11}$$

Having the inertia matrix D in hand, the Coriolis and centrifugal forces are computed directly from Eq. (7). Next, components of the gravity forces on the robotic manipulator can be derived from

$$g(\theta) = \frac{\partial V}{\partial \theta_i} \tag{12}$$

where the potential energy V is given by

$$V(\theta) = g(m_1 h_1(\theta) + m_2 h_2(\theta) + m_3 h_3(\theta) + m_4 h_4(\theta)) \tag{13}$$

where h_i is the height of the mass center of the i^{th} link given by

$$\begin{aligned}
h_1(\theta) &= r_1, h_2(\theta) = l_1 + r_2 \sin\theta_2, \\
h_3(\theta) &= l_1 + l_2 \sin\theta_2 + r_3 \sin(\theta_2 + \theta_3) \\
h_4(\theta) &= l_1 + l_2 \sin\theta_2 + l_3 \sin(\theta_2 + \theta_3) \\
&\quad + r_4 \sin(\theta_2 + \theta_3 + \theta_4).
\end{aligned}$$

According to the derived model with the symmetric inertia matrix obtained above, there is no dynamic interaction due to acceleration between the first link and the other three planar links. The only interaction is due to the nonlinear Coriolis and centrifugal forces.

3 PID Control

The PID control law is given by

$$\tau = K_P\tilde{\theta} + K_I \int \tilde{\theta}\mathrm{d}t + K_D\dot{\tilde{\theta}} \tag{14}$$

where symmetric positive-definite matrices K_P, K_I and K_D are called the proportional, integral and derivative gain matrices, respectively, and $\tilde{\theta}$ denotes deviation from the desired angle θ_d. Recently, PID controllers are used in the control of most of the industrial robotic manipulators. However, the tuning methodology for the PID controllers, that is, the methodology to choose suitable values for K_P, K_I and K_D, is far from trivial. The common approach is to assign some suitable values to them while observing the corresponding response and to continue until obtaining satisfactory performance to some extent. But, this procedure is not effective and time-consuming especially when the degree of freedom is high as presented below in Sect. 3.1.

The dynamic model of an n-DOF robot was given in previous section by Eq. (6). The objective here is to introduce a PID controller given by the above formula (14), to the robot whose model was given by Eq. (6). The integral action of the PID control law (11) introduces an additional state variable that is denoted by ξ whose time derivative is $\dot{\xi} = \tilde{\theta}$. The PID control law is then expressed as

$$\begin{aligned} \tau &= K_P\tilde{\theta} + K_I\xi + K_D\dot{\tilde{\theta}} \\ \dot{\xi} &= \tilde{\theta} \end{aligned} \tag{15}$$

The equation of the closed-loop is obtained by inserting the control action τ from Eq. (15) in the dynamic model of the robotic manipulator of Eq. (6), i.e.,

$$\begin{aligned} &D(\theta)\ddot{\theta} + C(\theta,\dot{\theta})\dot{\theta} + g(\theta) = K_P\tilde{\theta} + K_I\xi + K_D\dot{\tilde{\theta}} \\ &\dot{\xi} = \tilde{\theta}, \end{aligned} \tag{16}$$

which may be written in state form as

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} \xi \\ \tilde{\theta} \\ \dot{\tilde{\theta}} \end{bmatrix} = \begin{bmatrix} \tilde{\theta} \\ \dot{\tilde{\theta}} \\ \ddot{\theta}_d - D(\theta)^{-1}[K_P\tilde{\theta} + K_I\xi + K_D\dot{\tilde{\theta}} - C(\theta,\dot{\theta})\dot{\theta} - g(\theta)] \end{bmatrix}. \tag{17}$$

At the equilibrium, $\tilde{\theta} = 0$ and $\dot{\tilde{\theta}} = 0$, which implies that $\theta = \theta_d$. Hence, the above equation yields

$$[\xi^T\ \tilde{\theta}^T\ \dot{\tilde{\theta}}^T] = [\xi^*\ 0\ 0] \tag{18}$$

where

$$\xi^* = K_I^{-1}[D(\theta_d)\ddot{\theta}_d + C(\theta_d,\dot{\theta}_d)\dot{\theta}_d + g(\theta_d)]$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix} \theta_d \\ \dot{\theta}_d \end{bmatrix} = \begin{bmatrix} \dot{\theta}_d \\ D(\theta_d)^{-1}[\tau - C(\theta_d,\dot{\theta}_d)\dot{\theta}_d - g(\theta_d)] \end{bmatrix}.$$

3.1 Control Cases

For moving the robot manipulator from one position to another, two cases are considered. For the first case, the home (initial) position is started with all the joint angles at zero degrees and the angles are then shifted by 30 degrees, which is set as the home position for the second case. All the joint angles are further rotated by additional 30 degrees for the final position of the second case. The four joint angles of the two cases are given below for the sake of clarity. Obtained results reveal that the controller performance is not affected by changing the initial and final configurations of the manipulator, leading to the conclusion that the controller can bring the robot between any arbitrary configurations within its workspace.

3.1.1 Case 1

Initial and final positions of angles θ_i and θ_d are taken as:

$$\theta_i = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \quad \theta_d = \begin{bmatrix} \pi/6 \\ \pi/6 \\ \pi/6 \\ \pi/6 \end{bmatrix}.$$

3.1.2 Case 2

Initial and final angles are taken as:

$$\theta_i = \begin{bmatrix} \pi/6 \\ \pi/6 \\ \pi/6 \\ \pi/6 \end{bmatrix}, \quad \theta_d = \begin{bmatrix} \pi/3 \\ \pi/3 \\ \pi/3 \\ \pi/3 \end{bmatrix}.$$

3.2 Manual Tuning

In what follows, manual tuning of the PID controller for the robotic manipulator corresponding to the above control cases is shown. The manual tuning is performed by trial and error, i.e., by changing the gains and observing the resulting response. All the gains are first set to zero; then, the proportional gain is increased until the output of the loop oscillates. This is followed by the adjustment in integral gain to minimize the steady-state error. At the end, the differential gain is tuned, together with the other gains, if required, until an acceptable response is obtained. The final PID tuning matrices are given below.

