RESEARCH ARTICLE

The optimized algorithm based on machine learning for inverse kinematics of two painting robots with non-spherical wrist

Xiaoqi Wang*, Jianfu Cao, Lerui Chen, Heyu Hu

Faculty of Electronic and Information Engineering, Xi'an Jiaotong University, Xi'an, Shaanxi, China

* wangxiaoqi1031@163.com

Citation: Wang X, Cao J, Chen L, Hu H (2020) The optimized algorithm based on machine learning for inverse kinematics of two painting robots with non-spherical wrist. PLoS ONE 15(4): e0230790. https://doi.org/10.1371/journal.pone.0230790

Editor: Chen Lv, Nanyang Technological University, SINGAPORE

Received: September 27, 2019

Accepted: March 8, 2020

Published: April 3, 2020

Data Availability Statement: All relevant data are within the manuscript and its Supporting Information files.

Funding: National Natural Science Foundation (NNSF) of China [NO.61573272].

Competing interests: The authors have declared that no competing interests exist.

Abstract

This paper studies the inverse kinematics of two non-spherical wrist configurations of painting robot. The simplest analytical solution of orthogonal wrist configuration is deduced in this paper for the first time. For the oblique wrist configuration, there is no analytical solution for the configuration. So it is necessary to solve by general method, which cannot achieve high precision and high speed as analytic solution. Two general methods are optimized in this paper. Firstly, the elimination method is optimized to reduce the solving speed to 20% of the original one, and the completeness of the method is supplemented. Based on the Gauss damped least squares method, a new optimization method is proposed to improve the solving speed. The enhanced step length coefficient is introduced to conduct studies with the machine learning correlation method. It has been proved that, on the basis of ensuring the stability of motion, the number of iterations can be effectively reduced and the average number of iterations can be less than 5 times, which can effectively improve the speed of solution. In the simulation and experimental environment, it is verified.

Introduction

For the 6R painting robot, non-spherical wrists are usually adopted because of operation requirements. It is mainly divided into orthogonal non-spherical wrist and oblique non-spherical wrist. For the orthogonal non-spherical wrist configuration, three adjacent axes are parallel to each other to meet Pieper criterion, there is an analytical solution, and the simplest analytical solution is given in this paper. For the oblique non-spherical wrist configuration, the Pieper criterion is not satisfied and there is no analytical solution. Among many methods, the elimination method has the most complex calculation process, but it has the highest accuracy and can get all the solutions under the same terminal position. In this paper, the elimination method was optimized to improve the speed of solution and to supplement the integrity of the method. Gaussian damped iteration method is derived by the most widely used damped least squares method. The Gauss damped iteration method is an efficient and stable method. It has the advantages of fast convergence of Newton iteration method, smooth motion of damping method and less error after optimizing damping factor. But in the embedded environment, it cannot achieve the same speed as the analytic solution of general configuration. On the basis of the algorithm, this paper introduced the enhanced step length coefficient, analyzed the affecting factors of the coefficient, studied the coefficient through machine learning classification and regression methods, and compared the performance of various models. After optimization, the average number of iterations was less than 4 times, which effectively improved the solving speed. Next, we will introduce robotic products using two configurations, some general new methods and optimization methods in other papers, and machine learning methods for studying the enhanced step length coefficient.

Both configurations are common in robot products. For the orthogonal non-spherical wrist, Yaskawa's EPX series with Lemma wrist, Fanuc's M-710iC/50E, Kawasaki's K-series with BBR wrists, etc. For the oblique non-spherical wrist, Yaskawa's EPX and MPX series, Fanuc's P-50iB and P-250iB, Kawasaki's K Series with 3R wrist, Comau's NJ4 series with Hollow wrist, Staubli's TX250, etc.

