University of Thi_Qar Journal for Engineering Sciences ISSN 2075 –9746
http://www.doi.org/10.31663/tqujes.10.1.311(2019)
Vol10 No.1 (May 2019)

Available at http://jeng.utq.edu.iq
engjrnal@jeng.utq.edu.iq

Design and Simulation of Robotic Arm PD Controller Based on PSO

Salah Mahdi Swadi†, Ammar Ibrahim Majeed†, Mushreq Abdulhussain shuriji†, Adnan A. Ugla ‡
Department of Electrical Engineering, Al-Mustansiriyah University, Bagdad-Iraq.
‡ Department of Mechanical Engineering, Thi-Qar University, Al-Nasiriyah-Iraq.

Abstract

This paper deals with design and simulation of Proportional plus Derivative (PD) controller for two links robotic arm. The PD controller was used to compute the required joint's torque for tracking the desired trajectory. The Particle Swarm Optimization (PSO) algorithm has been used in order to tune the gains of the proportional and derivative parts of PD controller. Simulation results show an efficient and robust behavior of the proposed controller. Two cases of simulation tests were carried out. The first was tracking a sine wave trajectory while the second was tracking the same trajectory with the presence of disturbance torque on the dynamic model of the robotic arm. The simulation results were compared with the presence of disturbance torque on the dynamic model of the robotic arm. The simulation results were compared with the results obtained by other researchers, for similar robotic arm properties and same trajectory. The proposed controller offers a simple method for trajectory tracking problem with a satisfactory efficiency. A Mean Square Error (MSE) criterion was used to study the efficiency of the proposed controller in tracking the desired trajectory.

Keywords: *Robotic arm, Trajectory tracking, PD controller, PSO.*

1. Introduction

Robotic arms design was considered as a fresh field of the present technology that leads to a dramatic development in the industrial processes. The use of industrial robotic arms has grown in a fast manner due to the low cost of build, high efficiency, fast, more accuracy and more flexibility [1, 2]. In this paper, a robotic arm which is driven by an optimized PD controller was presented. Robotic arm is a form of kinematic chain consists of several links that are linked by joints. Normally, the joint allow rotation between any two attached links.urrently, industrial robots or manipulators are extensively used in countless application fields. Thus, producing such robots has continuously increased. Control engineering concentrate on design efficient controllers in order to have the manipulators functioned with high accuracy and low amount of errors. In addition, several approaches have been carried out for designing controllers for manipulators [3,4,5,6]. Saad Z. and Saeed suggested a neural solution for tuning the PID controller for Trajectory Tracking [3]. Also, Pooja K. proposed a trajectory control clarification of two link manipulator [4], while H. Delavari and et al anticipated an adaptive fractional of PID controller manipulator [5]. Similarly, Hossein S. and Hassan Z. projected tracking and force control for manipulator [6]. These previous works presented similar robotic arm properties and same trajectory, but with a complicated controller for trajectory tracking problem. Many controlling schemes were proposed in order to control robot arms. Undoubtedly, PID controls are the most widely used control scheme due to its simplicity in structure and robustness of its performance in a massive range of operating conditions. Although, PID control offers the simplest and yet most efficient solution for countless real world controls glitches, optimally tuning gain is pretty challenging [7].

Among several existing tuning techniques, the Ziegler-Nichols formula is the most well-known method. However, tuning is laborious and time consuming, precisely for processes with serious nonlinearities. Therefore, there is a need for retuning before being used to control an industrial process [8].

The evolutionary algorithms such as Bacterial Foraging Optimization (BFO), Particle Swarm Optimization (PSO), Genetic algorithm (GA), and Simulated Annealing (SA) are widespread despite to their abilities to find the global minima in both continuous and non-continuous domains [7].

The PSO algorithm has been proposed by Kennedy and Eberhart [9]. It has verified to be very effective to overcome the complex optimization issues. Commonly, PSO is regarded as a simple concept, computationally efficient, and easy to implement. Unlike the other heuristic techniques, PSO has a well-balanced and flexible mechanism to boost the global and local exploration capabilities [10].

2-Dynamic Model of Robotic Arm

In order to design a controller for the robotic arm, it is essential to have a mathematical model for that arm. A robot

arm can be defined as an open kinematics chain of usually rigid links, which is presented in Figure (1).

