Simulation Study on Robot Zero Angle Calibration Based on VWSA-PSO Algorithm

Cai Zefan1, 2, Huang Daoping1, 2, Liu Yiqi2
1. Department Electronic and Information Engineering
   Shunde Polytechnic, Shunde, 528300, China
2. Automation College, South China University of Technology
   Guangzhou, 510640, China;

Abstract — Absolute accuracy and repeated positioning accuracy are two aspects of the robot. In general, the repeated positioning accuracy is high enough, while the absolute accuracy must be calibrated in order to reach a high level. The robot absolute accuracy is affected by the positioning error, while the positioning error is controlled by many factors, such as manufacturing precision, installation process, sensor error, kinematic parameters, etc. This paper tried to study how to calibrate the zero position angle error based on variable weight simulated annealing particle swarm optimization algorithm assumed that other factors were accurate. The simulation results show that the method is simple and accurate, and can be applied to the actual robot calibration.

Keywords - Variable weight simulated annealing particle swarm optimization algorithm; Robot; Zero angle error calibration; Absolute accuracy.

I. INTRODUCTION

There are two aspects of the robot accuracy. They are absolute accuracy and repeated positioning accuracy. Absolute accuracy is the ability of the robot to reach the set points in the working space, while repeated positioning accuracy indicates the robot’s ability to reach the teaching points again. Generally speaking, the repeated positioning accuracy is very high, can reach the level of 0.1mm ~ 0.01mm. Absolute accuracy is not high enough before calibration, just can reach the level of 0.1cm ~ 1cm. After calibration absolute accuracy can be greatly improved and can reach the level of the repeated positioning accuracy. The robot absolute accuracy is affected by the positioning error, while the positioning error is controlled by many factors, such as manufacturing precision, installation process, kinematic parameters, sensor error, etc. [1]

By now there are two kinds of methods to improve absolute accuracy: [2]

1) To improve the robot hardware, such as improving manufacturing accuracy of mechanical components, assembly and installation precision, improving the rigidity of robot joints, etc. This kind of method will increase the money of manufacturing, and professional workers must be employed in order to guarantee the assembly and the installation precision. This method can’t deal with the positioning error in future.

2) To adopting calibration technology of software. In this kind of method, the robot's actual parameters are identified to improve the robot structure model. It avoids pouring money and people. When the robot changes in future, its parameters can be identified again. The calibration technology has a good application prospect in improving the absolute accuracy of the robot, and has great research value.

There are a lot of people to research on the calibration technology. Several algorithms are adopted, such as Levenberg-Marquarde [3], least square [4], maximum likelihood estimation [5], genetic algorithm [6], simulated annealing algorithm [7], etc. This paper tries to study how to calibrate the zero angle error based on the variable weight simulated annealing particle swarm optimization algorithm assumed that other factors are accurate.

Particle swarm optimization algorithm [8] (PSO) is one of the bionic optimization algorithms. PSO is a swarm intelligence algorithm that mimics the foraging behavior of birds which was developed by two American scholars Kennedy R. and C. Eberhart J in 1995. Because PSO is simple and easy to be realized, it has caused the attention of scholars, and hundreds of papers about PSO have been published. It has been applied successfully in many fields. [9] However, there are no papers about application of robot calibration based on PSO. In this paper, nonlinear variable weight and simulated annealing mechanism is introduced to improve the standard PSO. Then, this paper tries to apply the improving PSO to calibrate the zero angle error and achieves desired results.

