Research Article

Simulation based calculation of the inverse kinematics solution of 7-DOF robot manipulator using artificial bee colony algorithm

Serkan Dereli[1] · Raşit Köker[2]

Received: 21 July 2019 / Accepted: 26 November 2019 / Published online: 5 December 2019

Abstract
This paper presents an artificial bee colony algorithm for solving the inverse kinematics of 7-degree-of-freedom robotic arm which has been newly designed and not used in the literature. The kinematics analysis of this manipulator which has an excessive number of joints, is quite complex. In this study, artificial bee colony, which is one of the swarm-based heuristic algorithms, has been used for inverse kinematics solution and its results have been analyzed in terms of position error and calculation time. In order to ensure the accuracy of the algorithm, calculations have been also carried out in 100 different points selected from the workspace of the robot manipulator. The results have been compared with particle swarm optimization, which is another swarm algorithm in terms of position error and computation time. The results obtained by computer simulation clearly show that the artificial bee colony algorithm produces effective results compared with the literature.

Keywords Inverse kinematics · Robotic · Artificial bee colony · Redundant manipulator · Heuristic

1 Introduction

Although robot technologies that greatly simplify our lives is a very important technology, it is known inverse kinematics solution of robots is one of the difficult and very time-consuming problems and has a non-linear characteristic [1–3]. Therefore, it is of great importance to overcome these problems which have a very widespread research, such as reducing the computation time, getting the right solution, to achieve the shortest path planning [4]. Recently, 7-DOF robot manipulators, which have come to the forefront with their skill in avoiding obstacles, have become the focus of the research world. However, control of this robot so difficult and complicated owing to the large number of joint [5]. The basis of robot control is the kinematic calculations which are divided into two as forward and inverse kinematics. These calculations, which express robot movements, are a very common field of study. Forward kinematics that is used to obtain the position of the end effector in cartesian space from the joint angles is relatively easy [6]. On the other hand, inverse kinematics that is conversion of the position and orientation of the robot manipulator end-effector from Cartesian space to joint space is a more complex, non-linear equations and impossible to solution by conventional methods such as geometric, iterative and algebraic [7, 8]. Moreover, it has a large number of solutions. Hence, inverse kinematics problem is suitable for the heuristic methods such as particle swarm optimization, artificial bee colony, firefly algorithm and artificial neural network [9–11]. Because artificial intelligence techniques are indispensable methods for solving difficult, complex and long-time problems, it is actively used at every stage of robot technology [12, 13].

Momani et al. have implemented an inverse kinematics solution of an articulated robot manipulator using traditional and improved genetic algorithm methods [14]. Tejomurtula and Kak perform the inverse kinematic solution of 3-jointed robot arm using artificial neural networks that eliminates some of the disadvantages of this method BP algorithm,

✉ Serkan Dereli, dereli@subu.edu.tr | [1]Computer Technology Department, Adapazari Vocational High School, Sakarya University of Applied Sciences, 54290 Adapazarı, Sakarya, Turkey. [2]Department of Electrical and Electronics Engineering, Technology Faculty, Sakarya University of Applied Sciences, 54540 Serdivan, Sakarya, Turkey.

such as training time and accuracy [15]. Dash et al. [16] perform solution a new method based on artificial neural network and simulated this solution via six-jointed robot arm. Huang et al. [17] carry out inverse kinematics solution of seven-joined robot manipulator quickly and accurately using particle swarm optimization. Ayyıldız and Çetinkaya [18] have designed a 4-DOF robot manipulator and inverse kinematics solutions of its have achieved the four different (PSO, QPSO, GSA and GA) optimization algorithm and they demonstrated comparatively results obtained. Rokbani and Alimi studied the contribution to the solution of inverse kinematics using variants of PSO, such as inertia weight, constriction factor and linear decreasing weight. In addition, they have used a two-jointed robotic arm for simulation test [19]. Rokbani et al. prefer to use the firefly intelligent method which is a new heuristic algorithm based on swarm, in their work and tested the proposed method in a three-jointed robot arm [20]. Koker has proposed a hybrid method which was used with the neural network and genetic algorithm to solve the inverse kinematics solution of a six-joint robotic manipulator to minimize the error of the end effector [21]. Similarly, Pam et al. use together bee algorithm and artificial neural network of the inverse kinematics of an articulated robotic manipulator which has a three-joint. The bee algorithm was used to train the neural network which has a multilayer perceptron structure [22].

In general, the calculation steps used for heuristic methods have been also preferred in this study. The robot manipulator is manually directed to a predetermined position. The artificial bee colony algorithm and particle swarm optimization are used to obtain the optimal joint angles that reach the nearest point to this position of the end effector. The main focus of the study is to perform the inverse kinematics calculation with ABC technique and to compare the results with PSO which is another widely used technique in the literature. Therefore, the following parts of the study are organized in this framework. In Chapter 2, the newly designed robot manipulator used in this study is introduced and kinematics equations of this robotic arm are created. In the following, the artificial bee colony algorithm and particle swarm optimization is briefly summarized and the fitness function that finds the distance between the desired position and the actual position, to be used in this algorithm is introduced. In Chapter 3, simulation results are obtained and presented. In the last section, the results have been analyzed and compared with previous studies.

2 Materials and methods

2.1 Kinematics analysis of a 7-DOF redundant robot manipulator

Robotic manipulators which are available in two forms: prismatic and rotational, consist of links sequentially connected to each other with joints which perform a movement mechanism taking certain angles or by a certain percentage of elongation and shortening by actuators [23]. Designed robot manipulator for this study has seven rotational joints and is shown in Fig. 1. A 7-DOF robotic manipulator not only performs the movement from one position to another position in a comfortably but also has infinite inverse kinematics solutions. Of course, objective is to provide the end effector to be positioned correctly.

Today, kinematics calculations are done by homogeneous transformation matrices which are created with the help of four parameters that is called Denavit–Hartenberg parameters.

In Table 1, i shows the joint sequence. The lengths are given in meters and the angles are given in degree. The homogeneous transformation matrix can be used to obtain the forward kinematics of the robot manipulator, using the DH parameters in Eq. (1) [24, 25].

$$ {}^{i}_{i-1}T = \begin{bmatrix} cos\theta i & -cos\alpha i \cdot sin\theta i & sin\alpha i \cdot sin\theta i & ai \cdot cos\theta i \\ sin\theta i & cos\alpha i \cdot cos\theta i & -cos\theta i \cdot \sin\alpha i & ai \cdot sin\theta i \\ 0 & sin\alpha i & cos\alpha i & di \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1) $$