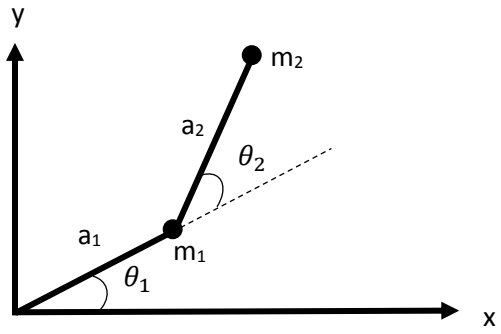

Fig.1 A basic two links robotic arm.

Where m1 and m2 = Mass of first and second links respectively, a1 and a2 = Length of first and second links respectively.

Also, it is used to describe the dynamic parameters and the relationship between displacement, velocity and acceleration to torque/force acting on the joints of the robot arm manipulator [11].

According to the Lagrangian formulation, dynamic of an n joint robot manipulator with revolute joints can be formulated as the follow:

$$\boldsymbol{M(q)\ddot{q} + V(q,\dot{q}) + F(\dot{q}) + G(q) + \tau_d = \tau} \quad \ldots(1)$$

Where, $\boldsymbol{M(q)}$ is a symmetric positive definite inertia matrix, $\boldsymbol{V_m(q,\dot{q})}$ is the centeriptal and coriolis matrix, $\boldsymbol{F(\dot{q})}$ denotes the vector of friction, $\boldsymbol{G(q)}$ is the gravitational vector, $\boldsymbol{\tau_d}$ denotes the bounded unknown disturbances including unstructured and unmodeled dynamics, $\boldsymbol{\tau}$ is the input vector.

Where:

$$\boldsymbol{q} = \begin{bmatrix}\theta_1 \\ \theta_2\end{bmatrix} \quad \ldots (2)$$

$$\boldsymbol{M(q)} = \begin{bmatrix} (m_1+m_2)a_1^2 + m_2a_2^2 + 2m_2a_1a_2cos\theta_2 & m_2a_2^2 + m_2a_1a_2cos\theta_2 \\ m_2a_2^2 + m_2a_1a_2cos\theta_2 & m_2a_2^2 \end{bmatrix} \quad \ldots (3)$$

$$\boldsymbol{V(q,\dot{q})} = \begin{bmatrix} -m_2a_1a_2\left(2\dot{\theta_1}\dot{\theta_2} + \dot{\theta_2^2}\right)sin\dot{\theta_2} \\ m_2a_1a_2\dot{\theta_1^2}\ sin\theta_2 \end{bmatrix} \quad \ldots (4)$$

$$\boldsymbol{G(q)} = \begin{bmatrix} (m_1+m_2)ga_1cos\theta_1 + m_2\ ga_2cos(\theta_1+\theta_2) \\ m_2ga_2\cos(\theta_1+\theta_2) \end{bmatrix} \quad \ldots (5)$$

Generally, the trajectory tracking problem targets at tracking a reference mobile robot with known posture, $q_d = [\theta_{1d}, \theta_{2d}]^T$. Therefore, the errors between the actual and desired posture are express as follows:

$$\mathrm{e}(t) = q_d - q = \begin{bmatrix}\theta_{1d} - \theta_1 \\ \theta_{2d} - \theta_2\end{bmatrix} \quad \ldots (6)$$

The time derivative of Equation 5 gives the dynamics error of posture, as follows:

$$\dot{\mathrm{e}}(t) = \dot{q}_d - \dot{q} \quad \ldots (7)$$

$$\ddot{\mathrm{e}}(t) = \ddot{q}_d - \ddot{q} \quad \ldots (8)$$

Then, the Brunovsky canonical from of the posture error dynamics can be developed in terms of the state e as follows [10]:

$$\frac{d}{dt}\begin{bmatrix}e \\ \dot{e}\end{bmatrix} = \begin{bmatrix}0 & I \\ 0 & 0\end{bmatrix}\begin{bmatrix}e \\ \dot{e}\end{bmatrix} + \begin{bmatrix}0 \\ I\end{bmatrix}\boldsymbol{u} \quad \ldots (9)$$

Where u is the control input.