[1] and [2] proposed an elimination method for inverse kinematics of 6R robots, which simplified the inverse kinematics of the robots into a 16-degree polynomial problem with one variable. Then the matrix eigenvalue decomposition was used to replace solving equation of higher degree. However, the elimination process is complex and time-consuming, and the final expression of the matrix is not derived, so the solution requires multiple steps. [3] proposed the pseudo-inverse Jacobian matrix method, which will lead to great joint velocity in singular configuration. Elimination method and Jacobian inverse iterations method were combined in [4] to obtain the full set of inverse kinematic solutions. In [5,6], a damping least squares method (DLS), also known as Levenberg-Marquardt stabilization method, was proposed, and the damping factor was selected based on the minimum singular value. In the singular configuration, the damping term was introduced to balance the error and joint velocity and make the motion more stable, which is an iterative algorithm with better comprehensive performance. The selection of damping factor directly affects algorithm performance. [7] proposed selective damped least squares method (SDLS), which is better than DLS in the selection of damping factors. [8] proposed the Gaussian damping method. Based on the Gaussian distribution of the damping factor, it is the method of determining damping factor with Gaussian function. This method can effectively ensure the continuity of joint velocity and avoid unnecessary errors. However, the average number of iterations of this method is large, and the solving speed is still not as fast as the analytical solution. A new numerical CCD method was proposed in [9], for any differentiable type of joint and demonstrated its use for serial-chain manipulators with coupled joints. The elimination method was optimized with geometric algebra and a new representation of the Euclidean motion group respectively in [10–11]. Four Jacobian-based methods of solving the inverse kinematics task have been improved and evaluated in [12]. Two methods (differential geometric method and variational method) of the extended Jacobian algorithm were examined in [13] to address the approximation of the Jacobian pseudo inverse. An implementation of the inverse kinematics solution for a robot based on Conformal Geometric Algebra was proposed in [14]. A double quaternion based kinematics formulation for the 6R robots was introduced in [15–16] to avoid kinematic singularities. The product of exponentials method based on screw theory was employed for kinematics modeling to avoid the problem of singularity. The inverse kinematics of the 6R robot manipulator was solved by adopting analytical, geometric, and algebraic methods combined with the Paden Kahan subproblem as well as matrix theory in [17–19]. Various optimized numerical and iterative methods failed to achieve the same performance as analytical solutions.

Various soft computing methods [20–26] based on artificial neural network and genetic algorithm are also the research hotspots in recent years. Three different soft computing methods were presented in [20]. Artificial neural network, adaptive neuro fuzzy inference system and genetic algorithm method were compared. Neural network and genetic algorithm were combined in [21]. A neural-network committee machine was designed in [22] to improve the precision of the solution. An intelligent algorithm based on extreme learning machine and sequential mutation genetic algorithm was proposed in [25]. An online adaptive strategy based on the Lyapunov stability theory was presented in [26]. Radial Basis Function (RBF) Neural Networks (NNs) was employed and Quadratic Programming (QP) method was incorporated in the training algorithm of the NNs. Various soft computing methods can achieve accuracy up to the micron level, but the solution speed is still a problem.

In this paper, the enhanced step length coefficient was studied by machine learning methods. Classification and regression methods commonly used in machine learning mainly include DecisionTree(DT) [27] and NaiveBayes(NB). DT is a prediction model to establish a mapping between object attributes and object values. It has fast speed, and is easy to visualize the model, but easy to overfitting. NB takes the maximum conditional probability based on Bayesian criterion. It needs few parameters to estimate and is insensitive to missing data. However, assuming that attributes are independent of each other, it will affect the accuracy. K-NearestNeighbor(KNN) [28] is based on the limit theorem. Decision-making depends on a very small number of the nearest neighbor samples. It is suitable for multi-classification problems. LogisticRegression(LR) is suitable for continuous and categorical independent variables and is widely used in medical related fields. It is sensitive to multiple collinearity of independent variables in the model. RandomForest(RF) [29] integrates the set of decision trees with control variance by using the idea of bagging. Gradient Boosting Decision Tree(GBDT) [30] generates a weak classifier in each iteration through multiple iterations. Each classifier trains on the basis of the residual of the previous one. Support Vector Machine(SVM) estimates regression based on a series of kernels. In this paper, the commonly used RBF nonlinear kernels are used. Bagging method constructs a series of predictive functions and combines them into a predictive function. ExtraTreeRegressor(ETR) [31] is similar to RF and consists of many decision trees. Each decision tree is obtained by using all training samples, and the bifurcation values are obtained completely randomly. XBGboost combines all the predictions of a group of weak learners to train a strong learner through additive training strategies.