II. DENAVIT-HARTENBERG PARAMETERS EXPRESS_ION OF MULTI-JOINT ROBOT

There are several expressions of robot pose, such as Euler angle, four element method, rotation matrix, etc. The most common one is the rotation matrix expression, and D-H parameters expression is the most famous of the rotation matrix method. D-H is developed by Denavit and Hartenberg in 1995. [10]

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 $\alpha_i$, link offset $d_i$ and joint angle $\theta_i$. 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. $T_i$ is as follow:

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

(1)

For rotating joints, $a_i$, $\alpha_i$ and $d_i$ are constant, while $\theta_i$ is variable. Instead for sliding joints, $a_i$, $\alpha_i$ and $\theta_i$ 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 terminal joint (the 6th joint) and base coordinate system. $T$ is as follow:

$$
T = T_1T_2T_3T_4T_5T_6
$$

(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 follow:

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

(3)

Fig. 1 (a) is the three-dimensional (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.

![Figure 1. 6 joints robot 3D simulation diagram](image)

![Figure 2. 6 joints robot schematic diagram](image)

The pose matrix $T$ of the robot is as Eq. (3). $p$, $n$, $o$ and $a$ in this equation are as follow:

$$
p_x = c_i[-s_{23}(d_ic_5s_1 - d_is_4) + s_{23}(d_is_1 + d_4) + a_1 + a_2c_5]
$$

$$
-p_y = s_i[-c_{23}(d_is_5s_1 - d_sc_4) + s_{23}(d_is_1 + d_4) + a_1 + a_2c_5]
$$

$$
p_z = s_{23}(d_is_4s_1 - d_sc_4) - c_{23}(d_is_1 + d_4) + a_2s_5,
$$

$$
n_x = c_i[s_{23}(c_4c_5s_1 - s_4s_6) + s_{23}s_5c_6] + s_i(s_4c_5c_6 + c_4s_6),
$$

$$
n_y = s_i[s_{23}(c_4c_5s_1 - s_4s_6) + s_{23}s_5c_6] - c_i(s_4c_5c_6 + c_4s_6),
$$

$$
n_z = -s_{23}(c_4c_5s_1 - s_4s_6) - c_{23}s_5c_6,
$$

$$
o_x = -c_i[s_{23}(c_4s_5s_1 + s_4c_6) + s_{23}s_5s_6] - s_i(s_4c_5s_6 - c_4c_6),
$$

$$
o_y = -s_i[s_{23}(c_4s_5s_1 + s_4c_6) + s_{23}s_5s_6] + c_i(s_4c_5s_6 + c_4c_6),
$$

$$
o_z = -s_{23}(c_4s_5s_1 + s_4c_6) + c_{23}s_5s_6,
$$

$$
a_x = -c_i[c_2s_3s_5c_6 + s_2s_3c_6],
$$

$$
a_y = -s_i[c_2s_3s_5c_6 + s_2s_3c_6] + c_i(s_2s_3c_6 + c_2s_3s_6),
$$

$$
a_z = -s_2s_3s_5c_6 - c_2s_3c_6,
$$

where $s_i = \sin(\theta_i)$, $c_i = \cos(\theta_i)$, $s_y = \sin(\theta_i + \theta_j)$, $c_y = \cos(\theta_i + \theta_j)$.

According to Eq. (1) and (4), as all the 6 joints of the robot are rotary joint, the variables of D-H parameters are $\theta_i$ ($i = 1 \sim 6$). If all the $\theta_i$ are known, it is easy get $T$ with Eq. (4). When a robot is assembled, $a_i$, $\alpha_i$ and $d_i$ are fixed, and we assume they are accurate, while $\theta_i$ are variables which can be determined by angle encoder. Since angle encoder always has zero error, $\theta_i$ in Eq. (1) and (4) must be replaced by $\theta_i'$, where $\theta_i' = \theta_i + \Delta \theta_i$, and $\Delta \theta_i$ is called zero error modifier. In order to improve positioning accuracy,
Δθi must be found. This paper will introduce how to get Δθi based on improved PSO.