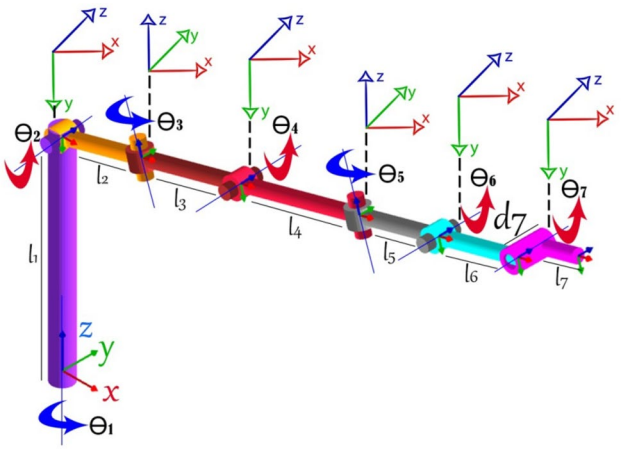

Fig. 1 The structure of robot manipulator

Table 1 D–H parameters for robot manipulator

i	a_i (m)	α_i (°)	d_i (m)	θ_i (°) (range)
1	0	−90	$l_1 = 0.5$	$-180 < \theta_1 < 180$
2	$l_2 = 0.2$	90	0	$-90 < \theta_2 < 30$
3	$l_3 = 0.25$	−90	0	$-90 < \theta_3 < 120$
4	$l_4 = 0.3$	90	0	$-90 < \theta_4 < 90$
5	$l_5 = 0.2$	−90	0	$-90 < \theta_5 < 90$
6	$l_6 = 0.2$	0	0	$-90 < \theta_6 < 90$
7	$l_7 = 0.1$	0	d7 = 0.05	$-30 < \theta_7 < 90$

$$A_{EE} = {}^{7}_{0}T = {}^{1}_{0}T \cdot {}^{2}_{1}T \cdot {}^{3}_{2}T \cdot {}^{4}_{3}T \cdot {}^{5}_{4}T \cdot {}^{6}_{5}T \cdot {}^{7}_{6}T = \begin{bmatrix} n_x & s_x & a_x & P_x \\ n_y & s_y & a_y & P_y \\ n_z & s_z & a_z & P_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2}$$

$$\begin{aligned} {}^{1}_{0}T &= \begin{bmatrix} c\theta_1 & 0 & -s\theta_1 & 0 \\ s\theta_1 & 0 & c\theta_1 & 0 \\ 0 & -1 & 0 & l_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{2}_{1}T = \begin{bmatrix} c\theta_2 & 0 & s\theta_2 & l_2c\theta_2 \\ s\theta_2 & 0 & -c\theta_2 & l_2s\theta_2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^{3}_{2}T &= \begin{bmatrix} c\theta_3 & 0 & -s\theta_3 & l_3c\theta_3 \\ s\theta_3 & 0 & c\theta_3 & l_3s\theta_3 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^{4}_{3}T &= \begin{bmatrix} c\theta_4 & 0 & s\theta_4 & l_4c\theta_4 \\ s\theta_4 & 0 & -c\theta_4 & l_4s\theta_4 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{5}_{4}T = \begin{bmatrix} c\theta_5 & 0 & -s\theta_5 & l_5c\theta_5 \\ s\theta_5 & 0 & c\theta_5 & l_5s\theta_5 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ {}^{6}_{5}T &= \begin{bmatrix} c\theta_6 & -s\theta_6 & 0 & l_6c\theta_6 \\ s\theta_6 & c\theta_6 & 0 & l_6s\theta_6 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad {}^{7}_{6}T = \begin{bmatrix} c\theta_7 & -s\theta_7 & 0 & l_7c\theta_7 \\ s\theta_7 & c\theta_7 & 0 & l_7s\theta_7 \\ 0 & 0 & 1 & d7 \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{3}$$

where ${}_{i}T^{i+1}$ is the transfer matrix of link i. ${}_{0}T^{7}$ matrix produces a Cartesian coordinate for any seven joint angles. Because the fitness function of the proposed approach is the Euclidian distance in Cartesian space between the obtained and the target points. ${}_{0}T^{7}$ can be used to calculate the Cartesian coordinate of the obtained point in the cost function.

In Eq. (2), p_x, p_y, and p_z denotes the elements of the position vector whereas n_x, n_y, n_z, s_x, s_y, s_z, a_x, a_y, a_z denote the rotational elements of the transformation matrix. In this study, only the position vector will be used to calculate the position error. The position vector equation is as follows (where s and c denote the sine and cosine functions):

2.2 Artificial bee colony (ABC) algorithm

Heuristic algorithms such as artificial neural network, simulated annealing, genetic algorithm, particle swarm optimization and firefly algorithm have the ability to easily solve NP problems which are very complex, non-linear and time-consuming problem to be solved by normal methods [26]. Recently, Artificial Bee Colony (ABC) based on search of food by honey bees, is a very popular heuristic method and was presented by Karaboğa in 2005 [27, 28].

According to ABC, this algorithm is consist of from three kinds of bees that their names are employed, onlooker and scout bee. Steps of ABC Algorithm are as follows [29, 30]:

- The initial food sources are generated randomly.
- Employed bees select a food source and return to the hive by storing nectar.
- After onlooker bees watch the waggle dance of the employed bees that came to the hive, they chose the food source with a certain probability.
- Onlooker bees that turned to the selected food sources begin to nectar storage like employed bees.
- Onlooker bees continue to nectar storage, until the limit value takes the maximum value.
- Employed bees convert into scout bees, as soon as the limit value reaches the maximum value.
- Scout bees search the new food source randomly and continue to nectar storage.
- All these steps constitute one cycle algorithm and these steps continue by the time the termination criterion is achieved.

When the basic steps of the ABC algorithm is examined in Fig. 2, in the first step, the random food sources are constructed as follows:

$$\begin{aligned} P_x = {} & \left(c\theta_1c\theta_2c\theta_3c\theta_4 - s\theta_1s\theta_3s\theta_4 - c\theta_1s\theta_2s\theta_4\right)\left(c\theta_5c\theta_6l_7c\theta_7 - c\theta_5s\theta_6l_7s\theta_7 - s\theta_5d_7 + c\theta_5l_6c\theta_6 + l_5c\theta_5\right)\left(-c\theta_1c\theta_2s\theta_3 - s\theta_1c\theta_3\right) \\ & \left(s\theta_5c\theta_6l_7c\theta_7 - s\theta_5s\theta_6l_7s\theta_7 + c\theta_5d_7 + s\theta_5c\theta_6l_6 + l_5s\theta_5\right)\left(c\theta_1c\theta_2c\theta_3s\theta_4s\theta_1s\theta_3s\theta_4c\theta_1c\theta_4s\theta_2\right)\left(-s\theta_6l_7c\theta_7 - c\theta_6l_7s\theta_7 - l_6s\theta_6\right) \\ & + c\theta_1c\theta_2\left(c\theta_3c\theta_4l_4 + l_3c\theta_3\right) - s\theta_1\left(s\theta_3c\theta_4l_4 + l_3s\theta_3\right) - c\theta_1s\theta_2l_4s\theta_4 + c\theta_1c\theta_2l_2 \end{aligned} \tag{4}$$

