Application of Cuckoo Search Algorithm in Optimal Solution of Robot Inverse Kinematics

Zefan Cai
1. Automation College
South China University of Technology
Guangzhou, China
2. Electronics and Information Engineering Department
Shunde Polytechnic
Foshan, China
Email: 12266423@qq.com

Daoping Huang *
Automation College
South China University of Technology
Guangzhou, China
audhuang@scut.edu.cn
* Corresponding Author

Abstract—Forward kinematics and inverse kinematics are two aspects of the robot kinematics solution, and the optimization of the inverse kinematics of robots is the most important and difficult part of robot kinematics. Parameters for each link of the robot need to be identified according to the position and orientation in both of teaching programming and interpolation. In view of the fact that to obtain the analytic solutions of the inverse kinematics of robots must meet special conditions, this paper used a general solution, cuckoo search algorithm, to get the optimization of inverse kinematics without any of special conditions. Simulation results show that, the algorithm can be effectively applied to the robot inverse kinematics, and is able to achieve sufficient accuracy.

Keywords-cuckoo search algorithm; D-H; multi-joint robot; inverse kinematics; optimization

I. INTRODUCTION

There are two aspects of kinematics of multi-joint robot, forward kinematics and inverse kinematics. The forward kinematics is to obtain the robot pose based on the robot joint variables. Instead, the inverse kinematics is to obtain the robot joint variables based on the robot pose. [1] The solution of forward kinematics can be obtained by the pose matrix, and the solution is analytical, determined, and unique. The best method for inverse kinematics is to obtain the analytical solution of joint variables according to the inverse kinematics equation, but the analytical solution can only be found out in the special conditions [2], otherwise, the optimal solution can just be obtained by various optimization algorithms.

Since 1980s, many bionic optimization algorithms developed rapidly, such as artificial neural network (ANN), genetic algorithm (GA), ant colony (ACO), simulated annealing (SA), cuckoo search algorithm (CS). Their common features are developed by simulating or revealing certain phenomena or processes of nature. They are also called intelligent optimization algorithms.

Cuckoo search algorithm (CS) [3-7] is a heuristic algorithm which is developed in 2009 by Xin-She Yang and Suash Deb. CS is based on the brood parasitism of some cuckoo species. In addition, this algorithm is enhanced by the so-called Lévy flights, rather than by simple isotropic random walks. Studies show that CS is potentially far more efficient than PSO and genetic algorithms. Because of its high efficiency, CS has caused the attention of scholars, and more and more papers about CS have been published, and has been applied successfully in many fields. [6-11] However, there are no papers about optimization of robot inverse kinematics based on CS. This paper tries to obtain the optimization of inverse kinematics with CS. Simulation result shows that, the algorithm can be effectively applied to the robot inverse kinematics, and is able to achieve sufficient accuracy.

II. D-H PARAMETERS EXPRESSION OF MULTI-JOINT ROBOT

There are several expressions of robot pose, such as Euler angle, rotation matrix and four element method, etc.. The most common one is the rotation matrix, and D-H parameters expression is the most common of the rotation matrix method. D-H is developed by Denavita and Hartenberg in 1995. [12]

In D-H, the D-H parameters of the link i must be determined firstly. There are 4 parameters: link length a_i, link twist α_i, link offset d_i and joint angle θ_i. Then homogeneous coordinate transformation matrix T_i will be calculated with D-H parameters. T_i is a primary coordinate transformation matrix, which describes the relative translation and rotation between the link coordinate systems, as in

$$T_i = \begin{bmatrix} \cos\theta_i & -\sin\theta_i\cos\alpha_i & \sin\theta_i\sin\alpha_i & a_i\cos\theta_i \\ \sin\theta_i & \cos\theta_i\cos\alpha_i & -\cos\theta_i\sin\alpha_i & a_i\sin\theta_i \\ 0 & \sin\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \quad (1)$$

For rotating joints, a_i, α_i and d_i are constant, while θ_i is variable. Instead for sliding joints, a_i, α_i and θ_i are constant, while d_i is variable.

