INTERNATIONAL JOURNAL OF ROBOTICS
(THEORY AND APPLICATIONS)
E-mail: IJR@kntu.ac.ir

PSO Based Solution for 6-DOF Serial Manipulator Inverse Kinematics Problem

K. Almaghout[a] and A. Rezaee[b, *]

[a] Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran
[b] Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran

ARTICLE INFO

Article history:
Submit: 2018-07-27
Revise: 2023-09-11
Accept: 2023-09-12

Keywords:
Particle Swarm Optimization
Inverse Kinematics
Manipulator
Orientation
Positioning

ABSTRACT

This paper introduces an optimization technique based on the particle swarm optimization algorithm (PSO) for solving the inverse kinematics problem for an n-DOF manipulator. The proposed algorithm trying iteratively to find the best set of angles that locus the manipulator at the desired position and orientation. Each iteration, a set of angles are assigned to the joints and derived to calculate the position and orientation of the end-effector using Denavit-Hartenberg (DH) method and Euler angles equations. Then obtaining the error between the current position/orientation $(\boldsymbol{P_c}/\boldsymbol{O_c})$ of the end-effector with the desired position/orientation $(\boldsymbol{P_d}/\boldsymbol{O_d})$. A 6-DOF manipulator has been used as an example in our simulation. Obtained results show that PSO can be efficiently used for inverse kinematics solution.

* Corresponding address: Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran
P.O. Box, 14399-57131,
E-mail address: arrezaee@ut.ac.ir

1. Introduction

The inverse problem of finding the joint variables in terms of the end-effector's position and orientation is the problem of inverse kinematics.

Inverse kinematics plays key role in several robot's control systems, such as off-line path planning control motion control

The exact solution of inverse kinematics is very important to control the robot. And it is, in general, more difficult than the forward kinematics problem. And for some structure of the machine arm is concerned, it doesn't even exist equipped with inverse solution.

There are traditional methods such as algebraic solutions, geometric solutions and iterative solutions in order to solve the inverse kinematics problem. However, these methods are time-consuming and suffer from numerical problems [1]. Furthermore, as joint structure of manipulator is more complex, inverse kinematics solution also is more difficult.

Over the past decade, many methods have been proposed to solve the inverse kinematics instead of traditional methods [2-3]. Tejomurtula and Kak proposed the NN using error-back propagation (BP) to solve inverse kinematics [2]. Köker et al. designed a multi-layer feed forward neural network (NN) for three-joint robot [4]. NN was trained until acceptable error. Other investigators discussed the applications of back propagation (BP) neural network and radial basis function (RBF) neural network to the kinematics problem of parallel manipulators [5,6]. The connecting weights of neural networks can be determined by training a large number of samples provided by a series of results of inverse kinematics.

The greatest disadvantage of NN's is that it must be trained for a long period.

The evolutionary methods such as genetic algorithm (GA) have been used to inverse kinematics problem. Zheng and Jiao transformed the forward kinematics problem into an optimization problem and then a genetic algorithm was used to minimize the difference between the computed and the given link length [7]. The computed length of each link can be obtained by solving the inverse kinematics. It should be noted that the results of the position and pose derived from a genetic algorithm are not always optimal solutions, given that a genetic algorithm can easily fall into a local minimum [8, 9]. Geem et al. have proposed a harmony search algorithm (HSA) based on evolution [10]. HSA have been used to various numerical problems. However, it was noted that, HSO gets into trouble in local search [11].

Recently, the particle swarm optimization (PSO) has been successfully applied to various optimization problems [12-13]. In this paper, the inverse kinematics problem of a 6-DOF robot manipulator has transformed into an optimization problem and then used the PSO algorithm to obtain an optimal inverse kinematics solution by taking advantage of the global optimization property of this algorithm.