$$\begin{aligned} P_y = {} & s\theta_1c\theta_2c\theta_3c\theta_4 + c\theta_1s\theta_3c\theta_4 - s\theta_1s\theta_2s\theta_4)\left(c\theta_5c\theta_6l_7c\theta_7 - c\theta_5s\theta_6l_7s\theta_7 - s\theta_5d_7 + c\theta_5c\theta_6l_6 + l_5c\theta_5\right) \\ & + \left(-s\theta_1c\theta_2s\theta_3 + c\theta_1c\theta_3\right)\left(s\theta_5c\theta_6l_7c\theta_7 - s\theta_5s\theta_6l_7s\theta_7 + c\theta_5d_7 + s\theta_5c\theta_6l_6 + l_5s\theta_5\right) \\ & + \left(s\theta_1c\theta_2c\theta_3s\theta_4 + c\theta_1s\theta_3s\theta_4 + s\theta_1s\theta_2c\theta_4\right)\left(-s\theta_6l_7c\theta_7 - c\theta_6l_7s\theta_7 - l_6s\theta_6\right) + s\theta_1c\theta_2\left(c\theta_3c\theta_4l_4 + l_3c\theta_3\right) \\ & + c\theta_1\left(s\theta_3c\theta_4l_4 + l_3s\theta_3\right) - s\theta_1s\theta_2s\theta_4l_4 + s\theta_1c\theta_2l_2 \end{aligned} \tag{5}$$

$$\begin{aligned} P_z = {} & \left(-s\theta_2c\theta_3c\theta_4 - c\theta_2s\theta_4\right)\left(c\theta_5c\theta_6l_7c\theta_7 - c\theta_5s\theta_6l_7s\theta_7 - s\theta_5d_7 + c\theta_5c\theta_6l_6 + l_5c\theta_5\right) \\ & + s\theta_2s\theta_3\left(s\theta_5c\theta_6l_7c\theta_7 - s\theta_5s\theta_6l_7s\theta_7 + c\theta_5d_7 + s\theta_5c\theta_6l_6 + s\theta_5l_5\right) \\ & + \left(-s\theta_2c\theta_3s\theta_4 + c\theta_2c\theta_4\right)\left(-s\theta_6l_7c\theta_7 - c\theta_6l_7s\theta_7 - s\theta_6l_6\right) - s\theta_2\left(c\theta_3c\theta_4l_4 + l_3c\theta_3\right) - c\theta_2s\theta_4l_4 - s\theta_2l_2 + l_1 \end{aligned} \tag{6}$$

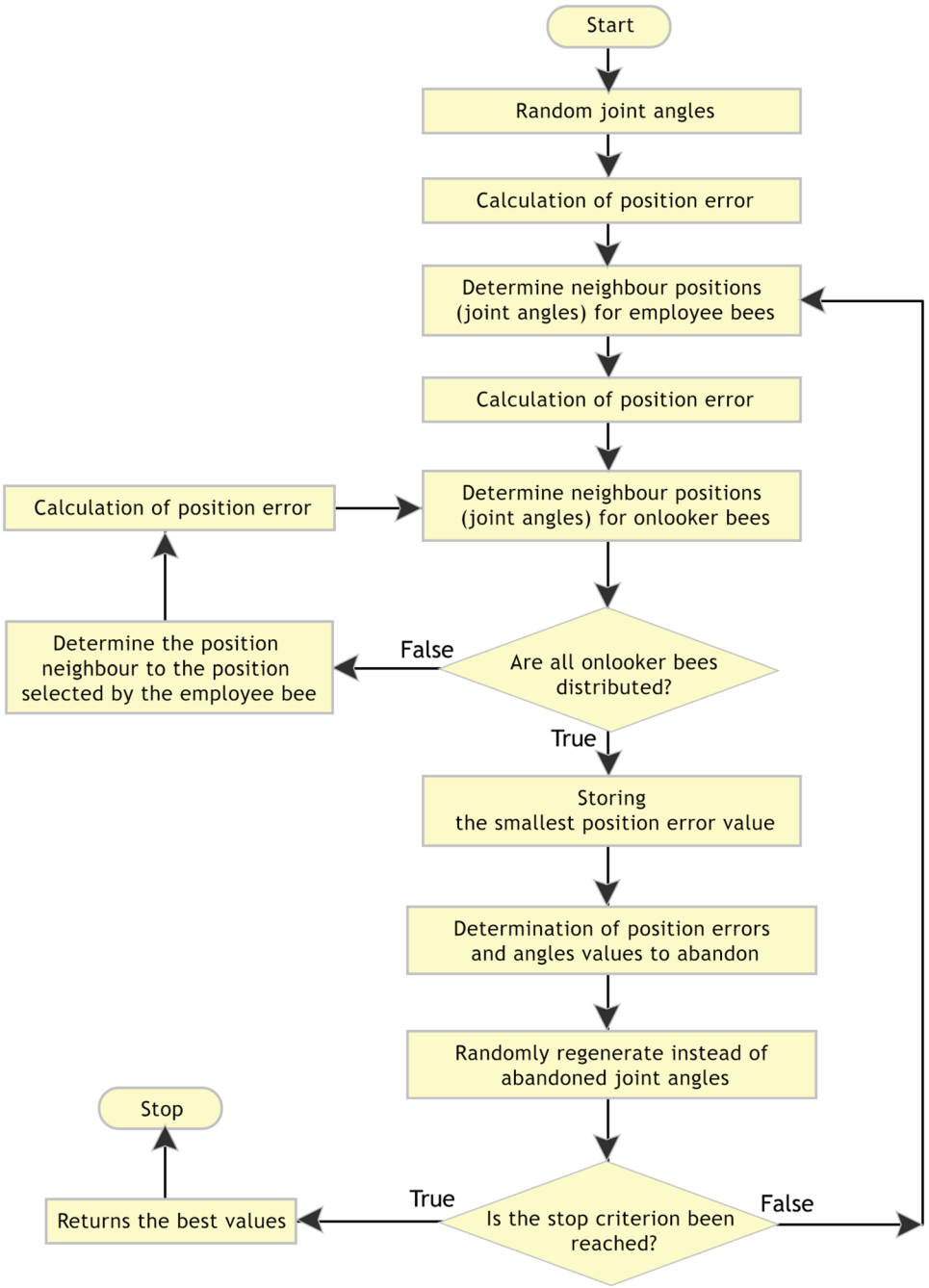

Fig. 2 Flowchart of basic ABC algorithm

$$x_{i,j} = x_j^{min} + rand(0,1)\left(x_j^{max} - x_j^{min}\right) \tag{7}$$

where i = 1…N, j = 1…D, N is the number of nectar sources, D is the number of optimization parameters, x_{min}^{j} and x_{max}^{j} are the maximum and minimum of parameter j. Initially, the value of limit parameters are reset.

Employed bees find neighbors solution in search of new sources of food and make comparisons between existing solutions for new solutions. If the new solution is kept in memory it is better than the old solution and the other is abandoned. If the new solution is not a good solution the counter of existing solutions is incremented.

$$v_{i,j} = x_{i,j} + \varphi_{i,j}\left(x_{i,j} - x_{k,j}\right) \tag{8}$$

where x_i indicates that the current solution which is selected by the employed bees. v_i is a new solution in neighborhood of x_i. k ∈ (1, N) is a randomly chosen and must be different than i, j is a random integer in the range [1, D]. $\varphi_{i,j}$ coefficients are randomly chosen value in the range [− 1.1].