For a 6 links multi-joint robot, the terminal pose matrix T is a relative translation between the hand (the 6th joint) and base coordinate system. T is in [12]

$$T = T_1T_2T_3T_4T_5T_6 \tag{2}$$

Assuming the position vector of the hand coordinate origin in the base coordinate is p, while the direction vectors are n, o and a, T can be described by a 4×4 matrix, as in

$$T = \begin{bmatrix} n_x & o_x & a_x & p_x \\ n_y & o_y & a_y & p_y \\ n_z & o_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Fig. 1 is the 3D simulation diagram of a low-cost 6 joints robot, and Fig. 2 is a schematic diagram of the robot and the corresponding coordinate system. The pose matrix T of the robot is as in (3). p、n、o and a in (3) are in

$$\begin{aligned}
p_x &= c_1\left[-c_{23}\left(d_4c_4s_5 - d_5s_4\right) + s_{23}\left(d_6c_5 + d_4\right) + a_1 + a_2c_2\right] \\
&\quad - s_1\left[d_6s_4s_5 + d_5c_4\right], \\
p_y &= s_1\left[-c_{23}\left(d_4c_4s_5 - d_5s_4\right) + s_{23}\left(d_6c_5 + d_4\right) + a_1 + a_2c_2\right] \\
&\quad + c_1\left[d_6s_4s_5 + d_5c_4\right], \\
p_z &= -s_{23}\left(d_4c_4s_5 - d_5s_4\right) - c_{23}\left(d_6c_5 + d_4\right) + a_2s_2, \\
n_x &= c_1\left[c_{23}\left(c_4c_5c_6 - s_4s_6\right) + s_{23}s_5c_6\right] + s_1\left(s_4c_5c_6 + c_4s_6\right), \\
n_y &= s_1\left[c_{23}\left(c_4c_5c_6 - s_4s_6\right) + s_{23}s_5c_6\right] - c_1\left(s_4c_5c_6 + c_4s_6\right), \\
n_z &= s_{23}\left(c_4c_5c_6 - s_4s_6\right) - c_{23}s_5c_6, \\
o_x &= -c_1\left[c_{23}\left(c_4c_5s_6 + s_4c_6\right) + s_{23}s_5s_6\right] - s_1\left(s_4c_5s_6 - c_4c_6\right), \\
o_y &= -s_1\left[c_{23}\left(c_4c_5s_6 + s_4c_6\right) + s_{23}s_5s_6\right] + c_1\left(s_4c_5s_6 + c_4c_6\right), \\
o_z &= -s_{23}\left(c_4c_5s_6 + s_4c_6\right) + c_{23}s_5s_6, \\
a_x &= -c_1\left(c_{23}c_4s_5 - s_{23}c_5\right) - s_1s_4s_5, \\
a_y &= -s_1\left(c_{23}c_4s_5 - s_{23}c_5\right) + c_1s_4s_5, \\
a_z &= -s_{23}c_4s_5 - c_{23}c_5.
\end{aligned} \tag{4}$$

where $s_i = \sin(\theta_i)$, $c_i = \cos(\theta_i)$, $s_{ij} = \sin(\theta_i + \theta_j)$, $c_{ij} = \cos(\theta_i + \theta_j)$.

According to (1) and (4), because all the 6 joints of the robot are rotary joints, the variables of D-H parameters are $\theta_i (i = 1 \sim 6)$. If all the θ_i are known, it is easy get T with Eq. (4). As in [2], the solution of the inverse kinematics of the 6 DOF robot can be obtained only when one of the two conditions in the Pieper criterions is met. The two Pieper criterions are:

(1) Three adjacent joint axes cross the same point;

(2) Three adjacent joint axes are parallel to each other.

The robot described in this paper can't meet the above two criterions, so the optimal solution can be obtained only by numerical method. This paper will introduce how to obtain the solution of the inverse kinematics based on CS.