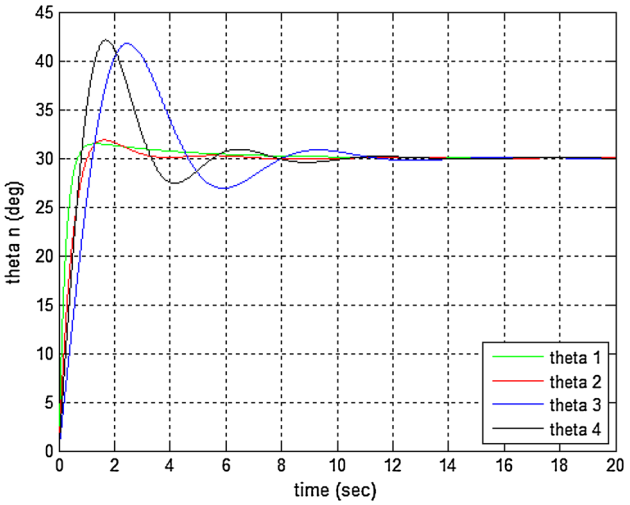

Fig. 2 Responses for PID control (manual tuning), Case 1

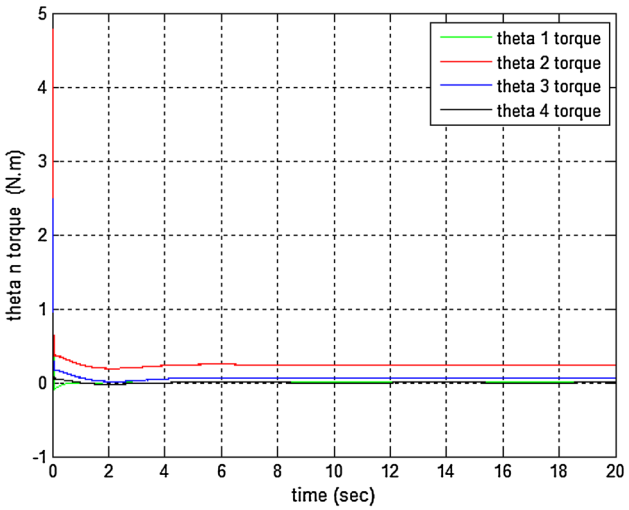

Fig. 3 Control signals for PID control (manual tuning), Case 1

$$K_P = \begin{bmatrix} 650 & 0 & 0 & 0 \\ 0 & 400 & 0 & 0 \\ 0 & 0 & 800 & 0 \\ 0 & 0 & 0 & 500 \end{bmatrix},$$

$$K_I = \begin{bmatrix} 180 & 0 & 0 & 0 \\ 0 & 350 & 0 & 0 \\ 0 & 0 & 903 & 0 \\ 0 & 0 & 0 & 730 \end{bmatrix},$$

$$K_D = \begin{bmatrix} 150 & 0 & 0 & 0 \\ 0 & 160 & 0 & 0 \\ 0 & 0 & 900 & 0 \\ 0 & 0 & 0 & 420 \end{bmatrix}.$$

Responses for angles θ_1 to θ_4 are given in degrees in Figs. 2 and 4 for control Cases 1 and 2 above. Although responses for θ_1 and θ_2 seem to be reasonable in both cases, but for θ_3 and θ_4, as these figures depict, responses are relatively poor in terms of both the overshoot and settling time. The settling time exceeds 10 sec, and the overshoot is around 42% for θ_3 in Figs. 2 and 4. Control signals (torques) are presented only for Case 1 in Fig. 3. Those for Case 2 are not shown due to their closeness to Case 1.

To overcome problems of high overshoot and settling time, the PID parameters must be properly tuned. Hence, the tuning algorithm of differential evolution (DE) is used in the next section for this purpose.

3.3 Differential Evolution Algorithm

This section summarizes the concept of the differential evolution algorithm used for tuning the PID parameters [17]. The DE algorithm is known to be a very effective global optimizer. In our case, the overall objective is to minimize the overshoot and settling time for the optimization scheme. The

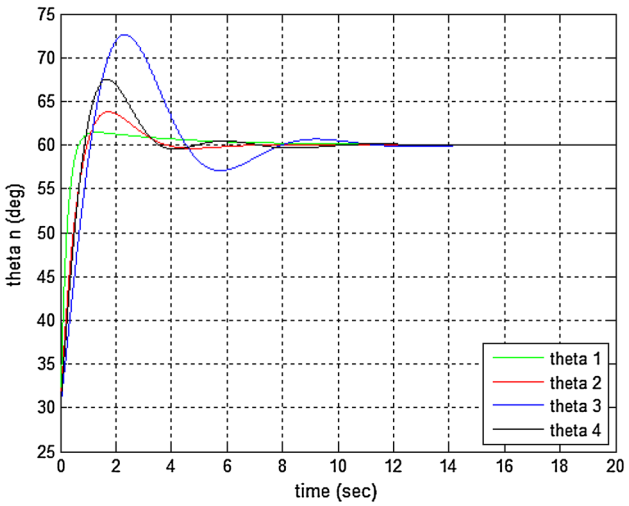

Fig. 4 Responses for PID control (manual tuning), Case 2

performance of the DE depends on three main operations: mutation, generation (reproduction) and selection. The mutation is the central operator of the DE and is the key process that makes the DE different from other evolutionary algorithms. The general structure of the DE algorithm is shown in Fig. 5 [18]. A simplified pseudocode is laid out as follows

1. Randomly initialize the parent population.
2. Calculate the objective function values $f(X_i)$ for all X_i.
3. Select three points from population and generate perturbed individual V_i.
4. Recombine each target vector X_i with the perturbed individual V_i generated in step 3 to construct a trial vector U_i.
5. Check whether each entry of the trial vector U_i is within the range.
6. Calculate the objective function value for the vector U_i.