Meanwhile, if the new neighborhood values exceed limit values, it is prevented by Eq. (9).

$$x_i = \begin{cases} x^i_{min}, & x_i < x^i_{min} \\ x^i_{max}, & x_i > x^i_{max} \end{cases} \quad (9)$$

After employed bees complete their research, probability values are calculated as in Eq. (10) so as to selection food sources of scout bees.

$$P_i = fit_i / \sum_{j=1}^{N} fit_i \quad (10)$$

where fit_i is normalized fitness function value. After probability values for each solution is calculated, employed bees compare with random-determined value and the probability value of the solution. If the selected probability value of the solution is greater than this value, scout bee tends to the food supply and search for new solution [31].

After all employed and scout bees complete their search, the abandonment counter of each solution is controlled. If the counter value of the solution has reached the limit value which is an important control parameter of the ABC algorithm, the employed bee that uses that resource, convert into scout bee. It is directed to a new point in the solution space using Eq. (7) than continues to search from here. All operations will continue until it reaches the maximum number of cycles [32].

2.3 Particle swarm optimization

It is a heuristic algorithm first used by Kennedy, inspired by the co-movement of birds and fishes [33]. Like other heuristic algorithms, it works by searching in a certain solution space within the frame of certain movements and stands out with its good values despite the small number of parameters [34].

As shown in Fig. 3, the PSO algorithm process depends on only two parameters, speed (Eq. 11) and position (Eq. 12) of particles. Because all the particles move to a defined position, depending on their own velocities. Therefore, each movement results in a new position [35].

$$v_{id} = v_{id} + c_1 \cdot r_1 \cdot \left(p_{best} - x_{id}\right) + c_2 \cdot r_2 \cdot \left(g_{best} - x_{id}\right) \quad (11)$$

$$x_{id} = x_{id} + v_{id} \quad (12)$$

2.4 Method

Here, the optimization problem is to find the optimum angle value for each joint with the given initial Cartesian coordinate and the target coordinate, so the end-effector of robot arm is transferred to desired location by joint angles. Obviously, the accurate calculations of joint angles values are very important. For this purpose the following scenario

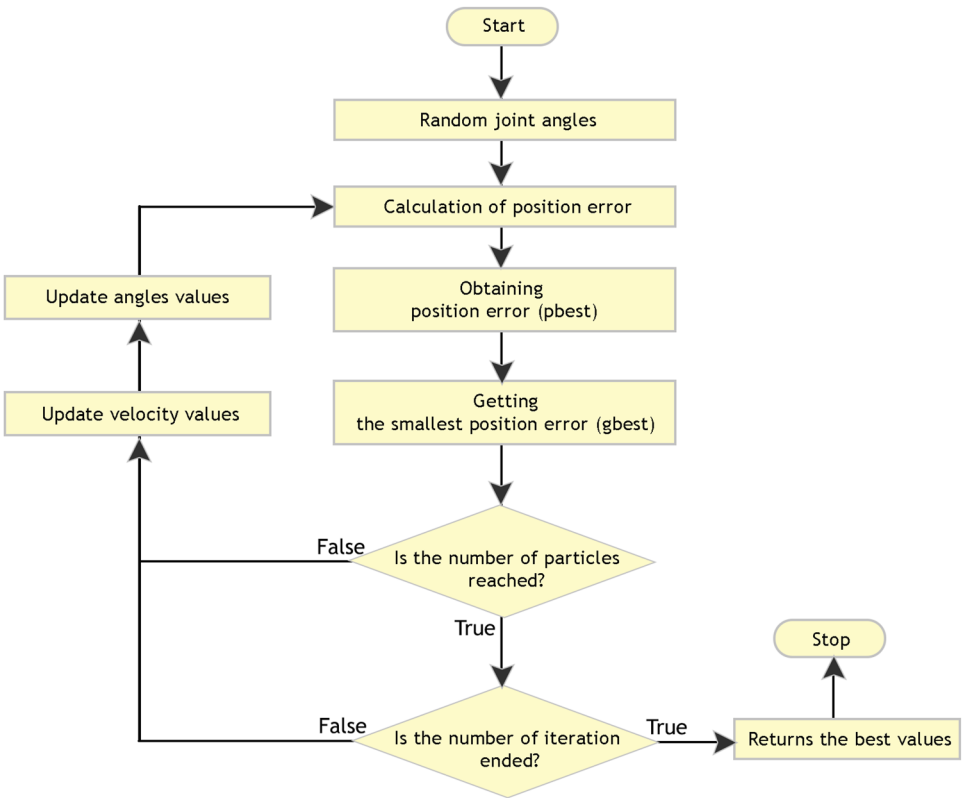

Fig. 3 Flowchart of basic ABC algorithm

is followed. Firstly, the optimal values of the 7-joint angles of the robot manipulator are found by the ABC algorithm. Then, the distance between the position of the end effector obtained by these optimal angles and the predetermined position is calculated.

$$Error = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2} \quad (13)$$

The main goal of this study is to solve this optimization problem by implementing the ABC. For this purpose, we designed a fitness function that is based on Euclidian distance given Eq. (13) between the desired location (x_2, y_2, z_2) and the current location (x_1, y_1, z_1) described. This cost can be used to calculate fitness function (Fig. 4).

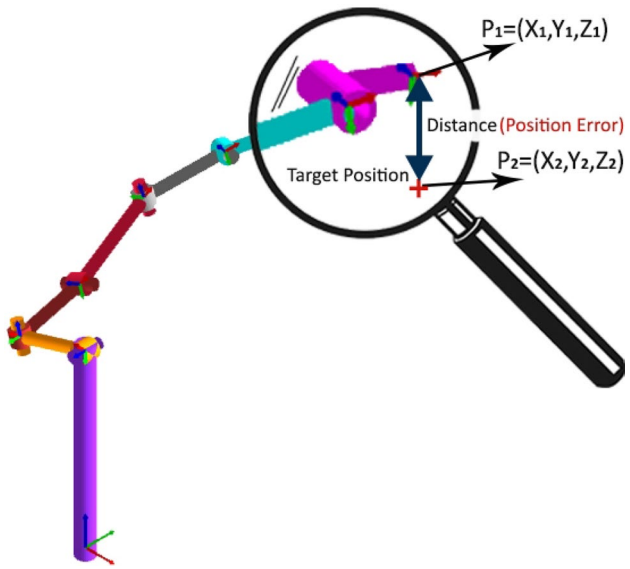

Fig. 4 Illustrating the position error

3 Simulation results

The purpose of this study is to demonstrate the effectiveness and performance of the artificial bee colony algorithm to solve the inverse kinematics solution of the 7-DOF robot arm. The initial and final positions of the end effector are seen Fig. 5a, b. However, the final position and angles appears in Fig. 5b is a position created by manual. Because the designed 7-DOF robotic arm has a redundant structure, it can be formed in the same position at different angles.