Materials and methods

Inverse kinematics of orthogonal non-spherical wrist robot based on analytical method

YASKAWA-EPX2050 (lemma wrist) was taken as an example in this paper. The model is shown in Fig 1, and DH parameters is shown in Table 1.

The transformation matrix between two adjacent coordinate systems of the robot is T_i. Positive kinematics can be obtained from Eq (1).

$$T_1 T_2 T_3 T_4 T_5 T_6 = T_{end} \tag{1}$$

Where $T_{end} = \begin{pmatrix} l_x & m_x & n_x & p_x \\ l_y & m_y & n_y & p_y \\ l_z & m_z & n_z & p_z \\ 0 & 0 & 0 & 0 \end{pmatrix}$

By reversible transformation of Eq (1), $T_3 T_4 T_5 = (T_2)^{-1}(T_1)^{-1} T_{end}(T_6)^{-1}$ was obtained.

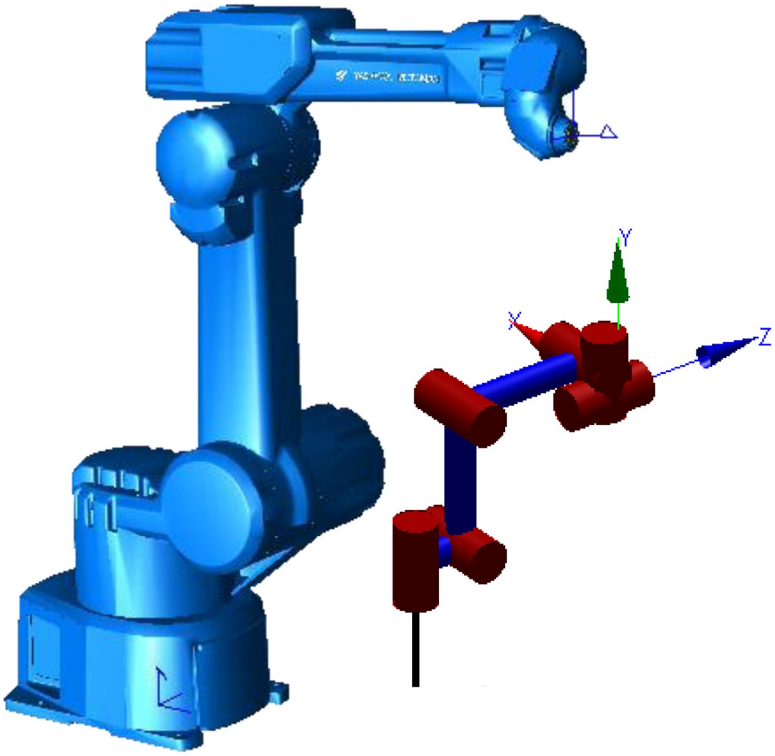

Fig 1. Model of orthogonal non-spherical wrist robot.

https://doi.org/10.1371/journal.pone.0230790.g001

Right and left matrices, elements (3,4), (3,3), (3,2) were correspondingly equal, obtaining the following equations.

$$d_6(c_1n_y - n_xs_1) - c_1p_y + p_xs_1 = d_4$$

$$n_xs_1 - c_1n_y = c_5$$

$$-c_6(c_1m_y - m_xs_1) - s_6(c_1l_y - l_xs_1) = 0$$

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

Table 1. DH parameters of orthogonal non-spherical wrist robot.

	$\boldsymbol{a_{i-1}(m)}$	$\alpha_{i-1}(^\circ)$	$\boldsymbol{d_i(m)}$	$\theta_i(^\circ)$
1(S)	a_1	90	0	θ_1
2(L)	a_2	0	0	θ_2
3(U)	a_3	0	0	θ_3
4(R)	0	90	d_4	θ_4
5(B)	0	−90	d_5	θ_5
6(T)	0	0	d_6	θ_6

https://doi.org/10.1371/journal.pone.0230790.t001

Then we can get Eqs (2)–(4).

$$x_1 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

$$\theta_1 = 2tan^{-1}(x_1) \tag{2}$$

$$\theta_5 = \pm cos^{-1}(n_x s_1 - n_y c_1) \tag{3}$$

$$\theta_6 = tan^{-1}\left(\frac{m_x s_1 - c_1 m_y}{c_1 l_y - l_x s_1}\right) \tag{4}$$

Where $a = p_y - d_6 n_y - d_4$, $b = 2(p_x - d_6 n_x)$, $c = d_6 n_y - p_y - d_4$.