From equations 6 and 9 a relation between control input u and joints torque τ can be developed. Through this relation, the wheels torque of mobile robot can be controlled as follows:

$$\ddot{\boldsymbol{e}} = (\ddot{\boldsymbol{q}}_{\boldsymbol{d}} - \ddot{\boldsymbol{q}}) = \boldsymbol{u} \quad \ldots (10)$$

$$\boldsymbol{\tau = M(q)(\ddot{q}_d - u) + V(q,\dot{q}) + F(\dot{q}) + G(q) + \tau_d} \quad \ldots (11)$$

In order to compute the control action of the disturbance torque $\boldsymbol{\tau_d}$, the vector of friction $\boldsymbol{F(\dot{q})}$ can be abandonment momentarily. Thus, equation 11 can be written as follows:

$$\boldsymbol{\tau = M(q)(\ddot{q}_d - u) + V(q,\dot{q}) + G(q)} \quad \ldots (12)$$

Equation 12 represents a nonlinear feedback control law, which is guarantee tracking the reference trajectory $q_d(t)$. Moreover, to choose the suitable control signal $\boldsymbol{u(t)}$ that stabilizes the tracking error equation, a PD controller for $\boldsymbol{u(t)}$ with a derivative gain K_d, and proportional gain K_p was implemented to process a controller counting on computation of the motor torque that required to follow the reference trajectory, as shown below:

$$\boldsymbol{u(t) = -K_p e(t) - K_d \dot{e}(t)} \quad (13)$$

Substituting 13 into 12 yields:

$$\boldsymbol{\tau = M(q)(q_d + K_p\ddot{e} + K_d\dot{e}) + V(q,\dot{q}) + G(q)} \quad \dots (14)$$

Furthermore, The PID, PD, and PI controllers are extensively being used in the industries to perform control application. In PD controller case, two parameters K_p *and* K_d should be adjusted to design the controller. In this work the PSO algorithm is used to find the optimal value for these parameters, then it compare with traditional Ziegler Nichols approach.

The complete schematic diagram of the proposed controller is illustrated in the Figure 2.

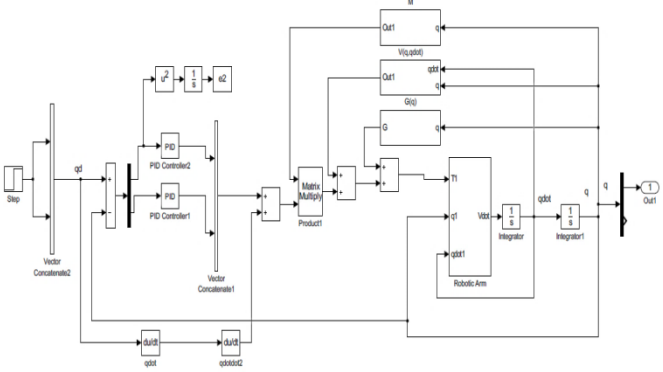

Fig.2 The Simulink model of the proposed controller.

2. Particle Swarm Optimization (PSO)

The particle swarm optimization (PSO) algorithm is a special approach of swarm intelligence based on simplified simulations of animals' social behaviors, such as fish schooling and bird flocking. PSO is a self-adaptive search optimization. The objective of particle swarm optimization is to solve the computationally hard optimization problems. In addition, PSO is a robust optimization technique based on the movement and intelligence of swarms. It's applied effectively to a wide diversity of search and optimization problems. It was inspired from the swarms in nature such as swarms of birds, fish, etc. The PSO developed in 1995 by James Kennedy and Russ Eberhart [9]. The algorithm adopted uses set of particles flying over a search space to locate a global optimum; where a swarm of n particles communicate either directly or indirectly with one another using search directions. Moreover, in PSO iteration, each particle updates its position based on three components. The first is to determine particle velocity using previous velocity. The second is to find the best previous position. And the third is to find the best previous position of its neighborhood. Figure 3 illustrate the flow chart of PSO algorithm. The basic concept of PSO lies in rushing each particle toward the best position found by it so far (pbest) and the global best position (gbest) obtained so far by any particle, with a random weighted acceleration at each time step. It can be executed through the use of equations (15) and (16):

$$v_{t+1} = w * v_t + c_1 * rand(0,1) * (pbest - x_t) + c_2 * rand(0,1) * (gbest - x_t) \quad \dots (15)$$

$$x_{t+1} = x_t + v_{t+1} \quad \dots (16)$$

Where

gbest = Global Best Position.

pbest = Self Best Position.