2. FORWARD KINEMATICS OF ABB IRB 1200

As illustrated in this document, the numbering for sections upper case Arabic numerals, and for the sub-sections, the upper case Arabic numerals, separated by periods. Initial paragraphs after the section title are not indented. Only the initial, introductory paragraph has a drop cap.

ABB IRB 1200 robot, Figure 1, is a robot with 6 degrees of freedom, all joints of it are rotational joints. The first three joint main influence at the end of the implementation of the position, after three joint determines the end actuators attitude, after a three joint axis to a little. You can see more detailed description in [14]. Table 1. Shows Denavit-Hartenberg (DH) parameters.

Based on DH Table, the homogenous transformation which describes the position and orientation of the end-effector, represented by the coordinate systems 6 $(x_6.y_6.z_6)$, respect to the reference coordinate systems $(x_0.y_0.z_0)$, shown in figure 2, can be obtained as below [15]:

$$ {}^{0}_{6}T = {}^{0}_{1}T\,{}^{1}_{2}T\,{}^{2}_{3}T\,{}^{3}_{4}T\,{}^{4}_{5}T\,{}^{5}_{6}T $$

where:

$$ {}^{i-1}_{i}T = \begin{bmatrix} C\theta_i & -S\theta_i & 0 & a_{i-1} \\ S\theta_i C\alpha_{i-1} & C\theta_i C\alpha_{i-1} & -S\alpha_{i-1} & -S\alpha_{i-1}d_i \\ S\theta_i S\alpha_{i-1} & C\theta_i S\alpha_{i-1} & C\alpha_{i-1} & C\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

Figure 1. ABB IRB 1200

TABLE 1
DH PARAMETER

Link i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	399	θ_1
2	0	$-\frac{\pi}{2}$	0	θ_2
3	448	0	0	θ_3
4	42	$-\frac{\pi}{2}$	451	θ_4
5	0	$\frac{\pi}{2}$	0	θ_5
6	0	$-\frac{\pi}{2}$	0	θ_6

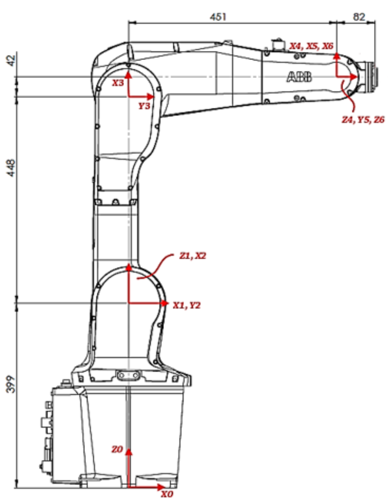

Figure 2. joints coordinate systems

$$ {}^0_1T = \begin{bmatrix} c1 & -s1 & 0 & 0 \\ s1 & c1 & 0 & 0 \\ 0 & 0 & 1 & 399 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^3_4T = \begin{bmatrix} c4 & -s4 & 0 & 42 \\ 0 & 0 & 1 & 451 \\ -s4 & -c4 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^1_2T = \begin{bmatrix} c2 & -s2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s2 & -c2 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^4_5T = \begin{bmatrix} c5 & -s5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ s5 & c5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^2_3T = \begin{bmatrix} c3 & -s3 & 0 & 448 \\ s3 & c3 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad {}^5_6T = \begin{bmatrix} c6 & -s6 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -s6 & -c6 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

$$ {}^0_6T = \begin{bmatrix} r_{11} & r_{12} & r_{13} & P_x \\ r_{21} & r_{22} & r_{23} & P_y \\ r_{31} & r_{32} & r_{33} & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} R(3,3) & \vdots & P(3,1) \\ \cdots & \cdots & \cdots \\ 0 & \vdots & 1 \end{bmatrix} $$

where:

$ci: cos(\theta_i),\ \ si: sin(\theta_i)$

$r_{11} = s6(c4s1 - c23c1s4) + c6(c5(s1s4 + c23c1c4) - s23c1s5)$

$r_{12} = c6(c4s1 - c23c1s4) - s6(c5(s1s4 + c23c1c4) - s23c1s5)$

$r_{13} = -s5(s1s4 + c23c1c4) - s23c1c5$

$r_{21} = -s6(c1c4 + c23s1s4) - c6(c5(c1s4 - c23c4s1) + s23s1s5)$

$r_{22} = s6(c5(c1s4 - c23c4s1) + s23s1s5) - c6(c1c4 + c23s1s4)$

$r_{23} = s5(c1s4 - c23c4s1) - s23c5s1$

$r_{31} = s23s4s6 - c6(c23s5 + s23c4c5)$

$r_{32} = s6(c23s5 + s23c4c5) + s23c6c4$

$r_{33} = s23c4s5 - c23c5$

$P_x = c1(\ 42c23 - 451s23 + 448c2)$

$P_y = s1(42c23 - 451s23 + 448c2)$

$P_z = 399 - 42s23 - 448s2 - 451c23$

$P_x . P_y . P_z$ represents the position of the end-effector as a function of to the angles of joints.

In order to obtain the orientation of the end-effector, we calculate the **Roll, Pitch, Yaw** angles ($\psi . \theta . \phi$) of the end-effector about the axes $x_0 . y_0$ *and* z_0 respectively, figure 3, by extracting the rotational matrix R(3,3) using the following equations [15]:

$$ \theta = Atan2\left(-r_{31} . \sqrt{r_{11}^2 + r_{21}^2}\right) $$

$$ \phi = Atan2(r_{21}/c\theta \,. r_{11}/c\theta) $$

$$ \psi = Atan2(r_{32}/c\theta \,. r_{33}/c\theta) $$

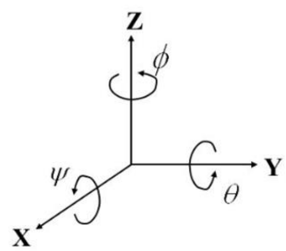

Figure 3. Roll ψ, Pitch θ, Yaw ϕ angles

3. PARTICLE SWARM OPTIMIZATION (PSO)

The PSO is an optimization algorithm based on swarm behaviors. It simulates social behavior of organisms such as fish schooling and bird flocking [16]. PSO arts generating initial population. Each individual in the initial population are randomly selected from search space. The performance of each individual is measured according to a pre-defined fitness function, which is related to the problem to be solved. A best solution is evolved through the generations. PSO optimization is obtained by individual's movement in the search space. The position and velocity of individuals of population are updated by applying an operator so that individuals can be expected to move towards the better solution. Updating procedure is defined by following equations.

$$V_i^d(t+1) = w(t).V_i^d(t) + c_1.r_1.\left(p_i^d(t) - x_i^d(t)\right) \ \ldots$$
$$+c_2.r_2.\left(p_g^d(t) - x_i^d(t)\right)$$

$$x_i^d(t+1) = x_i^d(t)V_i^d(t+1)$$

where $V_i^d(t)$ is the velocity of i_{th} agent at time t, $x_i^d(t)$ is the position of i_{th} agent at time t, $p_i^d(t)$ is the best previous position of i_{th} agent, $p_g^d(t)$ is the best previous position of the population, c_1 and c_2 are positive constants, r_1 and r_2 are random constants changing randomly each iteration in the range [0,1], d is dimension, and $w(t)$ is inertia weight.

4. SIMULATION RESULTS

In order to clarify the effectiveness of the PSO in solving the inverse kinematics, a simulation study was achieved using MATLAB, joints constraints, shown in table 2, were taken into consideration in designing PSO algorithm.

Table 3 shows that for $P_x = 766.87. P_y = 135.2199. P_z = -37.9406. \psi = 16.13°. \theta = 2.217°. \phi = 55.43°$ given as the desired position/orientation, the algorithm has been applied 5 times on this data and for it obtained an acceptable results for the angles of joints that locates the end-effector in the desired position/orientation with the negligible error. Figure 4 and 5 show the errors decreasing each iteration.