By reversible transformation of Eq (1), $T_2 T_3 T_4 = (T_1)^{-1} T_{end} (T_6)^{-1} (T_5)^{-1}$ was obtained.

Right and left matrices, elements (1,4), (2,4) were correspondingly equal, obtaining the following equations.

$$a_2 c_2 + a_3 c_{23} = c_1 p_x - d_6(c_1 n_x + n_y s_1) - d_5(c_6(c_1 m_x + m_y s_1) + s_6(c_1 l_x + l_y s_1)) - a_1 + p_y s_1$$

$$a_2 s_2 + a_3 s_{23} = p_z - d_6 n_z - d_5(c_6 m_z + l_z s_6)$$

Then we can get Eqs (5)–(6).

$$\theta_3 = \pm cos^{-1}\left(\frac{l^2 + h^2 - {a_2}^2 - {a_3}^2}{2a_2 a_3}\right) \tag{5}$$

$$x_2 = \frac{m \pm \sqrt{m^2 + n^2 - l^2}}{l + n}$$

$$\theta_2 = 2\ tan^{-1}(x_2) \tag{6}$$

Where $l = c_1 p_x - d_6(c_1 n_x + n_y s_1) - d_5\big(c_6(c_1 m_x + m_y s_1) + s_6(c_1 l_x + l_y s_1)\big) - a_1 + p_y s_1$, $h = p_z - d_6 n_z - d_5(c_6 m_z + l_z s_6)$, $m = -a_3 s_3$, $n = a_2 + a_3 c_3$.

By reversible transformation of Eq (1), $T_4 T_5 = (T_3)^{-1}(T_2)^{-1}(T_1)^{-1} T_{end}(T_6)^{-1}$ was obtained.

Right and left matrices, element (1,4) was correspondingly equal, obtaining the following equations.

$$d_5 s_4 = p_z s_{23} - d_6(n_z s_{23} + c_1 n_x c_{23} + n_y s_1 c_{23}) - a_1 c_{23} - a_3 - a_2 c_3 + c_1 p_x c_{23} + p_y s_1 c_{23}$$

Then we can get Eq (7).

$$\theta_4 = \pm sin^{-1}(g) \tag{7}$$

Where

$$g = \frac{p_z s_{23} - d_6(n_z s_{23} + c_1 n_x c_{23} + n_y s_1 c_{23}) - a_1 c_{23} - a_3 - a_2 c_3 + c_1 p_x c_{23} + p_y s_1 c_{23}}{d_5}$$

Inverse kinematics of oblique non-spherical wrist robot based on two optimization methods

YASKAWA-EPX2800 (lemma wrist) was taken as an example in this paper. The model is shown in Fig 2, and DH parameters is shown in Table 2.

Singularity analysis for oblique non-spherical wrist robot.. Differential transform method was used to construct the Jacobian matrix J of the robot in this structure. The robot in a singular configuration had less degree of freedom in operating space, and the determinant of J was 0. Therefore, the determinant could be used to determine the singular configuration. For simplified calculation, set the coordinate system $x_5 - y_5 - z_5$ to coincide with the coordinate system $x_6 - y_6 - z_6$. Via the complicated calculation and derivation, the determinant was obtained as shown in Eq (8).