III. BASIC PARTICLE SWARM OPTIMIZATION ALGORITHM

Particle swarm optimization algorithm is one of the bionic optimization algorithms. PSO is a swarm intelligence algorithm that mimics the foraging behavior of birds which was developed by two American scholars Kennedy R. and C. Eberhart J in 1995. The particle swarm optimization algorithm is realized by iteration. The particles update their speeds and positions by tracking their own optimal value pBest and global optimal value gBest in each iteration. The speed update equation is as Eq. (5) and the position update equation is as Eq. (6).

\[ v(t+1) = W \cdot v(t) + c_1 \cdot \alpha \cdot [pBest(t) - x(t)] + c_2 \cdot \beta \cdot [gBest(t) - x(t)] \]  
\[ x(t+1) = x(t) + \gamma \cdot v(t+1) \]  
where W is the coefficient of inertia weight, \( c_1 \) is the weight coefficient of the optimal value of the particle itself, which represents the particle's own cognition, and is usually set to 2. \( c_2 \) is the weight coefficient of the global optimal value of the whole population, which represents the knowledge of the entire group of particles, and is also usually set to 2. \( \alpha \) and \( \beta \) are random factors that are subject to the uniform distribution \([0, 1]\). \( \gamma \) is a constraint factor, and usually set to 1.

IV. OBTAIN ROBOT ZERO ANGLE ERRORS BASED ON VWSA-PSO

Although the concept of PSO algorithm is simple and needs fewer parameters to adjust, and easy to be realized, but it is easy to converge to local optima, so PSO is often used together with other optimal algorithms in order to improve the ability to convergence to the global optimum.

Inertia weight W in the PSO plays pivotal role. A big W is better for search range, and a small W better for fine search. Generally, in the evolutionary algorithm, the value of W must change from big to small, and the value will be restricted in a certain range. Ref. [11] described the some mechanisms of W. One of them is introduced in this paper. The mechanism is as follow:

\[ W = W_{\text{min}} \times \left( \frac{W_{\text{max}}}{W_{\text{min}}} \right)^{1/(t+10 \cdot \text{CurCount} / \text{LoopCount})} \]  
Where \( W_{\text{min}} \) is the maximum value of W, \( W_{\text{min}} \) is the minimum value of W, LoopCount is the maximum iteration, CurCount is the current iteration. When \( W_{\text{max}} = 0.95, W_{\text{min}} = 0.4 \) and LoopCount=500, the curve of W in Eq. (7) is as Fig. 3.

Simulated annealing algorithm is a random combination optimization method which developed in the early 1980s. It simulates the thermodynamic process of metal temperature cooling, and is widely used in combination optimization problem. The initial temperature is determined by the simulated annealing algorithm in the optimization, randomly select an initial state and investigate the objective function of the state; Attach a small perturbation to the current state and calculate the target function value for the new state; In the whole cooling process, the better state will be accepted in probability 1, while worse state will also be accepted in some probability which is less than 1. If the start temperature is high enough and the temperature falls slowly enough, SA can converge to the global optimum with probability 1. Because of its ability to accept the worse sample in some probability and improve the sample's diversity, it has the ability to jump out of the local optimal solution. [12]

The researchers combine the strong global convergence of SA with the simple and efficient performance of PSO, and get the simulated annealing particle swarm optimization (SA-PSO), and successfully apply it to many subjects. [12-14]

In this paper, the nonlinear decreasing inertia weight mechanism is introduced to further improve SA-PSO. The new algorithm can be called variable weight simulated annealing particle swarm optimization algorithm (VWSA-PSO). The solution T of forward kinematic is obtained with Eq. (1) ~ (4). The fitness function of VWSA-PSO is as Eq. (8) or Eq. (9).

\[ p_i = \sum_{n=1}^3 | T_{\text{act}} - T_{\text{ref}} | \]  
\[ p_i = \sum_{n=1}^3 \sum_{m=1}^4 | T_{\text{act}} - T_{\text{ref}} | \]  
where T is the target pose matrix which is the expected pose matrix of the robot in a certain time, \( T_{\text{act}} \) is the actual pose matrix of the certain particle, \( p_i \) is the deviation of the corresponding elements of the actual and expected pose matrix. Since the value of the last row of the matrix \( T_{\text{act}} \) in Eq. (2) is kept constant, last row is not within the range of computation.