The desired position of the end effector for the robot arm is illustrated in Fig. 5b. The joint angles for the robot manipulator orientation shown in this figure are 45°, 0°, 45°, 0°, 45°, 0°, 0° from θ_1 to θ_7, respectively. However, the values obtained by the algorithm have resulted in different orientations. Because the robot manipulator used in this study has an unlimited number of solutions. The subject of this study is to position the robot arm to the destination with the minimum error using artificial bee colony algorithm and compare the result with the PSO (Table 2).

The proposed approach has been simulated in MATLAB IDE software. Figure 6 presents the performance fitness function (position error) of the ABC algorithm to solve the redundant problem of the 7-DOF robotic manipulator. As can been seen in Fig. 6, the ABC algorithm successfully search for the optimal configurations of the robotic manipulator.

According to Fig. 7, the evolved optimal solution is (θ_{1f}, θ_{2f}, θ_{3f}, θ_{4f}, θ_{5f}, θ_{6f}, θ_{7f}) = (101.190080691136°, − 9.51890813773679°, − 14.3155800174267°, − 4.35578591323361°, 39.6787526304158°, 35.9420677837833°, 72.6774064516414°).

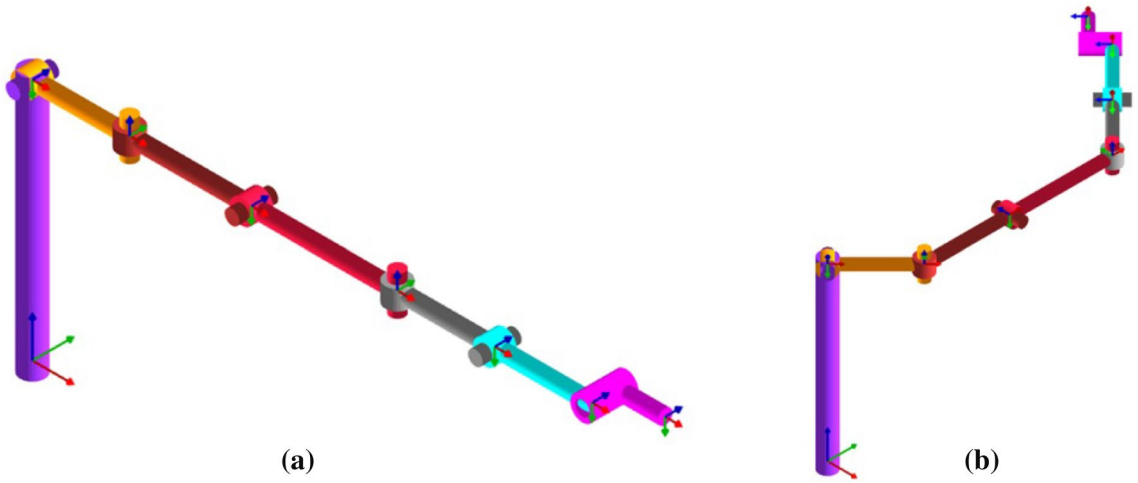

Fig. 5 Initial (**a**) and final (**b**) position of 7-DOF redundant robot arm end effector

Table 2 Joint angles obtained from calculation

Angles	Manuel	PSO	ABC
θ_1°	45	48,282	101.19
θ_2°	0	−6758	−9.52
θ_3°	45	47,497	−14.32
θ_4°	0	34,208	4.36
θ_5°	45	50,654	39.68
θ_6°	0	−60,107	35.94
θ_7°	0	−30	72.68

In this study, in order to reveal the performance of ABC algorithm, it has compared with PSO that is a swarm algorithm. As shown in Fig. 8, the ABC algorithm yielded a better result than the PSO algorithm in terms of position error.

Figure 9 shows the computation times of both the ABC and the PSO algorithm 500 iterations. Graphs clearly show that the computation times of both algorithms are very close to each other.

Figure 10 shows the orientations of joint angles resulting from position error calculated by ABC and PSO algorithms. Here, the final positions of the manipulator have been revealed in the RoboAnalyzer [36] interface through the joint angles obtained with each algorithm. That is, the