$$\begin{aligned} det(J) = a_2 s\beta^2[(a_1 a_3 s_3 - a_1 c_3 d_4 + {a_3}^2 c_{23} s_3 - c_3 {d_4}^2 s_{23} + a_3 d_4 s_3 s_{23} - a_2 c_2 c_3 d_4 - a_3 c_3 c_{23} d_4 \\ a_2 a_3 c_2 s_3 - a_3 c_3 c_{23} c\beta d_5 - c_3 c\beta d_4 d_5 s_{23} + a_1 d_5 s_3 s_4 s\beta + a_3 c_2 {c_4}^2 c\beta d_5 - a_1 c_3 c\beta d_5 {s_4}^2 - \\ a_2 c_2 c_3 c\beta d_5 {s_4}^2 + a_2 c_2 d_5 s_3 s_4 s\beta + a_3 c_{23} d_5 s_3 s_4 s\beta + d_4 d_5 s_3 s_4 s_{23} s\beta) s_5 + (a_1 c_3 + a_3 c_2 + \\ a_2 c_2 c_3) c_4 s_4 d_5 c_5] \end{aligned} \tag{8}$$

According to Eq (8), there were three cases when $det(J) = 0$.

$$\begin{cases} s_5 = 0, s_4 = 0 \\ s_5 = 0, c_4 = 0 \\ s_5 = 0, c_2 = 0, c_3 = 0 \end{cases} \tag{9}$$

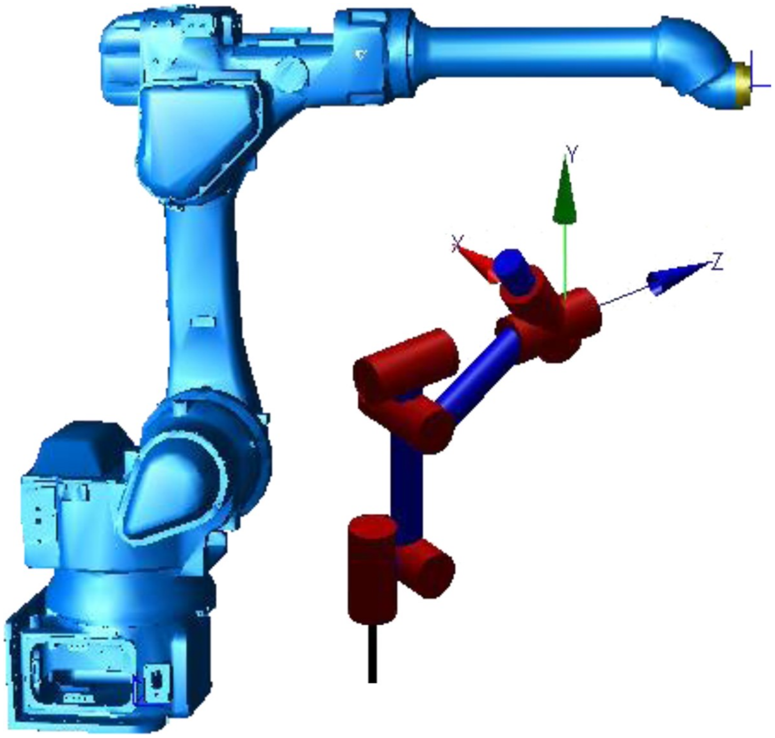

Fig 2. Model of oblique non-spherical wrist robot.

https://doi.org/10.1371/journal.pone.0230790.g002

Table 2. DH parameters of oblique non-spherical wrist robot.

	$\boldsymbol{a}_{i-1}(\boldsymbol{m})$	$\alpha_{i-1}(^\circ)$	$\boldsymbol{d}_i(\boldsymbol{m})$	$\theta_i(^\circ)$
1(S)	a_1	90	0	θ_1
2(L)	a_2	0	0	θ_2
3(U)	a_3	90	0	θ_3
4(R)	0	β	d_4	θ_4
5(B)	0	$-\beta$	d_5	θ_5
6(T)	0	0	d_6	θ_6

https://doi.org/10.1371/journal.pone.0230790.t002

By decomposing the Jacobian matrix J, $J = USV^T$ was obtained, of which S was a diagonal matrix composed of non-zero singular values of J. The diagonal elements were arranged in descending order as $\sigma_1 \geq \sigma_2 \geq \sigma_3 \geq \sigma_4 \geq \sigma_5 \geq \sigma_6$, of which the σ_6 referred to the minimum singular value.