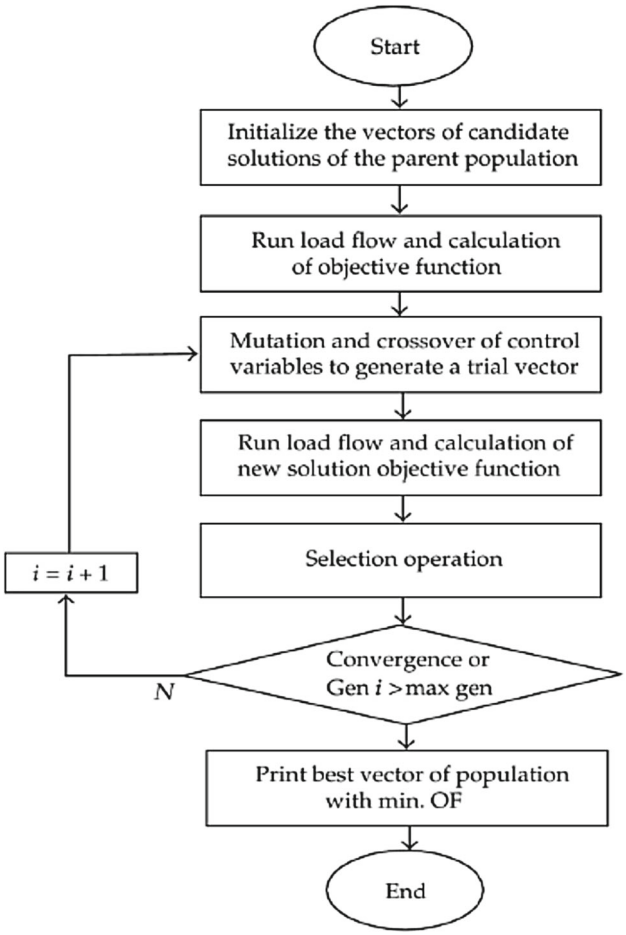

Fig. 5 Differential evolution algorithm [18]

7. Choose better of the two (function values at the target and trial points).
8. Check whether the convergence criterion is satisfied or not. If yes then stop, otherwise go to step 3.

Here the initial population is randomly chosen for the three controller gain matrices K_P, K_I and K_D, each of which is in the range of 0 to 10,000. The original population size is taken as $NP = 20$, generation size $NG = 10$, crossover factor $CR = 0.5$ and mutation factor $F = 0.5$. Referring to the manual tuning results, in some plots the settling time is high, while in others the overshoot is high, and thus, the objective function is selected such that both the overshoot and settling time are minimized in a balanced manner. Hence, the (overall) objective function to be minimized is constructed as $OF = 0.5 \times (OF_1 + OF_2)$, where OF_1 and OF_2 are individual objective functions for the maximum overshoot and settling time, respectively. The DE algorithm changes controller parameters randomly (but within a range) to find the best fitting function. In this case under investigation, the overshoot and settling time are given equal importance by

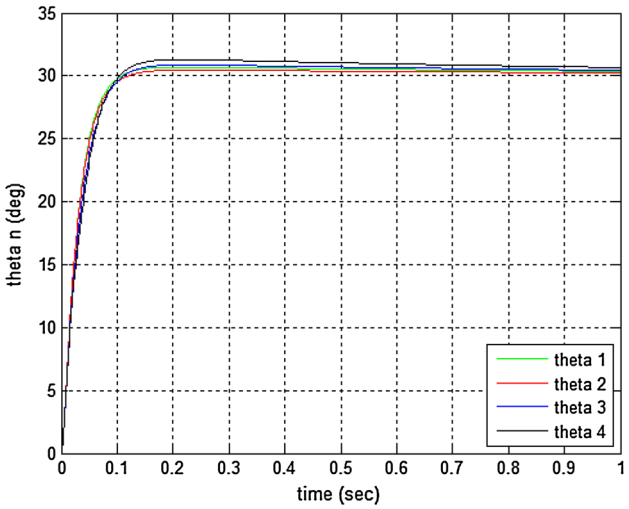

Fig. 6 Responses for PID control (DE tuning), Case 1

selecting the factor 0.5. Other response characteristics like rise time can also be included in the objective function, but it is observed from the manual tuning results that settling time and overshoot are notable to be minimized.

The DE algorithm is programmed to put the vector of the objective function in ascending order so that the first value in the resulting vector corresponds to the best parameter under the prescribed conditions. One limitation of this algorithm is that the resulting parameters for K_P, K_I and K_D are same for all the links, and hence, the response may not be optimal. Thus, the algorithm needs to be improved or other techniques might be required. The resulting controller matrices are found to be (in the form of positive-definite symmetric matrices)

$$K_P = \begin{bmatrix} 9721 & 0 & 0 & 0 \\ 0 & 9721 & 0 & 0 \\ 0 & 0 & 9721 & 0 \\ 0 & 0 & 0 & 9721 \end{bmatrix},$$

$$K_I = \begin{bmatrix} 8354 & 0 & 0 & 0 \\ 0 & 8354 & 0 & 0 \\ 0 & 0 & 8354 & 0 \\ 0 & 0 & 0 & 8354 \end{bmatrix},$$

$$K_D = \begin{bmatrix} 316 & 0 & 0 & 0 \\ 0 & 316 & 0 & 0 \\ 0 & 0 & 316 & 0 \\ 0 & 0 & 0 & 316 \end{bmatrix}.$$