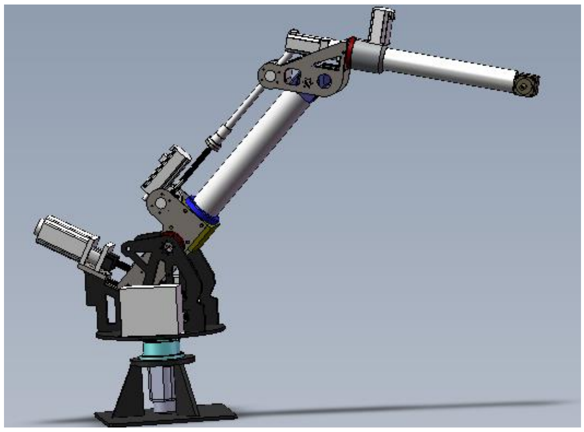

Figure 1. 6 joints robot 3D simulation diagram

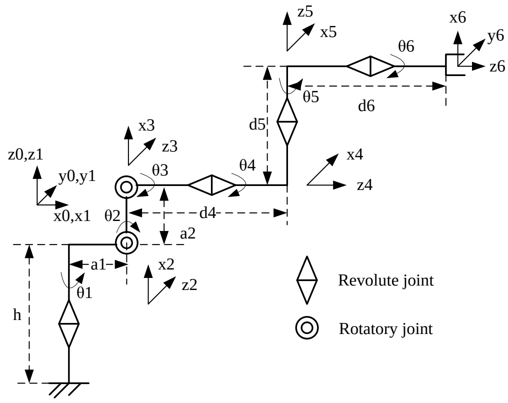

Figure 2. 6 joints robot schematic diagram

III. CUCKOO SEARCH ALGORITHM

When Xin-She Yang and Suash Deb described CS, they defined three rules as follow: [4-5]

Rule 1: Each cuckoo lays one egg at a time, and dumps it in a randomly chosen nest;

Rule 2: The best nests with high-quality eggs will be carried over to the next generations;

Rule 3: The number of available host nests is fixed, and the egg laid by a cuckoo is discovered by the host bird with a probability. In this case, the host bird can either get rid of the egg, or simply abandon the nest and build a completely new nest.

Based on these three rules, the concrete steps of CS are as follows:

(1) Initialization. Randomly generate an initial population of n nests at the position, $X_0 = \left(x_1^0, x_2^0, \cdots, x_N^0\right)$, then evaluate their fitness values F_0 and find the current global Best one.

(2) Search. Update the positions to the new positions $X_t = \left(x_1^t, x_2^t, \cdots, x_N^t\right)$ by Lévy flights, then evaluate their fitness values F_t and find the current global value. Record the global value and its corresponding position.

(3) Selection. Draw a random number r from a uniform distribution [0, 1]. Update X_{t+1} if r > p_a. Then evaluate their fitness values F_t again and find the current global

value. Record the global value and its corresponding position.

(4) Judgement. If the stopping criterion is met, then the best global position is found so far. Otherwise, return to step (2).

In the above steps, CS algorithm not only uses the Lévy flights search method, but also introduces the elite retention strategy. A good combination of local search and global search is obtained in this algorithm. The selection step has increased the diversity of positions. In the case of sufficient number of iterations, CS can converge to the global optimal solution in probability 1. [4]

In search step, Lévy flights use the random walk strategy shown as in (5). [13]

$$X_{t+1,i} = X_{t,i} + \alpha \oplus Le'vy(\beta) \tag{5}$$

Where $X_{t,i}$ represents ith solution in tth generation; α is the step size used to control the range of random search. A bigger α is better for global search and a smaller is better for local search. In the whole search, α should be from big to small. In general, α is controlled by search range, $\alpha = O(L/10)$, where L is the search range. Ref. [5] adopted the step size shown as follow:

$$\alpha = \alpha_0 \left(X_{t,i} - X_{best} \right) \tag{6}$$

Where, α_0 is constant which is 0.01 in the most time; X_{best} is the current global optimal solution.