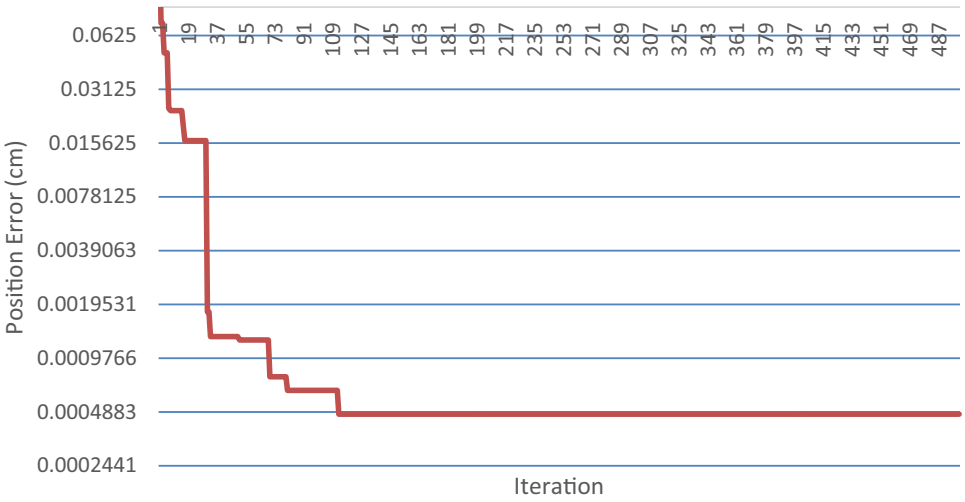

Fig. 6 Position error of end effector

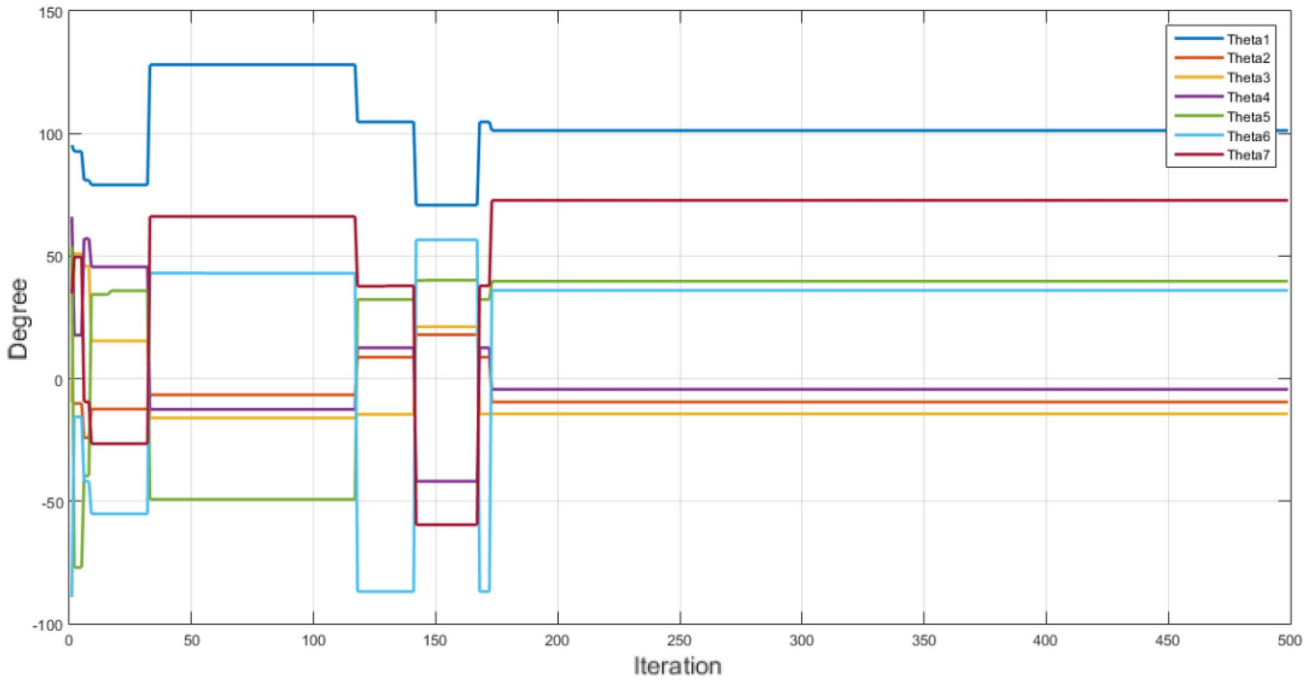

Fig. 7 The values of 7-joint angles during iteration

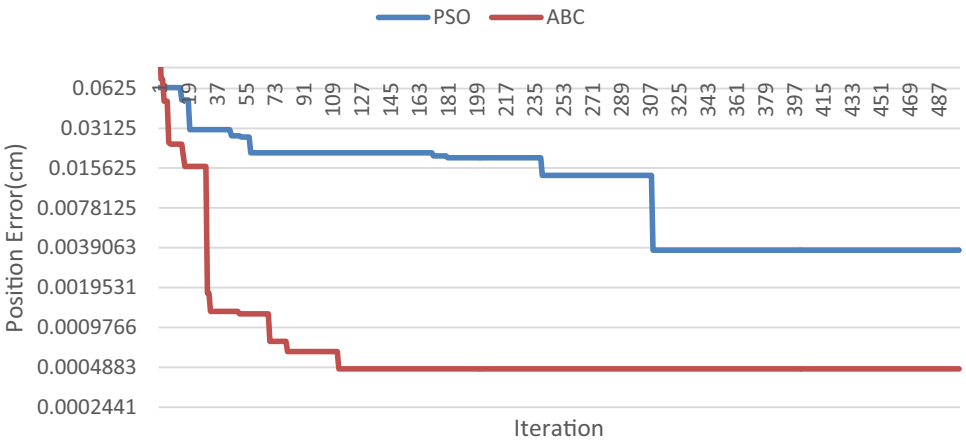

Fig. 8 ABC and PSO position error comparison

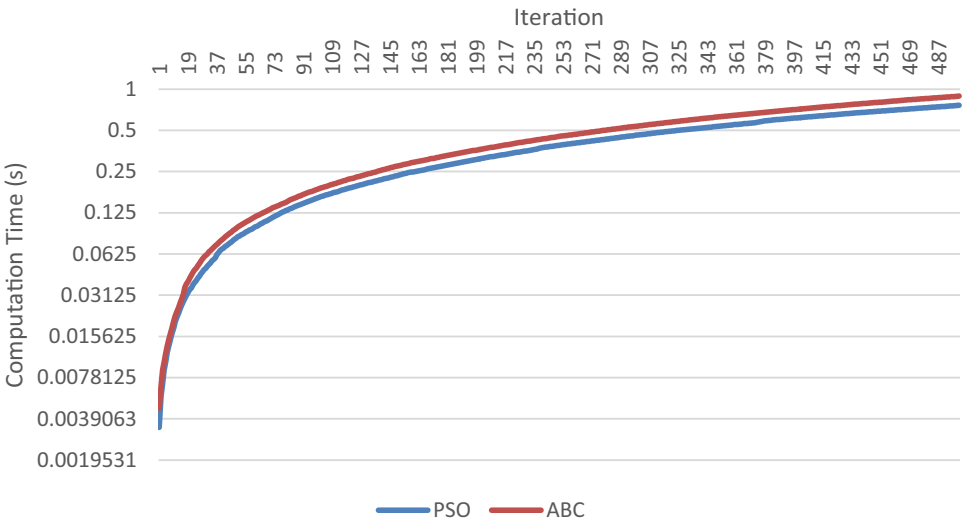

Fig. 9 ABC and PSO computation time comparison

expression of accuracy which redundant robot has numerous solutions for inverse kinematics problem is shown.

A more detailed comparison of the values obtained by ABC and PSO algorithms appears in Table 3. The ABC algorithm uses 100 population while the PSO is 300. In these algorithms, the number of population affects the quality of the solution positively while extending the solution time. Although the time for computation of the algorithm was similar to each other, the ABC solution reached a much shorter time.

In order to ensure the accuracy of the ABC algorithm, the calculation of 100 different points selected from the workspace of the robot manipulator is also performed in this article. Figure 11 shows a comparison of the position error of both algorithms for 100 different points, and Fig. 12 shows the solution time comparison.

It is clear that the ABC algorithm produces much better results than the PSO algorithm, except for the two of the results, which produce different results in both algorithms as position errors. When the calculation period is considered, the results are close to each other. However, the ABC algorithm has also shown slightly better results than the PSO algorithm.

Intelligent optimization techniques have emerged as a result of the complexity of the problems encountered today and have provided effective solutions. For this reason, these techniques have left their mark on the last 20 years, have settled in the focus of the research world and many engineering problems that take long time to solve by classical methods have been solved in a short time (Table 4).

