DE-based Algorithm for Solving the Inverse Kinematicson a Robotic Arm Manipulators

Tam B=UI1,2, Linh TAO1, Trung NGUYEN1, Tinh NGUYEN1, Hiroshi HASEGAWA2, Dai WATANABE2

1 School of Mechanical Engineering, Hanoi University of Science and Technology, Vietnam.
2 Department of Machinery and Control System, Shibaura Institute of Technology, Japan.

tambn@shibaura-it.ac.jp

Abstract. DE algorithm method for Solving the inverse kinematics is a fundamental problem in the robotics field especially in controlling industrial robots to move following a pre-defined trajectory. In this paper, we proposed to use a meta-heuristic optimization algorithm, namely Differential Evolution (DE), to solve the Inverse Kinematic (IK) problem for a five-degree-of-freedom (DOF) articulated robot. By considering joints' angles as continuous variables, the process of solving the Inverse Kinematic problem for the robot with the optimal algorithm has been significantly improved in terms of accuracy, execution time, standard deviation (STD) and the number of iterations needed as well. The algorithm has been applied to solve the Inverse Kinematic problem on the 5-Degree of Freedom 5R robot model. The simulation results for three case studies showed that our method can solve the inverse kinematics problem efficiently not only for minimum errors but also for smooth value of the joints' variables.

1. Introduction

Solving the inverse kinematics is an essential step in the field of robotics [1] for controlling the robot following a pre-defined trajectory. The results after solving the IK problem are joints' values that would be used to control the robot's hand completing desired tasks. Solving IK problem is one of the most critical issues in robotics before considering bigger problems such as motion planning, dynamic analysis, and control [2]. For articulated robots, IK problem is often far more complicated than forward kinetics problems. The reason for the complexity of IK problem lies in nonlinear formulas and geometric properties between the working space and the joint space. Besides, they are difficulty in the singularity and non-unique solution property of those formulas. Solutions of mathematical equations for IK problems are sometimes not suitable for real solutions.

There are several methods to solve the IK problem for an articulated robot such as geometric and iterative method. The geometric method based on geometric and trigonometric relationships to solve, while the iterative method is the method that often requires the inversion of a Jacobian matrix to solve the IK problem. These methods may have some disadvantages mentioned above.

In addition to those existing methods for solving the IK problems, there are a new trend of studies solving IK problems for articulated robots by applying meta-heuristic optimization algorithms. The primary purpose of an optimal algorithm is to minimize or maximize an objective function, and some constraints to get the optimal solution for a given problem. Meta-heuristic optimization algorithms
were applied to solve the IK problem as endpoints are in form of a single point or a whole trajectory [3]. The simulation results showed that the Particle swarm optimization (PSO) algorithm could solve the IK problem effectively. In [4], a Modified Artificial Bee Colony (ABC) was proposed and used to solve the IK problem of a 5 DOF robot arm manipulator. The simulations showed that PSO gave better results compared to the K-ABC in solving the helical path for the 5-DOF robot.

The PSO algorithm has been used in various studies to solve the IK problem of industrial robots. In [5], five variants of PSO were used to solve the IK problem of the 2-DOF double link system using forward kinematic formulas. Simulation results showed that inertia weight PSO performed better than the four other variants. However, the study just applied for a low DOF case. In [6], the authors applied PSO to solve the 6DOF of IK problems. The result showed the effectiveness of applying the meta-heuristic optimization methods to IK. However, the limit of this research is only solving IK problems for a single point (pose). In [7], the authors used a method of simulation techniques to determine the robot kinematics motion systems. In the paper [8], the author compared four different heuristic optimization algorithms for the IK solutions of a real 4-DOF serial robot manipulator. Four optimal algorithms were used in the paper: Genetic Algorithm (GA), PSO, Quantum Particle Swarm (Q-PSO), and Gravitational Search Algorithm (GSA). The robot was moved in the workspace in two different scenarios. The study shows that GA and GSA were found to be unsuitable for solving the IK problem. Besides, the study demonstrated that PSO and QPSO algorithms gave acceptance results in terms of the execution time and convenience of the solution. However, the study focused on optimal parameters such as execution time, the number of generations, and distance error without considering the value of joints' variables as the robot moves following a pre-defined trajectory. This can lead to invisible solutions that waste search resources. The reason is that joints' variables for continuous endpoints are dramatically changed. Therefore, in reality, it is difficult for the control algorithm to meet this requirement. The reason of this phenomenon is the multi solutions feature of the IK for articulated robots. At the same time, the optimal algorithm search is randomly generated in the robot workspace.