Joint angle time-history results for the closed-loop are shown in Figs. 6 and 8. Figure 7 depicts time responses of joint control torques (signals) for only Case 1. Input torques for Case 2 follow similar trends. It is clear from these figures that the control torques for all the joints drive the links to their respected desired positions in very short periods of time with small overshoots. However, control inputs are relatively

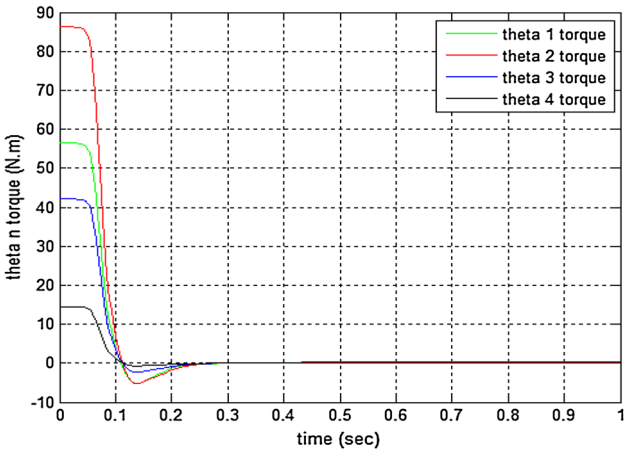

Fig. 7 Control signals for PID control (DE tuning), Case 1

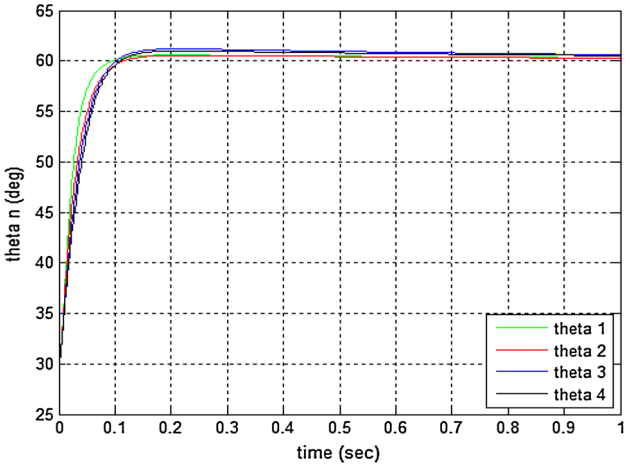

Fig. 8 Responses for PID control (DE tuning), Case 2

large, and hence, big servos may be required. On the other hand, the control effort of the manual tuning is small, but the obtained response is relatively poor. Thus, we need to bargain between the two requirements of control performance and effort.

3.4 Configuration Plots

For direct visualization, it is very useful to depict different configurations of the robot when explaining effects of the gravitational and external forces on the controller performance. Using the robotics toolbox for MATLAB [19,20], Figs. 9, 10 and 11 below are produced to display the robot configurations for Cases 1 and 2, whose initial and final joint angles are given before.

3.5 Analysis of Results

This section presents analysis of results using the PID control for the 4-DOF robotic manipulator. Since the robot is

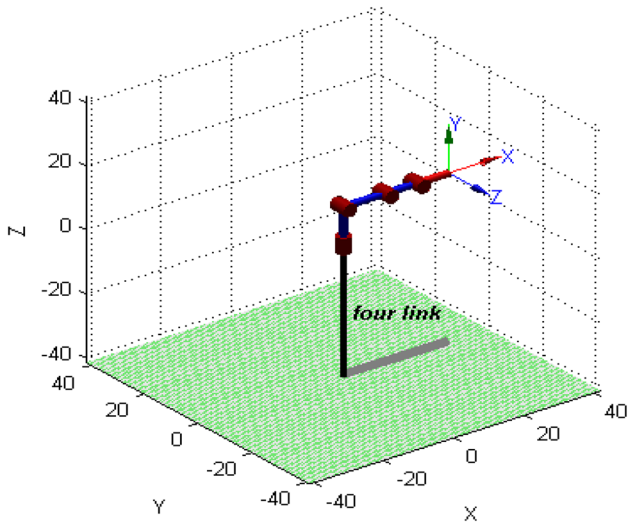

Fig. 9 Home/initial position, Case 1

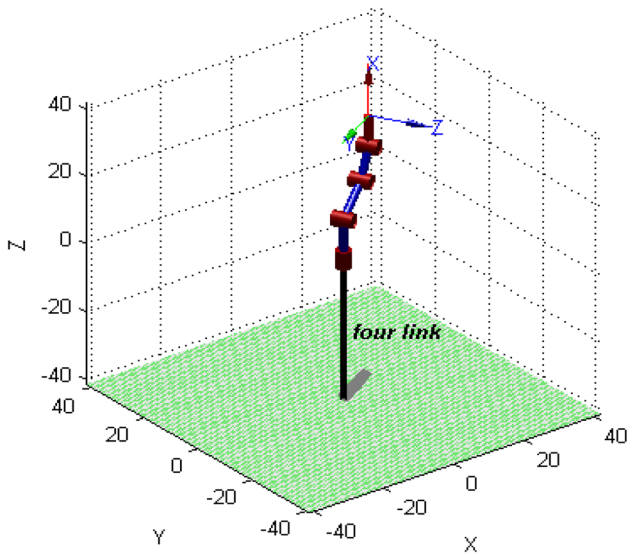

Fig. 10 Desired/final position, Case 1

targeted to move from one position to another for picking and placing purposes, different sets of initial and final positions are investigated. The time span was reduced from 20 seconds for the case of manual tuning to 1 second for that of the DE algorithm, which was largely sufficient for the convergence of the joint angles as the previous figures indicated. A comprehensive summary of the obtained results is listed in Table 1. It is clear from this table that the biggest overshoot occurring is 4.2763% in Case 1 corresponding to the fourth joint angle. The physical interpretation for this is that the fourth link is furthest from the origin, implying that effects of centrifugal and Coriolis forces are big as compared with the other links. So, in general, the larger the distance from the origin, the bigger the effect of the gravity on the controlled behavior. We can also see from this table that for the planar