The steps of SA-PSO are as follows [15, 16]:

![Figure 3. W curve](image-url)
(1) Randomly initialize the position and speed of each particle, calculate the fitness of each particle and regard the fitness as the local particle optimal fitness \( p_i \). Take the smallest \( p_i \) as the global optimal fitness \( p_{g} \).

(2) Initialize the maximum and minimum value of \( W \), the start temperature \( T \) and cooling rate \( K \).

(3) The optimal particle that satisfies the lower equation is used as an alternative for the global optimum.

\[
\left( \min(p) - p_{\text{best}} \right) < \epsilon \cdot \left( \frac{\min(p) - p_{\text{min}}}{2} \right) > \text{rand} \tag{10}
\]

where \( \epsilon \) is a decimal in \([0, 1]\) which is to improve the diversity of the population. \( \text{rand} \) is a random factor that is subject to the uniform distribution \([0, 1]\). \( T \) is the annealing temperature.

(4) Update the speed and position with Eq. (5) and (6), and make sure that the speed and position are not exceed the maximum.

(5) Calculate the new fitness \( p_{\text{Temp}} \) of each particle, and consider the \( p_{\text{Temp}} \), meeting Eq. (11) as the new \( p_i \).

\[
(p_{\text{Temp}}, -p_i) < \epsilon \cdot \left( \frac{p_{\text{Temp}} - p_{\text{min}}}{2} \right) > \text{rand} \tag{11}
\]

where, \( \epsilon \) and \( \text{rand} \) are the same as in Eq. (10).

(6) Update \( p_{g} \).

(7) Take simulated annealing operation.

(8) Update \( W \) with Eq. (7)

(9) If the stopping criterion is met, then the best global fitness is also near the actual value. In the algorithm with Eq. (8), the optimal value of the 6th joint zero angle error can’t be searched. The characteristic of the table 2 is basically similar to table 1. When \( N=7 \), \( F' / F \) is the smallest, the search value of the first 5 zero errors are close to the actual values, while the 6th is random. The reason why the 6th is random is that the 6th joint is a revolute joint, its revolute angle \( \theta_6 \) just affects its direction but not affects its position, and Eq. (8) can only represents the error of position. In the algorithm with Eq. (8), the optimal value of the 6th joint zero angle error can’t be searched. When using Eq. (9), we need a double theodolite tester to get the robot terminal position coordinate and direction, and it is complex and expensive, but all the zero angle errors can be searched.

V. SIMULATION AND VERIFICATION

In simulation, 6 small random zero angle errors \( \Delta \theta_i \) are generated from a uniform distribution \([-0.1, 0.1]\). For comparison, here they are set at \([-0.1336 -0.1856 +0.0116 +0.1728 -0.0688 -0.1301] \) (Unit: \(^\circ\)). \( N \) teams of joint thetas \( \theta_i \) are generated from a uniform distribution \([-180, 180] \) (Unit: \(^\circ\)). Also \( \theta_i \) are set at fixed values as follow:

\[
\begin{bmatrix}
178.7657 & -139.9853 & 54.0515 & -5.7608 & -74.7103 & -10.6793 \\
-57.9619 & -51.6066 & 49.3376 & 15.4147 & 46.4832 & 12.6455 \\
-136.3614 & -104.7952 & 97.0605 & -121.9867 & 22.6131 & 168.6381 \\
\end{bmatrix}
\]

There are 10 rows in the matrix above, representing 10 teams of joint thetas. When \( N \) changes from 1 to 10, the first \( N \) rows are taken as the \( N \) teams of joint thetas. Then the pose matrices are calculated with Eq. (4) based on \( \Delta \theta_i \) and \( \theta_i \), which are considered as the robot pose matrices.