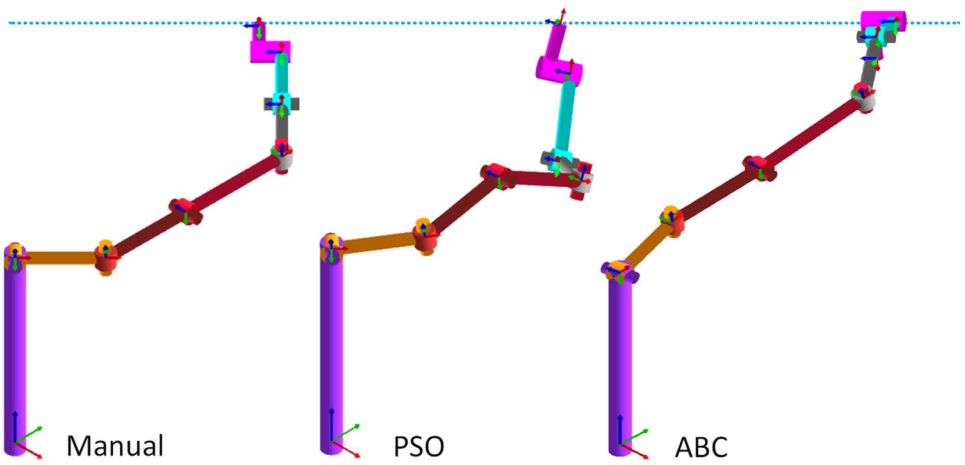

Fig. 10 According to obtained joint angles, the end effector position of the robot manipulator

Table 3 Comparison of ABC and PSO

	Population size	Max. iteration	Iteration of the solution	Position error (cm)	Solution time (s)	Computation time (s)
PSO	300	500	311	3.71e−03	0.4768	0.7612
ABC	100	500	115	4.75e−04	0.2087	0.8884

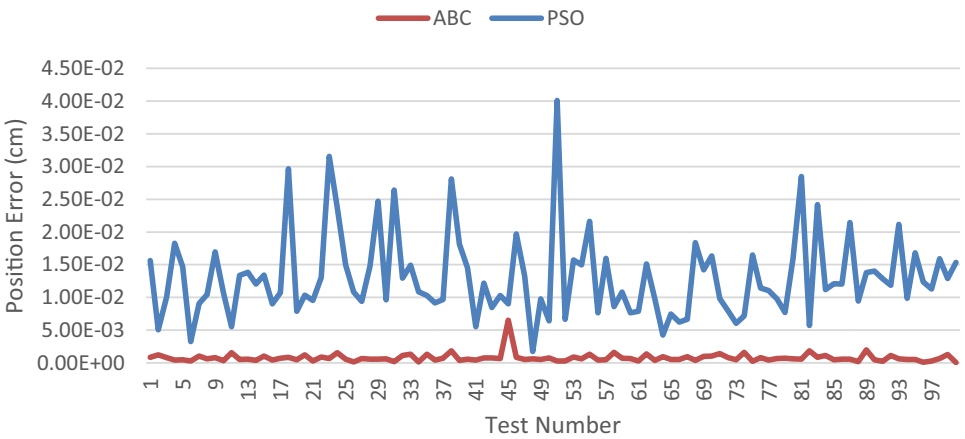

Fig. 11 Position error calculated for 100 different points

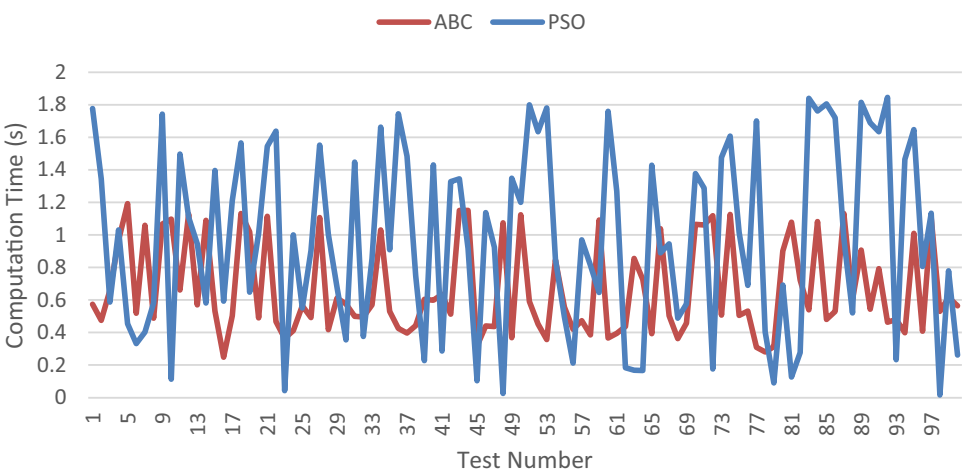

Fig. 12 Computation time for 100 different points

Table 4 Literature summary similar to this article

References	Techniques	Test manipulator	Result
[2]	New numerical algorithm (NIKA)	6-DOF industrial	NIKA technique can be used for inverse kinematic solution of robot manipulators which are not suitable for closed form solution
[3]	An evolutionary approach based on a real-coded GA	SCARA and PUMA	A GA-based approach has been proposed for multiple inverse kinematic solutions of industrial robots
[5]	A new method based on ANN	6-DOF denso robotic arm	They have got quite good results in terms of position error
[15]	Conventional GA and continuous GA	3R manipulator	As far as the inverse kinematic solution is concerned, continuous GA has produced much better results than conventional GA in many ways
[17]	PSO	7-DOF robotic manipulator	Real x, y, and z position: x = 20 cm, y = 20 cm, z = 10 cm Calculated x, y, z position: x = 19.84, y = 19.87 cm, z = 10.09 cm
[18]	GA, PSO, QPSO, GSA	4-DOF serial robot manipulator	QPSO has achieved better solutions compared to other techniques both in terms of position error and especially execution time
[20]	Firefly algorithm	3 links articulated planar system	Convergence time = 2.3394×10^{-3} s Position error = 3.116×10^{-8}
[37]	Closed form (analytically)	16 different industrial robot manipulators	In order to bring a simpler approach to the problem, the inverse kinematic problem of different class robotic manipulators used in industry has been solved analytically
[38]	Analytical and numerical techniques	6-DOF robot manipulator	Analytical and numerical methods have solved the inverse kinematics problem quickly and completely
[39]	ANN, ANFIS and GA	5-DOF robotic manipulator	Cartesian location MSE (ANN, ANFIS, GA) 0.001665 m–0.005426 m–0.000764 m Calculation time (ANN, ANFIS, GA) 0.483042 s–0.030833 s–83.123965 s

4 Conclusion

In this paper, Artificial Bee Colony algorithm has been proposed for the inverse kinematics problem of the robot arms, and it has been simulated on 7-DOF redundant robot manipulator. The most important innovation in this study is the use of a newly designed 7-jointed robot arm in the test process. In this study, artificial bee colony algorithm is used to approximate the robot manipulator to a predetermined position in the workspace with minimum error. The obtained position error and calculation times have been compared with particle swarm optimization which is another heuristic algorithm technique. In order to determine the accuracy of the algorithm used, a second scenario has been applied in 100 different tests. The simulations show the proposed ABC algorithm successfully search for the optimal joint angles of the robotic manipulator. So the ABC algorithm is a candidate method to solve inverse kinematics problem of robot arms with the high numbers of joint just like other heuristic methods. Despite all its performance, the long process stages of the algorithm and the excess of the parameters are seen as disadvantages of the artificial bee colony. If these disadvantages are eliminated or the algorithm is improved, this technique will be used more widely in the coming period. Especially after this study, it can be used to control complex robots.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Dereli S, Köker R (2016) In a research on how to use inverse kinematics solution of actual intelligent optimization method.