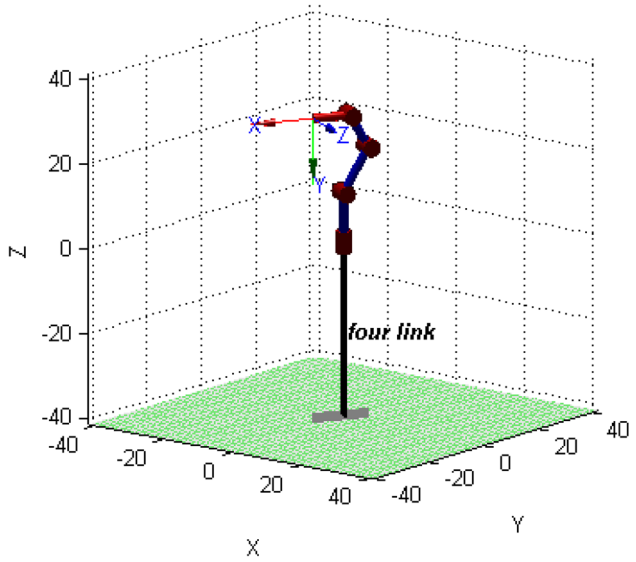

Fig. 11 Desired/final position, Case 2

Table 1 Summary of results

Case	θ_i	θ_f	Overshoot (%)	Settling time (s)	Rise time (s)
I	0	30	2.2904	0.3988	0.0562
	0	30	1.5257	0.0974	0.0582
	0	30	2.8776	0.6545	0.0631
	0	30	4.2763	1.0815	0.0638
II	30	60	0.9236	0.0749	0.0418
	30	60	0.8879	0.0952	0.0572
	30	60	1.9530	0.9791	0.0608
	30	60	1.6189	0.7946	0.0642

links 2 and 3 both the overshoot and settling time are proportional to the distance from the origin in ascending manner. This trend is also true for the planar link 4 in Case 1, but it is different in Case 2, where the overshoot observed 1.9530% for θ_3, which is greater than that of θ_4 (1.6189%). This can be explained as follows. At the beginning of Case 2, the link 4 is in the upright position, Fig. 10. During the motion of Case 2, the link 3 goes beyond the upright position as seen in Fig. 11, and the link 4 moves downward in the gravity direction. In other words, the gravity has a positive effect on the overshot in this case and helps the link 4 to move easily, reducing both the overshoot and settling time. It is clear from Table 1 that the settling time of θ_4 is less than that of θ_3, and rise times of θ_3 and θ_4 are almost identical. Regarding the non-planar link 1, the gravity has no effect, since it is in the same upright position.

As for the control torques, they converge very fast similar to the joint angles. But, as expected and shown in respected figures before, their values much increase from the manual tuning to the DE tuning in order to fast drive these joints from their home positions to final configurations.

4 Feedback Linearization

The basic idea behind the feedback linearization (FBL) is to design a nonlinear controller that linearizes the nonlinear system by an appropriate state space coordinate change. A second-stage control law can then be designed in the current coordinates to fulfill the main requirements of control such as trajectory tracking and disturbance rejection [21]. From the dynamic model given before by Eq. (6), the control law u is written as

$$u = \ddot{\theta} = D(\theta)^{-1}(\tau - C(\theta, \dot{\theta})\dot{\theta} - g(\theta)). \tag{19}$$

After defining the tracking error as $\tilde{\theta} = \theta - \theta_d$, the control law u can be shown to become [21]

$$u = \ddot{\theta}_d - 2\lambda\dot{\tilde{\theta}} - \lambda^2\tilde{\theta} \tag{20}$$

where $\lambda > 0$ leads to an exponentially stable closed-loop system, because the closed-loop error dynamics can be derived from Eqs. (19) and (20) as

$$\ddot{\tilde{\theta}} + 2\lambda\dot{\tilde{\theta}} + \lambda^2\tilde{\theta} = 0. \tag{21}$$

The closed-loop system model in the view of Eqs. (6) and (19) then becomes

$$D(\theta)u + c(\theta, \dot{\theta})\dot{\theta} + g(\theta) = \tau \tag{22}$$

known as the computed torque method in robotics. It is assumed that the above Eq. (22) represents a precise dynamic model. If there are uncertainties in robot dynamical parameters, then the controller performance is affected adversely [21]. Once the dynamic model is completed, the feedback controller can be obtained by substituting Eq. (20) into (22) and hence

$$D(\theta) = \left(\ddot{\theta}_d - 2\lambda\dot{\tilde{\theta}} - \lambda^2\tilde{\theta}\right) + c(\theta, \dot{\theta})\dot{\theta} + g(\theta) = \tau. \tag{23}$$

Both position and trajectory tracking control using the FBL are investigated as given below.