In (5) $\oplus$ represents point multiplication; $Le'vy(\beta)$ obeys Lévy probability distribution as follow:

$$Le'vy(\beta) \sim u = t^{-1-\beta}, 0 < \beta \le 2 \tag{7}$$

Reference [5] adopted (8) to calculate $Le'vy(\beta)$.

$$Le'vy(\beta) \sim \frac{\phi \times u}{|v|^{1/\beta}}, \beta = 1.5 \tag{8}$$

Where u and v obey the standard normal distribution. ϕ is as follow:

$$\phi = \left(\frac{\Gamma(1+\beta) \times \sin(\pi \times \beta / 2)}{\Gamma\left(\left(\frac{1+\beta}{2} \right) \times \beta \times 2^{(\beta-1)/2} \right)} \right)^{1/\beta} \tag{9}$$

According to (5)~(9), the update equation in Lévy flights is shown as (10).

$$X_{t+1,i} = X_{t,i} + \alpha_0 \frac{\phi \times u}{|v|^{1/\beta}} \left(X_{t,i} - X_{t,best} \right) \tag{10}$$

In selection step, after some bad positions are discarded, the same number of new positions will be produced by (11).

$$X_{t+1,i} = X_{t,i} + r\left(X_{t,j} - X_{t,k} \right) \tag{11}$$

Where r is a scaling factor, which is a random number in uniform distribution [0, 1]. $X_{t,j}$ and $X_{t,k}$ are two solutions in tth generation.

IV. Obtain the Solution of the Multi-joint Robot Inverse Kinematics Based on CS

The solution T of forward kinematic is obtained with (1)-(4). The fitness function of CS is as follow:

$$F_i = \sum_{m=1}^{3} \sum_{n=1}^{4} \left| T_{mn} - T_{imn} \right| \tag{12}$$

where T is the target pose matrix which is the expected pose matrix of the robot in a certain time, T_i is the actual pose matrix of the certain particle, F_i is the deviation of the corresponding elements of the actual and expected pose matrix. Since the value of the last row of the matrix T_i in (2) is kept constant, last row is not within the range of computation.

The steps of CS are as follows:

(1) Randomly initialize the position of each cuckoo, save the expected pose matrix, calculate the fitness F_i with (12), and save the current best fitness and the right position.

(2) Update the position of each cuckoo with (10).

(3) Calculate the fitness F_i with (12) again, and save the current best fitness and the right position.

(4) Draw a random number r from a uniform distribution [0, 1]. Update the positions if r > p_a with (11).

(5) Calculate fitness once again, and save the current best fitness and position.

(6) If the stopping criterion is met, then the best global position is found so far. Otherwise, return to step (2).

V. Simulation and Verification

To verify the feasibility of the algorithm, the inverse kinematic matlab simulation for the 6 DOF robot shown in Fig. 1 and Fig. 2 is given. In order to make the p, n, o and a in the pose matrix T the same magnitude, take the unit of each link parameter as meter. In the simulation, $a_1 = 0.232$, $a_2 = 1.000$, $d_4 = 0.600$, $d_5 = 0.043$, $d_6 = 0.025$. The random expected pose matrix is as follow:

$$T = \begin{bmatrix} -0.612372 & 0.612372 & -0.500000 & 1.359193 \\ 0.707106 & 0.707106 & 0.000000 & 0.030405 \\ -0.353553 & 0.353553 & 0.866025 & -0.013167 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{13}$$

In this CS, the population size is 35, the dimension is 6, $\alpha_0 = 0.01$, $\beta = 1.5$, $p_a = 0.25$. After 8500 iterations, the actual pose matrix is

$$T' = \begin{bmatrix} -0.612352 & 0.612372 & -0.500000 & 1.359183 \\ 0.707106 & 0.707126 & 0.000000 & 0.030405 \\ -0.353553 & 0.353573 & 0.866005 & -0.013137 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{14}$$

In the simulation, the minimum of F_i in (12) is 0.000074. The fitness curve is shown as Fig. 3. Fig. 4 is the case of the first 200 iterations of Fig. 3.