C1 and C2 = Acceleration Coefficients.

w = Inertial Weight.

Once the particle computes the new xt it then evaluates its new location. If fitness (xt) is better than fitness (pbest) then pbest = xt and fitness (pbest) = fitness (xt), in the end of iteration the fitness (gbest) = the better fitness (pbest) , and gbest = pbest [13].

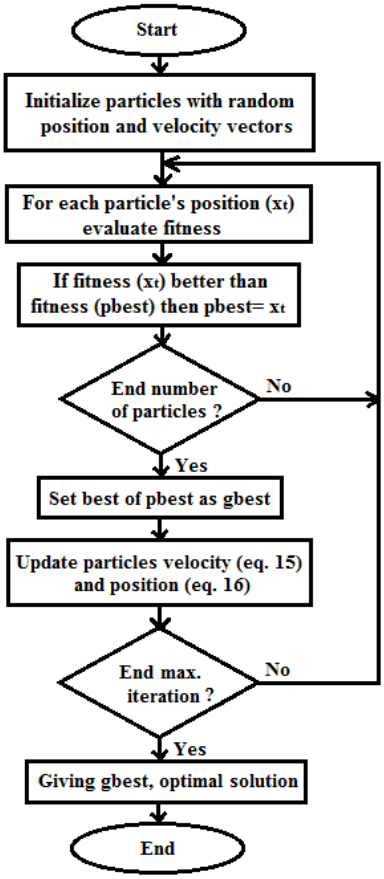

Fig.3 Flow chart of PSO algorithm [11].

3. Simulation and Results

Commonly, the following parameters are used in the simulation of the control laws for the two links robotic arm that shown in Figure 1;

m1= 1 (kg), m2= 1 (kg), a1= 1 (m), and a2=1 (m).

Firstly, the simulation results are obtained for step input in order to evaluate the controller parameter Kp and Kd by means of PSO algorithm. The PSO algorithm has implemented as (m-file) which interconnected to Simulink model shown in Figure 2. The optimization is performed with these initial parameters; number of particles 50, number of dimensions 2, maximum iteration 300, C1=1.5, C2=0.75. The RMS error is selected to be the cost function of the PSO, which are as follows:

$$RMS\theta_1 = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(\theta_{1d}(k) - \theta_1(k))^2} \quad \ldots(17)$$

$$RMS\theta_2 = \sqrt{\frac{1}{n}\sum_{k=1}^{n}(\theta_{2d}(k) - \theta_2(k))^2} \quad \ldots(18)$$

The system response for the step input is shown in Figure 4, while the cost function (RMS) error variation is revealed in Figure 5. Table 1 shows the optimal value of the proposed controller parameter that obtained from optimization program (PSO) with approximate value that obtained from Ziegler-Nichols approach [1].

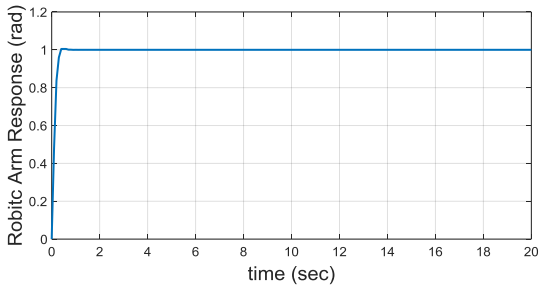

Fig.4 The response of the proposed controller for step input.

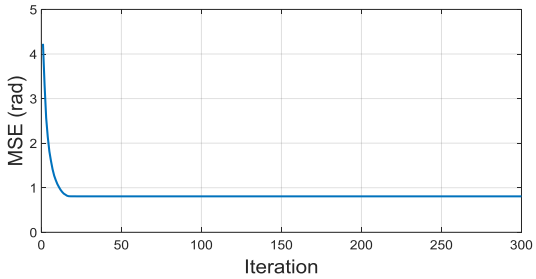

Fig.5 Fitness function in PSO algorithm.

Table 1 The parameters of PD controller obtained by PSO and Ziegler-Nichols approach.