4.1 FBL Position Control

For the position control using the FBL, only results of Case 1 of the previous section (Sect. 3.1.1) are reported, because those of Case 2 are found to be similar. Here the aim is to move all the joints from the rest position of 0 degree to 30 degree configurations. The controller gain λ is taken as 100

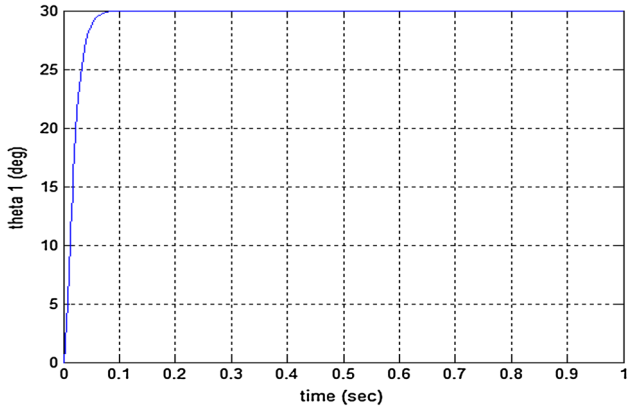

Fig. 12 θ_1 Response for FBL position control, Case 1

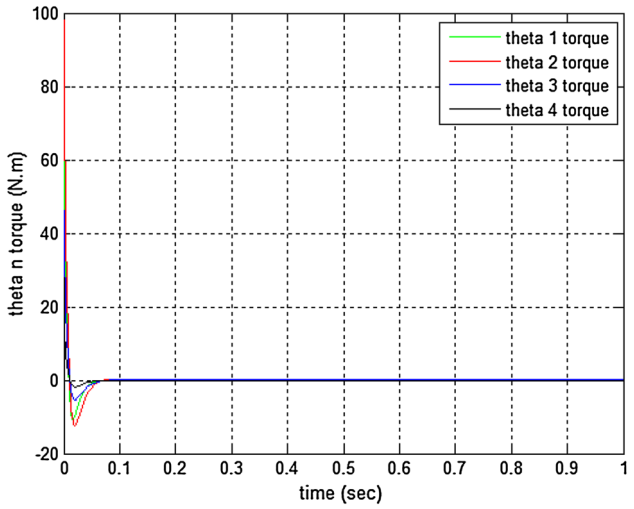

Fig. 13 Control Signals for FBL position control, Case 1

Table 2 Comparison of results

Theta	Controller	Overshoot (%)	Settling time (s)	Rise time (s)
θ_1	PID	2.2940	0.4007	0.0563
	FBL	0.0004	0.0585	0.0336
θ_2	PID	1.5022	0.0976	0.0583
	FBL	0.0004	0.0585	0.0336
θ_3	PID	3.1471	0.7505	0.0638
	FBL	0.0004	0.0585	0.0336
θ_4	PID	3.3943	0.8327	0.0606
	FBL	0.0004	0.0585	0.0336

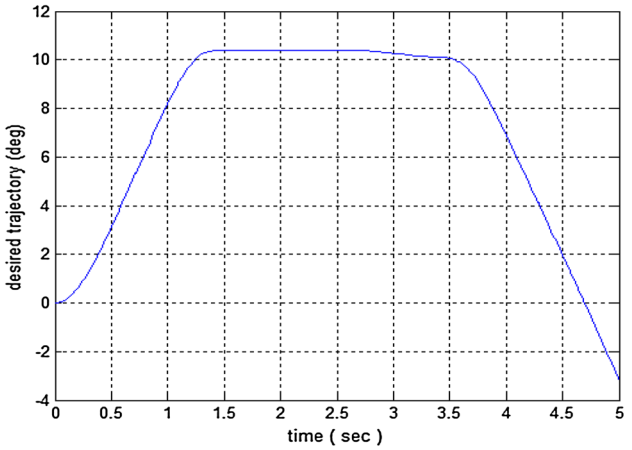

Fig. 14 Desired trajectory of joint angles for FBL trajectory tracking control

in this case. Figure 12 depicts only the first joint angle θ_1 for the position control. Other angles follow almost the same path. It is clear from this figure that the controller is able to accurately place the joints at their desired positions without any steady-state errors. Moreover, the joint torques are depicted in Fig. 13.

Comparing results of the position control using the FBL (Fig. 12) with those for Case 1 of the PID control by the DE algorithm (Fig. 6), as Table 2 presents, it is clear that for all the angles there are no overshoots in the case of the FBL technique and the settling times are much smaller than those of the PID control. It can then be concluded that the FBL position controller performs better than the PID position controller presented before.

4.2 FBL Trajectory Tracking Control

In many cases, joints of the robotic manipulator need to track a time-dependent trajectory in order for the gripper to follow a prescribed path. In such cases, the position control alone is not sufficient and a trajectory control scheme is necessary. Figures 14 and 15 show the desired path for all the joint angles and the tracking error for the joint angle θ_1 using the FBL technique. Only the joint angle θ_1 tracking error is shown, because tracking errors for the other angles are almost identical. There are spikes in Fig. 15 indicating little larger tracking errors around corners in desired path given in Fig. 14, but they can all considered to be negligible. Hence, the FBL control is capable of driving the joint variables into their prescribed path precisely using the same gain of $\lambda = 100$ as before.

5 Uncertainty Analysis

The results so far are obtained based on nominal parameters of the manipulator. It may be advantageous to do an analysis to see the effects of changes in the parameters like masses of the arms in the robot's controlled behavior.