In this research, we proposed the new method for solving the IK problem for a 5-DOF manipulator by using the DE algorithm. The DE algorithm was used to solve the IK problem where the end-effectors points move along a given spiral path. The objective function is amount of total energy that the robot needs to move the robot's hand from the current point to the next point of the orbit with the constraints of distance error's endpoints as well as the smallest number of moving joints. When applying these objective functions and the limit range for joints' variables, they ensure the continuity, reducing the search domain to speed up the convergence speed.

The rest of the paper is divided into the following sections:
- Section 2 describes the background on Kinematic problem and the review of DE.
- Section 3, the proposed algorithms for path learning, and the case studies are presented.
- Section 4, the results after applying the algorithm are shown and compared.
- Finally, conclusions and development directions are outlined in section 5.

2. Background

2.1. Kinematic problem

In the forward kinematics stage, the position and orientation of the end-effector will be calculated for a set of as given value of the angle revolute joint and transmit prismatic joints. In this study, a serial redundant robot manipulator was used to evaluate the algorithm. The robot model and coordinate systems were shown in the Fig.1. As in the figure, the 5-DOF serial robot manipulator is type 5R (R: Revolute). The standard Denavit-Hartenberg (DH) method is used in this research to find the homogeneous transformation matrix [9][10]. Using the DH parameters in table 1, the homogeneous transformation matrix is calculated as in Eq. (1) and Eq. (2). The Eq. (3) is used to find the position and orientation of the end.
Figure 1. The 5-DFO robot model and coordinate systems.

Figure 2. Working space of the robot.

Table 1. D-H parameters.

| Link | $\theta_i$ | $d_i$ | $a_i$ | $\alpha_i$ |
|------|-----------|------|-------|-----------|
| 1    | $\theta_1$ | $l_1$ | 0     | $\pi/2$   |
| 2    | $\theta_2$ | 0    | $l_2$ | 0         |
| 3    | $\theta_3$ | 0    | $l_3$ | 0         |
| 4    | $\theta_4$ | 0    | $l_4$ | $\pi/2$   |
| 5    | $\theta_5$ | $l_5$| 0     | 0         |

$$i^{-1}A_j = i^{-1}A_i \cdot iA_j$$

Where $S$ and $C$ denote the sine and cosine functions.

The Homogeneous transformation matrix is calculated as in Eq. (2)

$$^0A_5 = ^0A_1 \cdot ^1A_2 \cdot ^2A_3 \cdot ^3A_4 \cdot ^4A_5$$

The position (pose) and orientation of the end-effector:

$$x_e = ^0A_5 (1:3,4)$$

$$R_{0e} = ^0A_5 (1:3,1:3)$$

After calculation, the end-effector coordinates in the working space are determined by Eq. (4):

$$x_5 = Cq_1 \left( l_2^1 * Cq_2 + l_3^1 * Cq_3 + l_4^1 * Cq_4 + l_5^1 * Sq_5 \right)$$

$$y_5 = Sq_1 \left( l_2^1 * Cq_2 + l_3^1 * Cq_3 + l_4^1 * Cq_4 + l_5^1 * Sq_5 \right)$$

$$z_5 = 1 + l_1^1 * Sq_1 + l_2^1 * Sq_2 + l_3^1 * Sq_3 + l_4^1 * Sq_4$$

Where: $S$ and $C$ denote the sine and cosine functions, $l_i$ the length of each robot link $l_i=[l_1, l_2, l_3, l_4, l_5]$, $q$ the joint variables $q=[q_1, q_2, q_3, q_4, q_5]$, $\alpha_{ij}$ is the sum of $q_i$ and $q_j$, $q_{23}=q_2+q_3$, and $q_{234}=q_2+q_3+q_4$. 

3
When solving the IK problem for a trajectory, with the endpoint position of each point on the trajectory, we need to determine a matching set of the value of joints' variables $q_i$. However, according to the formula in Eq. (4), the number of equations will be less than the number of variables (three equations and five variables); this makes it very difficult to find a unique and exact match solution. In this research, we propose the new method that we will use the DE algorithm with search domain improvement to find joints' variable values to ensure continuity as well as improve the optimization process.