Parameters	K_p	K_d	K_i
PSO algorithm	645.477	43.905	-
Ziegler-Nichols approach [1]	310	60	400

After the completion of the parameters tuning process of PD controller, the next step is to evaluate the proposed controller in tracking a real trajectory. The sinewave trajectory has selected from [3,4,5] where the robotic manipulator joints are driven according to the following desired trajectory:

$$q_{1d} = \sin(t) \quad \ldots(19)$$

$$q_{2d} = \cos(t) \quad \ldots(20)$$

Furthermore, to challenge the proposed controller from tracking the desired trajectory, the initial value of link position and velocity were selected to be as follows:

$$q_{1d(0)} = 0.5\ (rad)\,, \qquad q_{2d(0)} = 0\ (rad),$$
$$\dot{q}_{1d(0)} = 0.1\left(rad/s\right), \qquad \dot{q}_{2d(0)} = 0.1\left(rad/s\right)$$

In this part, the performance of the proposed controller is carried out where Figures (6 and 7) shows the trajectory tracking of the link1 and link2 of the robotic arm.

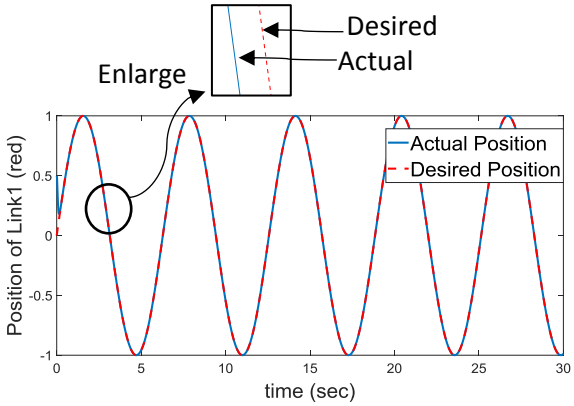

Fig.6 Performance of link1 of the robotic arm.

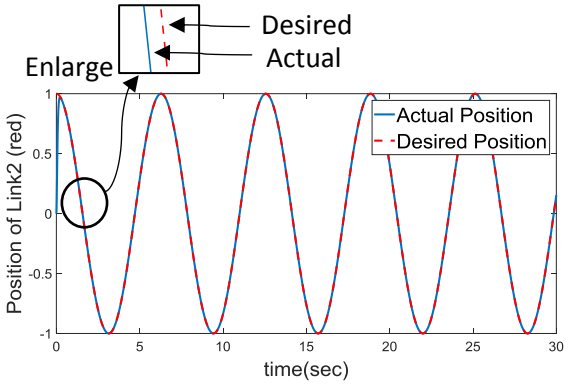

Fig.7 Performance of link2 of the robotic arm.

Figures (8 and 9) shows the efficiency of the proposed controller from tracking the desired joints velocity (rad/s) of link1 and link2.

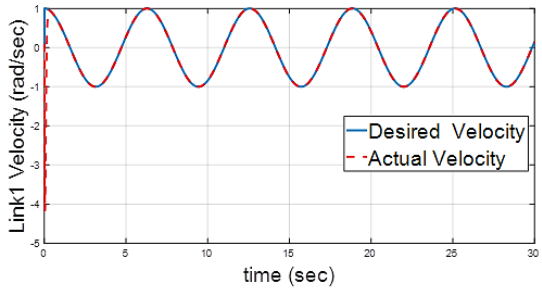

Fig.8 Velocity tracking of link1.

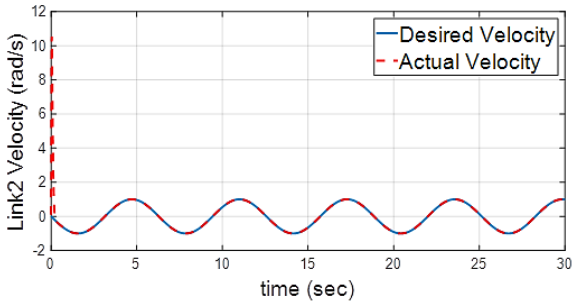

Fig.9 Velocity tracking of link2.

Figures (10 and 11) demonstrate the power of the proposed controller from tracking the desired trajectory with minimum error tracking. It was found that the value of the RMS error of joints position was (0.0016 and 4.2775e-4 (rad)) for link1 and link2 respectively.