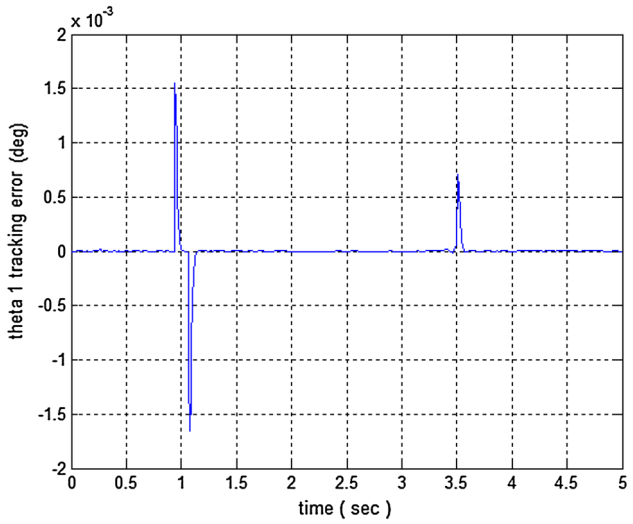

Fig. 15 θ_1 tracking error for FBL trajectory tracking control

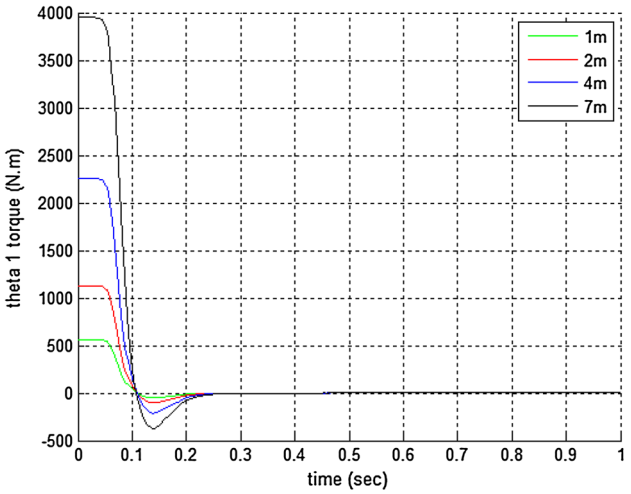

Fig. 17 Joint 1 control signal of PID position control for varying masses, Case 1

5.1 Performance of PID Position Control under Varying Masses

Without loss of generality, masses of the robotic arms are increased up to 7 times of their original values. Due to similarities, only results of Case 1 are presented. It is assumed that the external forces acting on the robot can be added directly as weights acting at the center of gravity of each link to simplify the analysis. Therefore, each time the mass is changed, the corresponding inertia tensor is calculated. Figure 16 shows that the joint angle θ_1 responses of the PID control are almost identical for different masses, indicating that the performance of the PID control is not affected by changing masses of the manipulator links. The other joint angles θ_2, θ_3 and θ_4 also yield very close results. However, heavier links are expected to require higher control torques in order to overcome effects of the gravity and inertia forces, evident in Fig. 17. The problem of high torque requirement can be alleviated by adjusting the PID tuning parameters, but usually at the expense of control performance.

5.2 Performance of FBL Position Control under Varying Masses

Analysis is performed with the same assumptions as the PID control Case 1. Figure 18 for θ_1 indicates that the FBL control leads to closed-loop results that are very close to each other for different masses. Therefore, as above, it can be concluded that variation in masses of robot arms does not greatly affect the closed-loop FBL control behavior. However, Fig. 19 shows that high control torques are needed as the masses increase, but much less than those for the PID control.

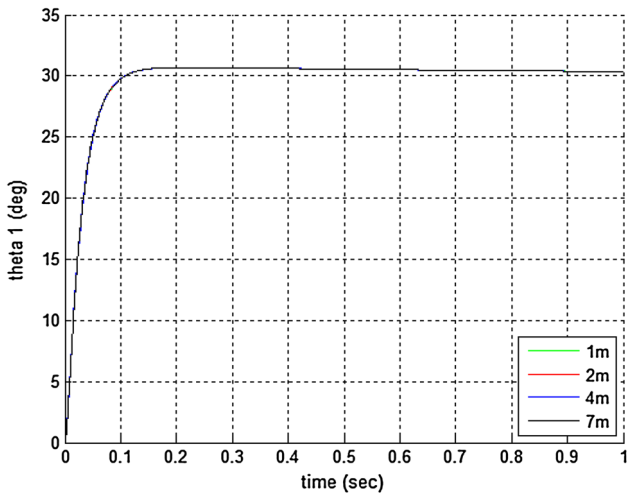

Fig. 16 θ_1 Response of PID position control for varying masses, Case 1

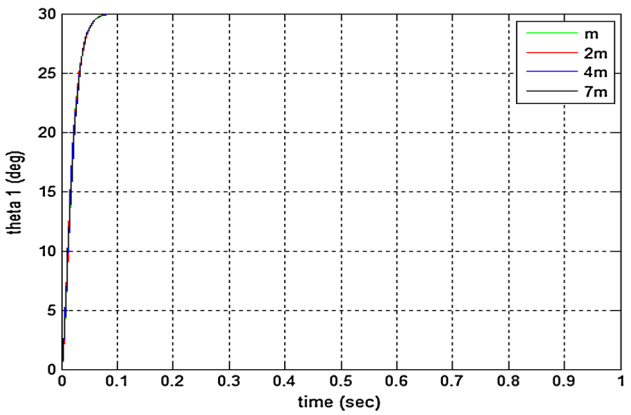

Fig. 18 θ_1 Response of FBL position control for varying masses, Case 1

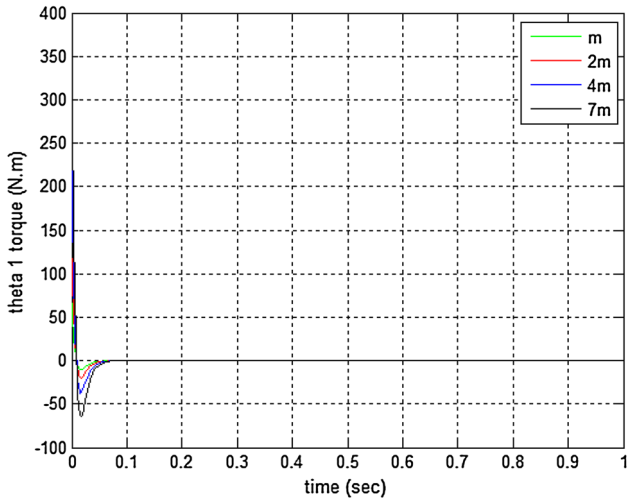

Fig. 19 Joint 1 control signal of FBL position control for varying masses, Case 1