In the VWSA-PSO, the population size is 40, the dimension is \( 6 \), \( c_1 = c_2 = 2 \), \( \epsilon = 0.01 \), \( W_{\text{min}} = 0.95 \), \( W_{\text{max}} = 0.4 \), start temperature \( T_0 = 100 \), cooling rate \( K = 0.99 \), iterations is 500. The program is run 5 times at each \( N \), and the averages of the 5 results are shown as table 1 and table 2, where \( \Delta \theta_i (i = 1 - 6) \) are zero angle errors searched in VWSA-PSO, \( F \) are the averages of distances between each pair of theoretical positions and actual positions before calibration, while \( F' \) are the averages after calibration. \( F' / F \) are the multiples of the position distance errors before and after calibration.

According to table 1, when \( N=8 \), \( F' / F \) is the smallest, the search value of the first 5 zero errors are close to the actual values, while the 6th is random. The reason why the 6th is random is that the 6th joint is a revolute joint, its revolute angle \( \theta_6 \) just affects its direction but not affects its position, and Eq. (8) can only represents the error of position. In the algorithm with Eq. (8), the optimal value of the 6th joint zero angle error can’t be searched. The characteristic of the table 2 is basically similar to table 1. When \( N=7 \), \( F' / F \) is the smallest, the search value of the first 5 zero errors are close to the actual values, while the 6th is also near the actual value.

Table 1. VWSA-PSO simulation result (with Eq. 8)

<table>
<thead>
<tr>
<th>( \Delta \theta_i (i = 1 - 6) )</th>
<th>( N )</th>
<th>( F )</th>
<th>( F' )</th>
<th>( F' / F )</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.1478 -0.2371</td>
<td>1</td>
<td>0.9895</td>
<td>0.0013</td>
<td>0.001314</td>
</tr>
<tr>
<td>0.1538 0.1095 -0.0371 0.1201</td>
<td>2</td>
<td>0.9175</td>
<td>0.0038</td>
<td>0.004142</td>
</tr>
<tr>
<td>-0.1344 -0.1855 0.0100 0.1849</td>
<td>3</td>
<td>0.9128</td>
<td>0.0010</td>
<td>0.00196</td>
</tr>
<tr>
<td>-0.0679 -0.0690 0.1336 -0.1858</td>
<td>4</td>
<td>0.8873</td>
<td>5.4612e-04</td>
<td>0.000615</td>
</tr>
<tr>
<td>-0.0113 -0.1723 0.0868 0.0843</td>
<td>5</td>
<td>0.9286</td>
<td>7.3088e-04</td>
<td>0.000787</td>
</tr>
<tr>
<td>0.1317 -0.1731 0.0681 0.1671</td>
<td>6</td>
<td>0.9117</td>
<td>5.2769e-04</td>
<td>0.000579</td>
</tr>
<tr>
<td>-0.1356 -0.1856 0.0118 0.1721</td>
<td>7</td>
<td>0.8927</td>
<td>5.1186e-04</td>
<td>0.000573</td>
</tr>
<tr>
<td>-0.0633 0.2828 0.0136 -0.1856</td>
<td>8</td>
<td>0.8760</td>
<td>4.6911e-08</td>
<td>5.36E-08</td>
</tr>
<tr>
<td>0.0116 0.1728 0.0088 -0.0748</td>
<td>9</td>
<td>0.8700</td>
<td>6.7337e-05</td>
<td>7.4E-05</td>
</tr>
<tr>
<td>-0.1336 -0.1856 0.0016 0.1728</td>
<td>10</td>
<td>0.8655</td>
<td>0.0030</td>
<td>0.003466</td>
</tr>
</tbody>
</table>

In this algorithm, the fitness curve throughout the iteration is shown in Fig. 4 and Fig. 5. In Fig. 4 and Fig. 5,
the thick curve represents the global optimal value, and the thin curve is the surrogate value of the global optimum value; Fig. 4 is the simulation result for all the 500 iterations, and Fig. 5 is the result for the first 100 iterations. From Fig. 4, it can be seen that the fitness is quickly close to the global optimum in exponential pattern in the first 150 iterations, and continue to move to the global optimum very slowly after 150 iterations; From Fig. 5, it can be seen that the alternative value of the global optimal value presents multiple peaks and troughs. The simulation result shows that VWSA-PSO can effectively jump out of the local optimal value and has the strong ability to achieve the global optimal value.