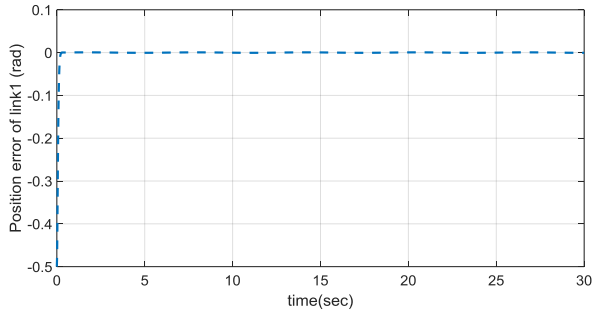

Fig.10 Position tracking error of link1

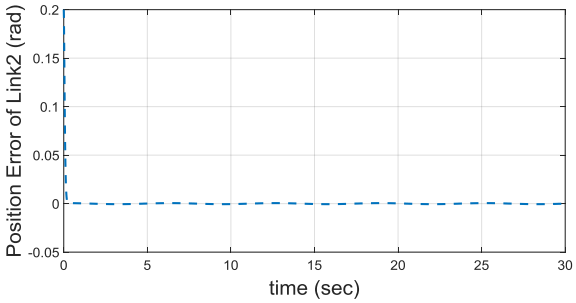

Fig.11 Position tracking error of link2.

In order to validate the proposed controller in tracking the desired trajectory a disturbance torque has added to the system where this disturbance torque has the following expression [1]:

$$\boldsymbol{\tau_d} = \begin{bmatrix} 0.4\sin(5\pi) \\ 0.6\sin(5\pi) \end{bmatrix} \quad \ldots (21)$$

The robotic manipulator joints are driven according to the following desired trajectory:

$$q_{1d} = \sin(t) \quad \ldots (22)$$

$$q_{2d} = \sin(t) \quad \ldots (23)$$

While the initial conditions were selected to be as follows:

$$q_{1d(0)} = 0.5\ (rad)\ , \quad q_{2d(0)} = 0\ (rad),$$

$$\dot{q}_{1d(0)} = 0\ \left(rad/s\right), \quad \dot{q}_{2d(0)} = 0\ \left(rad/s\right)$$

The performance of the proposed controller is shown in Figures (12-17). Figures (12 and 13) show the power disturbance elimination in the proposed controller where the added disturbance torque has no effects on trajectory tracking performance of the robotic arm.

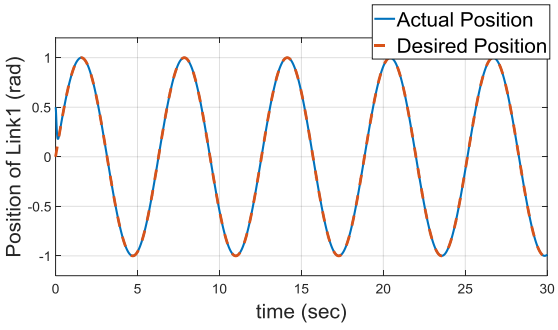

Fig.12 Performance of link1 of the robotic arm.

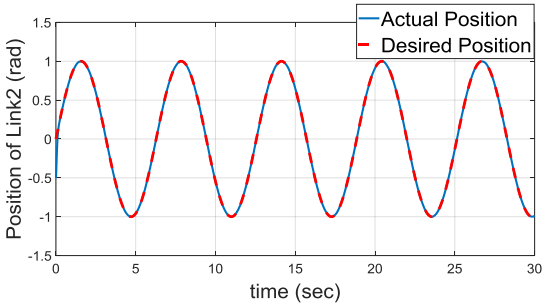

Fig.13 Performance of link2 of the robotic arm.

As in the trajectory tracking results, the velocity tracking results shows the same performance as shown in Figures (14 and 15);

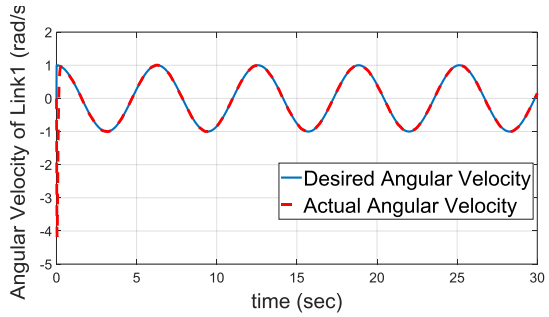

Fig.14 Velocity tracking of link1.