6 Conclusions

The modeling of an existing 4-DOF robotic manipulator is presented using the Euler–Lagrange method. The derived model is then utilized in controlling the manipulator behavior for its position and trajectory by the PID and FBL control schemes. Using the DE algorithm, the PID parameters are tuned to obtain a satisfactory manipulator response in terms of rise and settling times, overshoot and steady-state error for its joint angles. The PID scheme tuned with the DE algorithm leads to good manipulator performance, but the resulting control torques are high. On the other hand, manual tuning of the PID parameters lessens the amount of torque needed, but the control performance is also reduced. Increasing masses of links of the manipulator does not change performance of the PID controller, but amounts of control torques are also increased in proportion to increases in masses. As for the FBL, the position and trajectory tracking controls are implemented and good results are obtained for both. Furthermore, comparison of the results for the PID and FBL position control reveals that both schemes yield satisfactory performances, but results obtained with the FBL approach are superior to those of the PID control.

Acknowledgements Authors gratefully acknowledge the support provided by King Fahd University of Petroleum & Minerals through the NSTIP project 09-ELE786-04.

References

1. Jazar, R.N.: Theory of Applied Robotics. Springer US, Boston (2010). doi:10.1007/978-1-4419-1750-8
2. Kostic, D.; de Jager, B.; Steinbuch, M.; Hensen, R.: Modeling and identification for high performance robot control: an RRR-robotic arm case study. IEEE Trans. Control Syst. Technol. **12**(6), 904–919 (2004)
3. Long, Z.M.; Guo, S.Q.; Chen, G.J.; Yin, B.L.: Modeling and simulation for the articulated robotic arm test system of the combination drive. Appl. Mech. Mater. **151**, 480–483 (2012). doi:10.4028/www.scientific.net/AMM.151.480
4. Wallin, P.J.: Robotics in the food industry: an update. Trends Food Sci. Technol. **8**(6), 193–198 (1997). http://www.sciencedirect.com/science/article/pii/S092422449701042X
5. Meike, D.; Ribickis, L.: Energy effcient use of robotics in the automobile industry. In: 2011 15th International Conference on Advanced Robotics (ICAR), pp. 507–511, Jun (2011)
6. Erzincanli, F.; Sharp, J.M.: A classi_cation system for robotic food handling. Food Control. **8**(4), 191–197 (1997). http://www.sciencedirect.com/science/article/pii/S0956713597000480
7. Ouerfelli, M.; Kumar, V.; Harwin, W.S.: Methods for kinematic modeling of biological and robotic systems. Med. Eng. Phys. vol. **22**(7), 509–520 (2000). http://www.sciencedirect.com/science/article/pii/S1350453300000631
8. Krasilnikyants, E.V.; Varkov, A.A.; Tyutikov, V.V.: Robot manipulator control system. Autom. Remote Control **74**(9), 1589–1598 (2013). doi:10.1134/S0005117913090154
9. Yadav, P.S.; Singh, N.: Robust control of two link rigid manipulator. http://www.ijiee.org/vol5/530-A0016.pdf
10. Xia, Q.X.; Yu, Y.Q.; Liu, Q.B.: Fuzzy control for underactuated manipulator. Appl. Mech. Mater. **397–400**, 1490–1493 (2013). doi:10.4028/www.scientific.net/AMM.397-400.1490
11. Soylemez, M.; Gokasan, M.; Bogosyan, O.: Position control of a single-link robot-arm using a multi-loop PI controller. In: Proceedings of 2003 IEEE Conference on Control Applications, CCA 2003, vol. 2, pp. 1001–1006, Jun (2003)
12. Piltan, F.; Bayat, R.; Aghayari, F.; Boroomand, B.: Design error-based linear model-free evaluation performance computed torque controller. Int. J. Robot. Autom. **3**(3), 151–166 (2012). http://www.researchgate.net/profile/FarzinPiltan3/publication/263969238DesignError-basedLinearModel-freeEvaluationPerformanceComputedTorqueController/links/00b4953c7749e749ec000000.pdf
13. Piltan, F.; Sulaiman, N.; Marhaban, M.H.; Nowzary, A.; Tohidian, M.: Design of FPGA-based sliding mode controller for robot manipulator. Int. J. Robot. Autom. **2**(3), 173–194 (2011). http://www.academia.edu/download/30900697/IJRAV2I3.pdf#page=52
14. Al-Khayyt, S.Z.S.: Tuning PID controller by neural network for robot manipulator trajectory tracking. Al-Khwarizmi Eng. J. **8**(1), 19–28 (2013). http://aliraq.ws/LionImages/News/69953.pdf
15. Receanu, D.: Modeling and simulation of the nonlinear computed torque control in simulink/MATLAB for an industrial robot. SL Struct. Longev. **10**(2), 95–106 (2013). doi:10.3970/sl.2013.010.095.pdf
16. http://www.imagesco.com/kits/robotic-arm.html

17. Arunachalam, V.: Optimization using di_erential evolution (2008). http://ir.lib.uwo.ca/wrrr/22/
18. Taher, S.A.; Afsari, S.A.: Optimal location and sizing of UPQC in distribution networks using differential evolution algorithm. Math. Problems Eng. **2012**, 1–20 (2012). http://www.hindawi.com/journals/mpe/2012/838629/
19. Corke, P.: A robotics toolbox for MATLAB. IEEE Robot. Autom. Mag. **3**(1), 24–32 (1996)
20. Corke, P.I.: A computer tool for simulation and analysis: the robotics toolbox for MATLAB. In: Proceedings of National Conference Australian Robot Association, pp. 319–330 (1995). http://www.pessoal.utfpr.edu.br/winderson/arquivos/ARA95.pdf
21. Fahimi, F.: Autonomous Robots. Springer US, Boston (2009). http://link.springer.com/10.1007/978-0-387-09538-7