We then analyzed the relations between the singularities of the configuration and the singular value, and the characteristics of the singular values. One thousand positions were selected on a random basis to analyze the singular values, as shown in Fig 3. 10^6 positions were selected at random. When the singular configuration and non-singular configuration, the mean singular value and variable coefficient were in statistics and shown in Table 3.

Fig 3 intuitively showed the bigger change in σ_2, σ_3, σ_6, and smaller change in σ_1, σ_4, σ_5, of which the σ_6 had the greatest change. Therefore, in various damping iteration methods for analyzing the inverse kinematics, the method taking all singular values into account had a better effect than the method of determining damping factors only by the minimum singular value. The statistics data in Table 3 showed that the mean value of minimum singular values varied greatly from variable coefficients for singular configuration and non-singular configuration. Therefore, in analyzing the influence factors of the enhanced step length coefficient, the singularity of the configuration could be roughly represented by the minimum singular values.

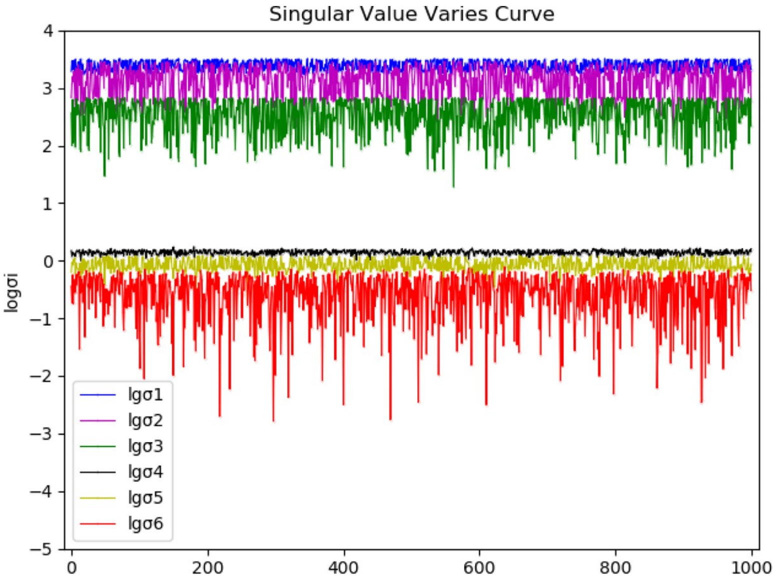

Fig 3. The characteristics of the singular values.

https://doi.org/10.1371/journal.pone.0230790.g003

Table 3. Singular value comparison of singular points and non-singular points.

	Non-singular		Singular	
	mean	variation coefficient	mean	variation coefficient
σ_1	2553.239	0.180	2524.623	0.183
σ_2	1495.735	0.517	1464.438	0.517
σ_3	399.138	0.505	386.052	0.538
σ_4	1.394	0.086	1.389	0.091
σ_5	0.861	0.224	0.975	0.239
σ_6	0.347	0.566	1.639e^{-14}	2.217

https://doi.org/10.1371/journal.pone.0230790.t003

Optimized elimination method.. The specific process of elimination method was as follows. By reversible transformation of Eq (1), it was obtained that:

$$T_3 T_4 T_5 = (T_2)^{-1}(T_1)^{-1} T_{end}(T_6)^{-1}$$

Right and left matrices, elements in column 3, 4 were correspondingly equal, obtaining Eq 6 unrelated to θ_6.

$$p_1 : c_3 f_1 + s_3 f_2 + a_2 = c_2 h_1 + s_2 h_2 \ (e_1)$$

$$p_2 : c_3 f_2 - s_3 f_1 = s_2 h_1 - c_2 h_2 \ (e_2)$$

$$p_3 : f_3 = h_3 (e_3)$$

$$l_1 : c_3 r_1 + s_3 r_2 = c_2 n_1 + s_2 n_2 \ (e_4)$$

$$l_2 : s_3 r_1 - c_3 r_2 = c_2 n_2 - s_2 n_1 \ (e_5)$$

$$l_3 : r_3 = n_3 (e_6)$$

Via the vector calculation of p and l, $p \cdot p$, $p \cdot l$, $p \times l$, $(p \cdot p)l - 2(p \cdot l)p$, Eq 8 were derived.