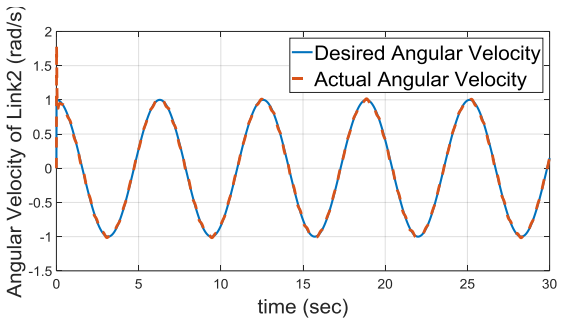

Fig.15 Velocity tracking of link2.

Figures (16 and 17) demonstrate the power of the proposed controller from tracking the desired trajectory with minimum error tracking and it was noticed very small waves in the graph due to the presence of disturbance torque which can be neglected consequently. The value of the RMS error of joints position was (0.0016 and 4.2775e-4 (rad)) for link1 and link2 respectively.

Comparing the above results with the results introduced in [1], which are (0.02445 and 0.0233 (rad)) for link1 and link2 respectively, it indicate that the proposed controller has lower MSE.

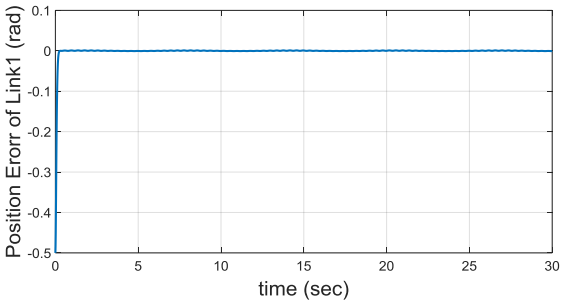

Fig.16 Position tracking error of link1.

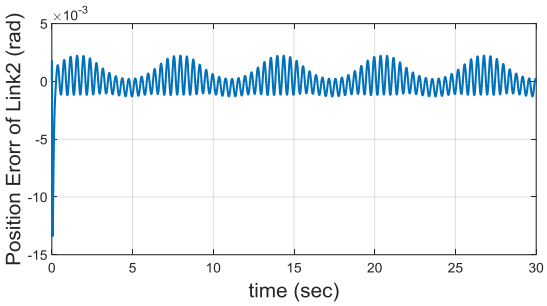

Fig.17 Position tracking error of link2.

Figure (18) shows the signal of the disturbance torque that added to the dynamic system of robotic arm to show the robustness of the proposed controller form reject any uncertainty in dynamic model of robotic arm.

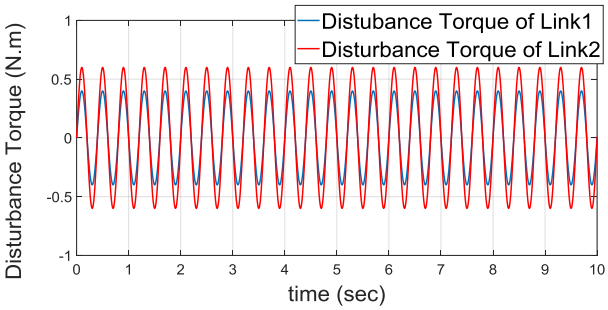

Fig.18 Disturbance torque that add on system.

Finally, Figures (19 and 20) illustrate the control input torque that produce from the proposed controller for jont-1 and joint -2 respectively. It's clear to say that, there are some waves in both graphs, and it is necessary to eliminate the disturbance torque.

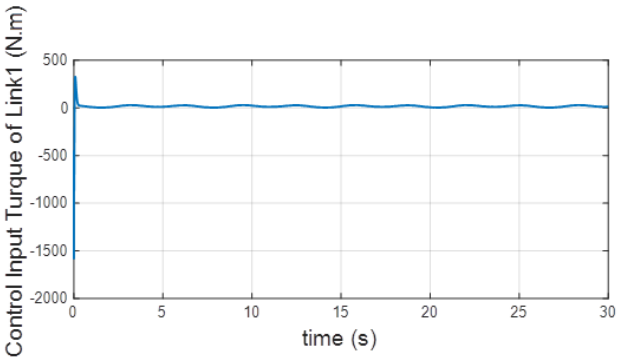

Fig.19 Control signal of the proposed controller of link1.