25 samples that chosen randomly, related position P_1 and Euler angles E_1 have been calculated, then these results taken as input for the proposed algorithm, the resulted sets of angles from the algorithm used again to calculate position P_2 and Euler angles E_2. Figure 6 and 7 show the errors $e_E = E_1 - E_2$, and $e_P = P_1 - P_2$ respectively.

TABLE 2
JOINTS WORKING RANGE

Joint i	*Working Range (degree)*
θ_1	*+170 to -170*
θ_2	*+130 to -100*
θ_3	*+70 to -200*
θ_4	*+270 to -270*
θ_5	*+130 to -130*
θ_6	*+400 to -400*

TABLE 3
SIMULATION EXAMPLE

	P_x mm	P_y mm	P_z mm	ψ degree	θ degree	ϕ degree
Desired	766.870	135.2199	-37.9406	116.1293	12.217	55.4387
PSO Result 1	766.8862	135.2363	-37.9636	116.1293	12.217	55.4387
PSO Result 2	766.8883	135.3116	-37.9483	116.1293	12.2170	55.4387
PSO Result 3	766.8989	135.1804	-37.9108	116.1293	12.2170	55.4387
PSO Result 4	766.8517	135.2156	-37.9245	116.1293	12.2170	55.4387
PSO Result 5	766.8685	135.2237	-37.9360	116.1293	12.2170	55.4387

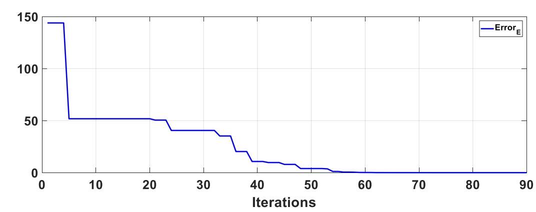

Figure 4. Error between desired Euler Angles (ψ, θ, ϕ) and the calculated ones for each iteration

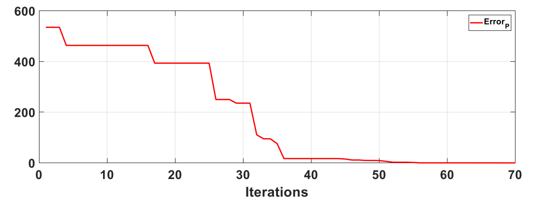

Figure 5. Error between desired position (x, y, z) and the calculated one for each iteration

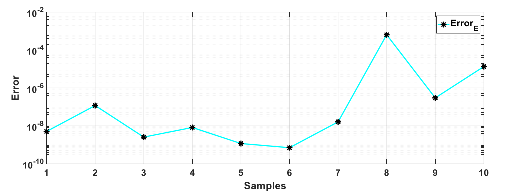

Figure 6. Error between desired Euler Angles (ψ, θ, ϕ) and the calculated ones for each iteration

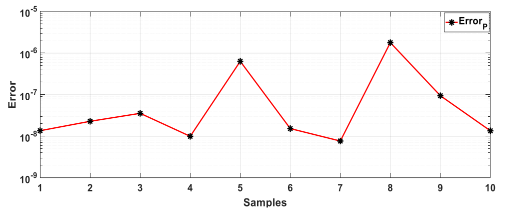

Figure 7. Error between desired position (x, y, z) and the calculated one for each iteration

5. CONCLUSIONS

In this paper, inverse kinematics problem has been transformed into an optimization problem. PSO algorithm was applied. The average elapsed time is 0.030 second for obtaining

the angles that put the end-effector in the desired position (x, y, z) and it is 0.250 second for obtaining the angles those put the end-effector at the desired orientation. The simulation was performed using MATLAB 2017b and medium-specifications laptop (the elapsed time depends on the used computer specifications mainly). The simulation results showed the effectiveness of the proposed algorithm.