2.2. Working space of the robot

The range of all joints' variables is given as in Table 2 and the working spaces of the robot are described as in the Fig.2.

| Table 2. Joint angle ranges |
|-----------------------------|
| joint | $q_1$ | $q_2$ | $q_3$ | $q_4$ | $q_5$ |
| Max limit (radian) | $-\pi$ | $-\pi$ | $-\pi$ | $-\pi$ | $-\pi$ |
| Min limit (radian) | $\pi$ | $\pi$ | $\pi$ | $\pi$ | $\pi$ |

2.3. DE algorithm

To find the solution to the constrained optimization problems in this paper, we will apply a meta-heuristic, namely DE algorithms. The reason why we chose DE is that this algorithm is the evolutionary-based method that can balance the local search and global search. Therefore, DE can provide suitable solutions. DE is initially developed by Storn and Price [11] based on the theory of evolution.

The DE algorithm applied the principle of mutation, crossover, and selection of human populations to solve problems. The core idea behind this algorithm is that the more fitness, the more change to survive of individuals in the social collection. The generation will repeat until the terminal condition is met.

3. Proposed DE Algorithms and Case Study

3.1. Proposed algorithm and objective function

As explained above, the drawback of many studies when using optimization algorithms to solve the IK problem is to focus only on the optimal running process such as computer time, the error distance, etc ... Nevertheless, they have not yet considered the continuity of the joints' variable values found. This research will focus on improving the continuity of these values by limiting the initial initialization domain of particles. Since then, the program can achieve the dual goal of increasing calculation speed, accuracy, and ensuring continuity for the value of joints' variables.

The objective function is the total amount of energy that the robot spends to move end-effector from point $j$ to point $j + 1$ as shown in the Eq. (5), while the constrain will be the error distance, as shown in the Eq. (6), and finally, the limit of joint angle variables as in Eq. (7).

**Objective function:**

$$f(q) = \sqrt{\sum_{i=1}^{5} \omega_i \cdot (q_i - q_{0,i})^2} \rightarrow \min$$

**Constraints:**
\[ C q_i + \left( l_1 * C q_1 + l_2 * C q_2 + l_3 * S q_3 \right) - x_f = 0 \]

\[ S q_1 + \left( l_1 * C q_1 + l_2 * C q_2 + l_3 * S q_3 \right) - y_f = 0 \]

\[ l_1 + l_2 * S q_2 + l_3 * S q_3 - z_f = 0 \]  

**Boundary condition:**  
\[ q_{a, i} \leq q \leq q_{a, i+1} \]  

Where: \( \omega_i \) the moment that causes the movement of each joint \( i \) from point \( j \) to point \( j+1 \) on the trajectory \((x_E, y_E, z_E)\) is the endpoint coordinates on the trajectory.

To solve the non-linear constrained optimization problem above Eq. (5,6,7), we will apply the DE algorithm to combine with the penalty method in this research. The proposed new method is shown in Fig. 3. In this algorithm, firstly, the robot hand moves from any position to the first point of the trajectory. With this first point, the initialization values for the particles are randomly selected in the full range of motion of joints. After that, the forward kinematic model is used to determine the end-effector coordinate, and finally, the objective function and some constrained functions are calculated. The primary operator of DE will repeat until the terminal condition reached, and the optimum value of the joints angle for the first point is determined. The algorithm will move to the next end-effector, in this step some parameters such as the initial angle joint \( q_0 \) and the boundary need to be updated, this process repeats for all end-effector points on the trajectory. One the algorithm finishes, the optimum solution set of joint angle values is determined.

**Figure 3.** Propose new algorithm.  
**Figure 4.** The desired end-effector.
3.2. Case study

The manipulator is required to move the end-effector following a specific trajectory. The path was selected as a spiral trajectory shown in Fig.4 and described by the Eq. (8):

\[
\begin{align*}
t & = 100 : -1 : 1; \\
xE & = 610 \ast \cos \left( t \ast 2 \ast \pi / 100 \right) \\
yE & = 610 \ast \sin \left( t \ast 2 \ast \pi / 100 \right) \\
zE & = t \ast 2 \ast \pi 
\end{align*}
\]

(8)

Where: \((x_E, y_E, z_E)\) is the coordinates of the endpoints on the trajectory. In this study, the program will run in the following cases:

3.2.1. Case 1: The method used in this case (conventional method), as shown below:

The boundary of all joint variables is set to a full range of link movement as in Eq. (9).

\[
\begin{align*}
q_{s, s} & = [-\pi, -\pi, -\pi, -\pi, -\pi] \\
q_{s, a} & = [\pi, \pi, \pi, \pi, \pi]
\end{align*}
\]

(9)

The particle initialization domain is fixed for all points on the trajectory as in Eq. (10).

\[
q_{i}^0 = \text{const for all } j = 1 : n
\]

(10)

Where \(j=1\ldots n\), \(n\) is the number of endpoints that belong to the trajectory, \(i=1\ldots 5\) is the joint variable's index.

3.2.2. Case 2: In this case ((proposed method 1)), we focus on reducing the total amount of energy that the robot spends to move end-effector from point \(j\) to point \(j + 1\)

- The boundary of each joint's variables is limited to the workspace of each joint same as case 1.

- The particle initialization domain is around the previous joint variable, as in Eq. (12). By proposed this method, the energy consumption can be reduced in comparison with case 1.

\[
q_{s, a}^{\text{a1}} = q_{i}^{j}
\]

(12)

Where \(j=1\ldots n\), \(n\) is the number of endpoints that belong to the trajectory, \(i=1\ldots 5\) is the joint variable's index. \(\Delta q\) is the fixed value of the angle.

3.2.3. Case 3: In this case (proposed method 2), we control the boundary of all variables adaptively. The purpose of this work is to control the continuity of the joint's values during the movement of the robot manipulator follow the trajectory.

- The boundary of each variable is limited to the previous trajectory's points, as shown in Eq. (9).

\[
\begin{align*}
q_{s, a}^{\text{a1}} & = q_{i}^{j} - \Delta q_{i} \\
q_{s, a}^{\text{a2}} & = q_{i}^{j} + \Delta q_{i}
\end{align*}
\]

(11)

Where \(j=1\ldots n\), \(n\) is the number of endpoints that belong to the trajectory, \(i=1\ldots 5\) is the joint variable's index.

- The particle initialization domain is updated for all points on the trajectory as in Eq. (12). The purpose of solving case 2 and case 3 is to compare the proposed solution with the conventional method (case 1) when applying DE to solve IK problems. In particular, the difference between case 2 and case 3 is that the initial points in the trajectory were generated fix and dynamically, respectively.
4. Discussion on the Simulation Results for the Case Studies

In order to compare the proposed methods with the conventional method in this research, we will use the DE algorithm of scheme DE/best/2. The algorithm will repeat the main operator until the distance errors equal to or less than the specific accuracy of 1.0E-3. In Table 3, F and CR are weight factors and Crossover ratio, respectively. NP is the population size. iter_max is the maximum number of iterations. The three case studies were programed on MATLAB R2019a with the same condition of parameter on Window10 operating system with hardware configuration as Intel® Core™ i7-7700CPU@ 3.6GHz. In the case 3 the boundary of each joint's variables is set to π/4 (rad).

Table 3. Parameters of the DE Algorithm

| F | CR | NP | iter_max | accurat |
|---|----|----|----------|---------|
| 0.5 | 0.9 | 40 | 2000 | 1.0E-3 |

The results of the applying DE for all cases are given in Fig. 5 to Fig.7 and Table 4. It can be seen that the DE algorithm produced appropriate solutions for all of the points belonging to the trajectory in all cases. However, As seen in the Table 4 and Fig.6 and Fig.7 the average energy consumption, execution CPU time, iteration in case 1 is much more than the other cases. The main reason for this was that in the case 2 and case 3 we update the values for all points on the trajectory as in Eq. (12), by proposed this technique, the energy consumption can be reduced dramatically. More specifically, as seen in the Fig.5 the case 3 achieved the solution of the of joint's values continuity. While the case 1 and case 2 got the solution the joints' value changed dramatically. Among these angular values, only the value of the q5 joint variable has a sudden change. The reason for this phenomenon is that q5 joints do not affect the endpoint position accuracy shown in Eq. (4).