### Table 2. VWSA-PSO Simulation Result (with Eq. 9)

<table>
<thead>
<tr>
<th>$\Delta \theta_i$ ($i = 1 \ldots 6$)</th>
<th>$N$</th>
<th>$F$</th>
<th>$F'$</th>
<th>$F''$ / $F$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$-0.1420$ $-0.0988$ $0.2118$ $0.1352$ $0.1518$ $-0.0867$</td>
<td>1</td>
<td>1.0146</td>
<td>0.0141</td>
<td>0.013897</td>
</tr>
<tr>
<td>$-0.1337$ $-0.1846$ $0.0102$ $0.1724$ $-0.1333$ $-0.1855$</td>
<td>2</td>
<td>0.9394</td>
<td>0.0087</td>
<td>0.009261</td>
</tr>
<tr>
<td>$-0.0633$ $-0.1371$ $-0.1336$ $-0.1854$ $0.0104$ $0.1731$</td>
<td>3</td>
<td>0.9339</td>
<td>0.0051</td>
<td>0.005461</td>
</tr>
<tr>
<td>$-0.1602$ $-0.1858$ $0.0116$ $0.1728$ $0.0087$ $-0.1180$</td>
<td>4</td>
<td>0.9089</td>
<td>0.0183</td>
<td>0.020134</td>
</tr>
<tr>
<td>$-0.1337$ $-0.1857$ $0.0111$ $0.1747$ $-0.1337$ $-0.1857$</td>
<td>5</td>
<td>0.9503</td>
<td>0.0059</td>
<td>0.006209</td>
</tr>
<tr>
<td>$-0.1336$ $-0.1856$ $0.0116$ $0.1728$ $0.0088$ $-0.1761$</td>
<td>6</td>
<td>0.9325</td>
<td>0.0117</td>
<td>0.012547</td>
</tr>
<tr>
<td>$-0.1336$ $-0.1856$ $0.0116$ $0.1728$ $0.0088$ $-0.1397$</td>
<td>7</td>
<td>0.9112</td>
<td>0.0011</td>
<td>0.001207</td>
</tr>
<tr>
<td>$-0.1340$ $-0.1861$ $0.0124$ $0.1767$ $0.0084$ $-0.1210$</td>
<td>8</td>
<td>0.8950</td>
<td>0.0106</td>
<td>0.011844</td>
</tr>
<tr>
<td>$-0.1338$ $-0.1865$ $0.0135$ $0.1694$ $0.0084$ $-0.1590$</td>
<td>9</td>
<td>0.8896</td>
<td>0.0054</td>
<td>0.006070</td>
</tr>
<tr>
<td>$-0.1336$ $-0.1856$ $0.0116$ $0.1729$ $0.00692$ $-0.1529$</td>
<td>10</td>
<td>0.8842</td>
<td>0.0095</td>
<td>0.010744</td>
</tr>
</tbody>
</table>

IV. Conclusion

The absolute accuracy of the robot is affected by many aspects. The joints’ zero angle errors are one of them. With VWSA-PSO introduced in this paper, the optimal values of the zero angle errors can be found quickly. In this algorithm, just a few position messages need to be obtained. It is easy to operate. This method has good generality. It not only can be applied to the robot calibration all with revolute joints, but also can be applied to the robot calibration with sliding joints.

REFERENCES

[11] Chen GuiMin, Jia JianYuan, Han Qi. Study on the Strategy of Decreasing Inertia Weight in Particle Swarm Optimization


Particle Swarm Optimization, Mechanical Drive, vol. 34(10), 2010, pp. 43-47.