REFERENCES

[1] Kucuck, S., and Zafer Bingul. "The inverse kinematics solutions of fundamental robot manipulators with offset wrist." Mechatronics, 2005. ICM'05. IEEE International Conference on. IEEE, 2005.

[2] Tejomurtula, Sreenivas, and Subhash Kak. "Inverse kinematics in robotics using neural networks." Information sciences 116.2-4 (1999): 147-164.

[3] Kalra, Parveen, P. B. Mahapatra, and D. K. Aggarwal. "On the solution of multimodal robot inverse kinematic functions using real-coded genetic algorithms." Systems, Man and Cybernetics, 2003. IEEE International Conference on. Vol. 2. IEEE, 2003.

[4] Köker, Raşit, et al. "A study of neural network based inverse kinematics solution for a three-joint robot." Robotics and autonomous systems 49.3-4 (2004): 227-234.

[5] Parikh, Pratik J., and Sarah SY Lam. "A hybrid strategy to solve the forward kinematics problem in parallel manipulators." IEEE Transactions on Robotics 21.1 (2005): 18-25.

[6] Song, Wei-gang, and Guo-wei Zhang. "Direct Kinematic Problem Based on RBFNN of Parallel Manipulator." JOURNAL-NORTHEASTERN UNIVERSITY NATURAL SCIENCE 25 (2004): 386-389.

[7] Zheng, C. H., and L. C. Jiao. "Forward kinematics of a general Stewart parallel manipulator using the genetic algorithm." Journal of Xidian University 30.2 (2003): 165-173.

[8] Hwang, Shun-Fa, and Rong-Song He. "Improving real-parameter genetic algorithm with simulated annealing for engineering problems." Advances in Engineering Software 37.6 (2006): 406-418.

[9] Ghazanfari, Mehdi, et al. "Comparing simulated annealing and genetic algorithm in learning FCM." Applied Mathematics and Computation 192.1 (2007): 56-68.

[10] Geem, Zong Woo, Joong Hoon Kim, and Gobichettipalayam Vasudevan Loganathan. "A new heuristic optimization algorithm: harmony search." simulation 76.2 (2001): 60-68.

[11] Lee, Kang Seok, and Zong Woo Geem. "A new structural optimization method based on the harmony search algorithm." Computers & structures 82.9-10 (2004): 781-798.

[12] Kennedy, James. "Particle swarm optimization." Encyclopedia of machine learning. Springer US, 2011. 760-766.

[13] Zhao, Liang, et al. "Automatically extracting T–S fuzzy models using cooperative random learning particle swarm optimization." Applied soft computing 10.3 (2010): 938-944.

[14] https://library.e.abb.com/public/a72558f2c3d142cdaf469 12468404883/IRB1200-ROB0275EN-Rev.F.pdf

[15] Craig, John J. Introduction to robotics: mechanics and control. Vol. 3. Upper Saddle River, NJ, USA: Pearson/Prentice Hall, 2005.
[16] J. Kennedy and R. C. Eberhart, “Particle swarm optimization”, in Proceedings of 1995 IEEE International Conference on Neural Networks, vol. 4, Perth, WA, Australia, pp. 1942-1948.

Biography

Karam Almaghout is currently a Master of mechatronics engineering student at the faculty of New Sciences and Technologies at University of Tehran. He received his BSc., degrees in mechatronics engineering from Albaath University, Homs, Syria in 2014. His research interests include control systems and robotics.

Alireza Rezaee received his BS degree in Control Engineering from Sharif University of Technology, Iran (2002) and the MS degree and PhD in Electrical Engineering from Amirkabir University of Technology (2005 and 2011). He is currently an Associate professor of mechatronic Engineering, Faculty of New sciences & technologies, University of Tehran, Tehran, IRAN. His field of research is machine learning, Bayesian networks and robotics.