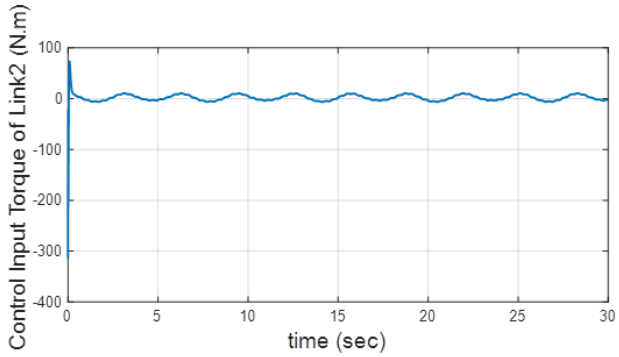

Fig.20 Control signal of the proposed controller of link2.

Conclusions

The main contribution of this work is using simple controller (PD) with the nonlinear dynamic system (robotic arm) to compensate any uncertainty in the system model. PSO algorithm has been used as a powerful tool to find optimal value of the PD parameters with minimum cost function. In conclusion, simulation results show very satisfactory performance for the proposed controller in tracking difficult trajectory with varying initial conditions. Although a disturbance torques have added to the dynamic system of robotic arm, the proposed controller shows a very decent efficiency in tracking the desired trajectory with minimum RMS.

References

[1] John J. Craig, "Introduction to Robotics Mechanics and Control", Pearson Education International, ISBN 0-13-123629-6, 2005.

[2] Mark W. Spong, Seth Hutchinson, And M. Vidyasagar," Robot Modeling And Control", John Wiley & Sons, Inc., ISBN: 978-0-471-64990-8, 2006.

[3] Saad Zaghlul Saeed Al-Khayyt, "Tuning PID Controller by Neural for Robot Manipulator Trajectory Tracking", Al-Kwarizmi Engineering Journal, Vol. 8, No. 1, p.p 19-18, 2013.

[4] Pooja Kharti and et al., "Trajectory Control of Two Link Robotic Manipulator Using PID", Golden Research Thoughts, ISSN 223-5063, Vol. 3, Issue 5, Nov., 2013.

[5] H. Delavari, R. Ghaderi N., S. H. HosseiNia, and S. Momani, "Adaptive Fractional PID Controller for Robot Mainpulator", Proceeding of FDA 10. The 4th IFAC Workshop Fractional Differentiation and its Applications. Badajoz, Spain, Octorber 18-20, 2010.

[6] Hossein Sadegh Lafmejani and Hassan Zarabadipour, " Modeling, Simulation and Position Control of 3DOF Articulated Manipulator", Inonesian Journal of Electrical Engineering and Informatics (IJEEI), Vol. 2, No. 3, pp. 132-140, September, 2014.

[7] Mickael Aghajarian, Kourosh Kiani, and Mohammad Mehdi Fateh, "Design of Fuzzy Controller for Robot Manipulators Using Bacterial Foraging Optimization Algorithm ", Journal of Intelligent Learning systems and Applications, Vol. 4, pp. 53-58, 2012.

[8] Gawthrop, P. J., Nomikos, and P.E., "Automatic Tuning of Commercial PID Controller for single-loop and Multiloop Application", IEEE Control Systems Magazine, Vol. 10, pp. 34-42, 1990.

[9] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," IEEE International Conference on Neural Networks, Vol. 4, Perth, pp. 1942-1948, 27 November-1 December 1995.

[10] M. Clerc and J. Kennedy, "The Particle Swarm-Explosion, Stability and Convergence in A Multidimensional Complex Space," IEEE Transection on Evolutionary Computation, Vol. 6, No. 1, pp. 58-73, 2002.

[11] Reham H. Mohammed, Fahmy Bendary, and Kamel Elserafi, "Trajectory Tracking Control for Robot Manipulator using Fractional Order-Fuzzy-PID Controller," Inernational Journal of Computer Application, Vol. 134, No. 15, January 2016.

[12] Salah Mahdi Swadi, "Trajectory Tracking Control of Autonomous Chaotic Wheeled Mobile Robot," Ph.D. Dissertation, University of Technology, Iraq, Baghdad, p. 42, April, 2016.

[13] V. Gazi, and K. M. Passino, "Swarm Stability and Optimization ", German; Springer Science Business Media, 2011.