Table 4. Comparative results of different cases

| Case study | Case 1 (DE) | Case 2 Proposed | Case 3 Proposed |
|------------|-------------|-----------------|-----------------|
| Energy (j) | Avg 4.84    | 2.11            | 0.26            |
|            | STD 1.23    | 2.05            | 0.45            |
| CPU time (s) | Avg 0.171 | 0.161           | 0.093           |
|            | STD 0.081   | 0.068           | 0.030           |
| Iteration  | Avg 661.41  | 633.21          | 357.19          |
|            | STD 313.38  | 262.04          | 107.10          |

Table 4 reports the results of the average error, the standard deviation of the error (STD), the average iteration, and the average CPU execution time and energy consumption of each case. As in the table, the proposed algorithm performs better results with the average error, the average iteration, and the average execution CPU time.
5. Conclusions

In this research, we proposed a new method for solving the Inverse Kinematics of 5 DOF-serial manipulator robots by using DE and proposed new methods. The 5-DOF robot was applied to move in the spiral trajectory in three different cases for validating the new proposed algorithm. The simulation results were compared in the three-case study by using the MATLAB robotics toolbox. In case 1 (convention method), 100 equidistant continuous points were generated in the path, and the DE solved the problem. In case 2 (proposed method), the boundary of each joint's variables is limited to the previous trajectory's points, and the initialization domain is fixed as same as in case 1. In case 3 (proposed method), the boundary of each joint's variables is limited to the previous trajectory's points, and the particle initialization domain is around the previous joint variable. The simulation results showed that the proposed method (case 2 and case 3) could give proper solutions for the inverse kinematics problem for a path in all aspects such as accuracy, computer time, especially the continuity of joints angle for all end-effector points. This proposed algorithm can be used for controlling the robot effectively because the robot can avoid the sudden change in the value of joints angle during the moment of a robot. In future work, this algorithm could be extended to solving more complex path in which the robot not only to move following a path but also the robot has to avoid collision with obstacles during the moving.
6. References

[1] Köker R, Öz C, Çakar T, Ekiz H (2004) A study of neural network based inverse kinematics solution for a three-joint robot. Rob Auton Syst 49:227–234

[2] Küçük S, Bingil Z (2014) Inverse kinematics solutions for industrial robot manipulators with offset wrists Appl Math Model 38:1983–1999

[3] Carlos Lopez-Franco, Jesus Hernandez-Barragan, Alma Y. Alanis, Nancy Arana-Daniel. A soft computing approach for inverse kinematics of robot manipulators. Engineering Applications of Artificial Intelligence, 74, 104-120, 2018.

[4] Ahmed El-Sherbiny, Mostafa A. Elhosseini, Amira Y. Haikal. A new ABC variant for solving inverse kinematics problem in 5 DOF robot arm. Applied Soft Computing, 73, 24-38, 2018.

[5] Adel. M. Alimi, Nizar Rokbani. Inverse Kinematics Using Particle Swarm Optimization, A Statistical Analysis. International Conference on Design and Manufacturing (IConDM): Procedia Engineering, 64. 1602-1611, 2013.

[6] Malek. A, and Tarek A. T, PSO-based Algorithm for Inverse Kinematics Solution of Robotic Arm Manipulators, 20th International Conference on Research and Education in Mechatronics (REM) May 23-24, Wel, Austria 2019.

[7] W. Widhiada and G. P. A. Suryawa, Dexterity Robot Manipulator Gripper Using Auto Tuning of PID Control, International Journal of Mechanical Engineering and Robotics Research Vol. 6, No. 6, November 2017

[8] Mustafa Ayyddiz, Kerim cetinkaya. Comparison of four different heuristic optimization algorithms for the inverse kinematics solution of a real 4-DOF serial robot manipulator, Neural Comput & Appl (2016) 27:825–836

[9] Legnani, Giovanni; Casolo, Federico; Righettini, Paolo; Zappa, Bruno (1996). "A homogeneous matrix approach to 3D kinematics and dynamics - I. Theory". Mechanism and Machine Theory. 31 (5): 573-587. doi:10.1016/0094-114X(95)00100-D.

[10] Legnani, Giovanni; Casolo, Federico; Righettini, Paolo; Zappa, Bruno (1996). "A homogeneous matrix approach to 3D kinematics and dynamics-II. Applications to chains of rigid bodies and serial manipulators".

[11] Storn R., and Price K., Differential Evolution-A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, Journal Global Optimization, Vol.11 (1997), pp.341-359.

Acknowledgments

This research is funded Project for Research Grant (Linked with KAKENHI) under the Grant no: 720MB55858.