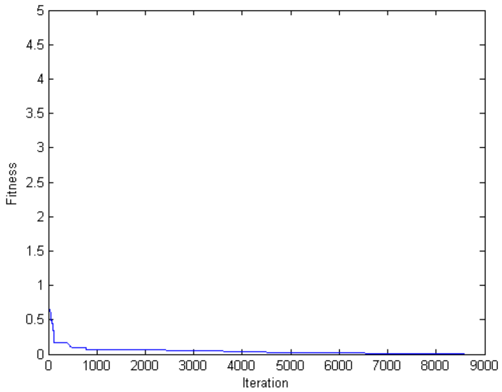

Figure 3. Fitness curve

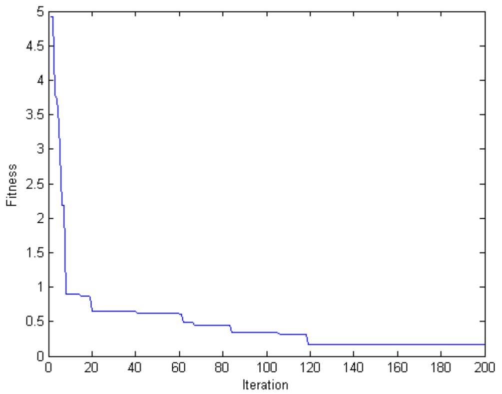

Figure 4. The first part of fitness curve

According to (13) and (14), the maximum deviation of direction dimension is 0.00001, the maximum deviation of position dimension is 0.000020m=0.020mm. The result shows that the robot pose accuracy can meets the requirements. From Fig. 3, it can be seen that the fitness is quickly close to the global optimum in exponential pattern in the first 120 iterations, and continue to move to the global optimum very slowly after 120 iterations.

VI. CONCLUSION

The forward kinematics and inverse kinematics are two aspects of robot kinematics. Based on D-H, the forward kinematics equation and pose matrix can be obtained quickly. But the solution of the inverse kinematics is not easy to obtain. If one of the two conditions in the Pieper criterions is met, the analytical solution can be obtained by using the analytical method, otherwise, the optimal solution can only be obtained by numerical solution. In the case of the known pose matrix, the CS algorithm can search the angle values of every joint, and thus the expected position and direction can be obtained accurately enough. The simulation result shows that the algorithm can be applied to inverse kinematics of multi DOF robot.

REFERENCE

[1] Xie CunXi, Zhang Tie. Robotics and its application, Press of Machinery Industry, 2012, ch. 2, pp. 44-54.

[2] Xiong YouLun. Robot technology foundation, Huazhong University of Science and Technology Press, 1996, ch. 3, pp. 46.

[3] Yang Xin-She. "Nature-Inspired Metaheuristic Algorithm", Luniver Press, 2008.

[4] Yang Xin-She. "Cuckoo Search and Firefly Algorithm Theory and Applications", Springer, 2014.

[5] Yang Xin-She, Deb Suash. "Cuckoo Search via Lévy flights", Proceeings of World Congress on Nature and Biologically Inspired Computing (NaBIC 2009), pp. 210–214. IEEE Publications, USA, 2009.

[6] Yang Xin-She, Deb Suash. "Engineering optimization by cuckoo search", Int. J. Math. Model. Num.Opt. vol. 1(4), 2010, pp. 330-343.

[7] Yang Xin-She, Deb Suash. "Multiobjective cuckoo search for design optimization", Comput. Oper. Res., vol. 40(6), 2013, pp. 1616-1624.

[8] Gandomi, A.H., Yang, X.S., Alavi, A.H. "Cuckoo search algorithm: a meteheuristic approach to solve structural optimization problems", Engineering with Computers, vol. 29(1), 2013, pp. 17-35.

[9] Gandomi, A.H., Yang, X.S., Talatahari, S., Deb, S. "Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization", Comput. Math. Appl., vol. 63(1), 2012, pp.191-200.

[10] Jiang Hao, Ruan Qi, Nian MeiLing, Zhang WenXing. "Modified cuckoo search algorithm and its application to optimization in multiple-effect evaporation", Computers and Applied Chemistry, vol. 31(11), 2014, pp. 1363-1368.

[11] He SiLu, Han JianHua. "Parameter Selection of Support Vector Regression Based on Cuckoo Search Algorithm", Journal of South China Normal University (Natural Science Edition), vol. 46(6), 2014, pp. 33-39.

[12] Han JianHai. "Industrial robot", Huazhong University of Science and Technology press, 2009.

[13] Wang LiJin, Yin YiLong, Zhong YiWen. "Cuckoo Search Algorithm with Dimension by Dimension Improvement", Journal of Software, vol. 24(11), 2013, pp. 2687-2697.