In: Proceeding of the international symposium on innovative technologies in engineering and science (2016)
2. Kucuk S, Bingul Z (2014) Inverse kinematics solutions for industrial robot manipulators with offset wrists. Appl Math Model 38:1983–1999
3. Kalra P, Mahapatra PB, Aggarwal DK (2006) An evolutionary approach for solving the multimodal inverse kinematics problem of industrial robots. Mech Mach Theory 41:1213–1229
4. Gao W, Liu S (2011) Improved artificial bee colony algorithm for global optimization. Inf Process Lett 111:871–882
5. Almusawi AR, Dülger CL, Kapucu S (2016) A new artificial neural network approach in solving inverse kinematics of robotic arm (denso VP6242). Comput Intel Neurosci 2016:1–10
6. Dereli S, Köker R (2017) Design and analysis of multi-layer artificial neural network used for training in inverse kinematic solution of 7-DOF serial robot. Gaziosmanpasa J Sci Res 6:60–71
7. Köker R, Çakar T, Sarı Y (2014) A neural-network committee machine approach to the inverse kinematics problem solution of robotic manipulators. Eng Comput 30:641–649
8. Fu Z, Yang W, Yang Z (2013) Solution of ınverse kinematics for 6R robot manipulators with offset wrist based on geometric algebra. J Mech Robot 5:81–87
9. Dereli S, Köker R, Öylek İ, Ay M (2019) A comprehensive research on the use of swarm algorithms in the inverse kinematics solution. J Polytech 22:75–79
10. Martin JAH, Lope J, Santos M (2009) A method to learn the inverse kinematics of multi-link robots by evolving neuro-controllers. Neurocomputing 72:2806–2814
11. Zhang D, Lei J (2011) Kinematic analysis of a novel 3-DOF actuation parallel manipulator using artificial intelligence approach. Robot Comput Integr Manuf 27:157–163
12. Dereli S, Köker R (2018) IW-PSO approach to the inverse kinematics problem solution of a 7-DOF serial robot manipulator. Sigma J Eng Nat Sci 36:77–85
13. Russell S, Dewey D, Tegmark M (2015) Research priorities for robust and beneficial artificial intelligence. AI Mag 36:105–114
14. Renzi C, Leali F, Cavazzuti M, Andrisano AO (2014) A review on artificial intelligence applications to the optimal design of dedicated and reconfigurable manufacturing systems. Int J Adv Manuf Technol 72:403–418
15. Momani S, Abo-Hammour ZS, Alsmadi OMK (2016) Solution of inverse kinematics problem using genetic algorithms. Appl Math Inf Sci 10:225–233
16. Dash KK, Choudhury BB, Khuntia AK, Biswal BB (2011) A neural network based inverse kinematic problem. In: IEEE 2011 recent advances in intelligent computational systems (RAICS), 22–24 September; Trivandrum, India. IEEE, pp 471–476
17. Huang H, Chen C, Wang P (2012) Particle swarm optimization for solving the inverse kinematics of 7-DOF robotic manipulators. In: IEEE 2012 international conference on systems, man, and cybernetics, 14–17 October, Seoul, Korea. IEEE, pp 3105–3110
18. Ayyıldız M, Çetinkaya K (2016) Comparison of four different heuristic optimization algorithms for the inverse kinematics solution of a real 4-DOF serial robot manipulator. Neural Comput Appl 27:825–836
19. Rokbani N, Alimi AM (2013) Inverse kinematics using particle swarm optimization, a statistical analysis. Int Conf Des Manuf 64:1602–1611
20. Rokbani N, Casals A, Alimi AM (2014) IK-FA, a new heuristic inverse kinematics solver using firefly algorithm. Stud Comput Intel 575:369–395
21. Köker R (2013) A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization. Inf Sci 222:528–543
22. Pham DT, Castellani M, Fahmy AA (2008) Learning the inverse kinematics of a robot manipulator using the bees algorithm. In: The IEEE 2008 International conference on industrial informatics, 13–16 July, Daejeon, Korea. IEEE, pp 493–498
23. Köker R, Çakar T (2016) A neuro-genetic-simulated annealing approach to the inverse kinematics solution of robots: a simulation based study. Eng Comput 32:1–13
24. Craig JJ (2005) Introduction to robotics mechanics and control, 2nd edn. Pearson/Prentice Hall, New York
25. Yang G, Mustafa SK, Yeo SH, Lin W, Lim WB (2011) Kinematic design of an anthropomimetic 7-DOF cable-driven robotic arm. Front Mech Eng 6:660–669
26. Çavdar T, Mohammad M, Milani RA (2012) A new heuristic approach for inverse kinematics of robot arms. Am Sci Publ 19:329–333
27. Karaboğa D (2005) An idea based on honey bee swarm for numerical optimization. Kayseri: Erciyes University, Technical report
28. Savsani V, Rao R, Vakharia DP (2010) Multi objective optimization of mechanical elements using artificial bee colony optimization technique. In: ASME 2010 Early career technical conference, 1–2 October, Atlanta, Georgia, USA. ASME, pp 146–155
29. Savsani PV, Jhala RL (2012) Optimal motion planning for a robot arm by using artificial bee colony (ABC) algorithm. Int J Mod Eng Res (IJMER) 2:2249–6645
30. Karaboga D, Gorkemli B, Ozturk C, Karaboga N (2014) A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artif Intel Rev 42:21–57
31. Wei G, Guang TZ, Qiu YW, Chun YY (2014) Improved artificial bee colony algorithm based gravity matching navigation method. MDPI Sens 14:12968–12989
32. Akay B, Karaboğa D (2012) A modified artificial bee colony algorithm for real-parameter optimization. Inf Sci 192:120–142
33. Uguz S, Sahin U, Sahin F (2015) Edge detection with fuzzy cellular automata transition function optimized by PSO. Comput Electr Eng 43:180–192
34. Kucuk S (2016) Maximal dexterous trajectory generation and cubic spline optimization for fully planar parallel manipulators. Comput Electr Eng 56:634–647
35. Dereli S, Köker R (2019) A meta-heuristic proposal for inverse kinematics solution of 7-DOF serial robotic manipulator: quantum behaved particle swarm algorithm. Artif Intel Rev. https://doi.org/10.1007/s10462-019-09683-x
36. Gupta V, Chittawadigi RG, Saha SK (2017) RoboAnalyzer: robot visualization software for robot technicians. In: Proceedings of the advances in robotics. ACM, p 26
37. Kucuk S, Bingul Z (2004) The inverse kinematics solutions of industrial robot manipulators. In: Proceedings of the IEEE international conference on mechatronics, 2004, ICM'04. IEEE, pp 274–279
38. Kucuk S, Bingul Z (2005) The inverse kinematics solutions of fundamental robot manipulators with offset wrist. In: IEEE international conference on mechatronics, ICM'05. IEEE, pp 197–202
39. El-Sherbiny A, Elhosseini MA, Haikal AY (2017) A comparative study of soft computing methods to solve inverse kinematics problem. Ain Shams Eng J 9:2535–2548