$$p_1^{\ 2} + p_2^{\ 2} + p_3^{\ 2} \ (e_7) p_1 l_1 + p_2 l_2 + p_3 l_3 \ (e_8)$$

$$p_1 l_2 - p_2 l_1 \ (e_9) p_2 l_3 - p_3 l_2 \ (e_{10}) p_3 l_1 - p_1 l_3 \ (e_{11})$$

$$(p_1^{\ 2} + p_2^{\ 2} + p_3^{\ 2}) l_1 - 2(p_1 l_1 + p_2 l_2 + p_3 l_3) p_1 \ (e_{12})$$

$$(p_1^{\ 2} + p_2^{\ 2} + p_3^{\ 2}) l_2 - 2(p_1 l_1 + p_2 l_2 + p_3 l_3) p_2 \ (e_{13})$$

$$(p_1^{\ 2} + p_2^{\ 2} + p_3^{\ 2}) l_3 - 2(p_1 l_1 + p_2 l_2 + p_3 l_3) p_3 \ (e_{14})$$

Via eliminating the Eq 14 $e_1 \sim e_{14}$ by θ_1, θ_2, we obtained Eq 6. Wrote Eq 6 in the form of a matrix equation, obtaining the matrix equation

$$(\Sigma) v_1 = 0 \tag{10}$$

Of which $v_1 = [s_4 s_5 s_4 c_5 c_4 s_5 c_4 c_5 s_4 c_4 s_5 c_5 \ 1]^T$, (Σ) referred to a 6*9 matrix containing only unknowns s_3, c_3. Converted s_3, c_3, s_4, c_4, s_5, c_5 into $s_3 = \frac{2x_3}{1+x_3^{\ 2}}$, $c_3 = \frac{1-x_3^{\ 2}}{1+x_3^{\ 2}}$, $s_4 = \frac{2x_4}{1+x_4^{\ 2}}$, $c_4 = \frac{1-x_4^{\ 2}}{1+x_4^{\ 2}}$,

$s_5 = \frac{2x_5}{1+{x_5}^2}$, $c_5 = \frac{1-{x_5}^2}{1+{x_5}^2}$, and obtained the matrix equation

$$(\Sigma')v_2 = 0 \tag{11}$$

Of which $v_2 = [{x_4}^2{x_5}^2{x_4}^2x_5{x_4}^2x_4{x_5}^2x_4x_5x_4{x_5}^2x_5 1]^T$, (Σ') referred to a 6*9 matrix containing only unknowns x_3. Multiplied each line of the matrix equation by x_4 and merged the equation obtained with the original equation, to get the matrix equation

$$(\Sigma^{''})v = 0 \tag{12}$$

Of which $v = [{x_4}^3{x_5}^2{x_4}^3x_5{x_4}^3{x_4}^2{x_5}^2{x_4}^2x_5{x_4}^2x_4{x_5}^2x_4x_5x_4{x_5}^2x_5\ 1]^T$, (Σ'') referred to a 12*12 matrix with only unknowns x_3. This was an over-constrained linear system. The coefficient matrix (Σ'') should be singular for the equation to have a solution, and the determinant of the coefficient matrix was a 16th degree polynomial in one variable regarding x_3. The roots of the polynomial were corresponded to the solution to inverse kinematics. For improving the solving speed, the problem of solving 16th degree polynomial was simplified to find the matrix eigenvalues and eigenvectors. (Σ'') can be written as $A{x_3}^2 + Bx_3 + C$, and the Eq (12) can be written as

$$(A{x_3}^2 + Bx_3 + C)v = 0 \tag{13}$$

A, B, C were a known 12*12 matrix, the Eq (13) can be written as

$$\begin{bmatrix} 0 & I \\ -A^{-1}C & -A^{-1}B \end{bmatrix}\begin{bmatrix} v \\ x_3v \end{bmatrix} = x_3\begin{bmatrix} v \\ x_3v \end{bmatrix} \text{ i.e. } M\begin{bmatrix} v \\ x_3v \end{bmatrix} = x_3\begin{bmatrix} v \\ x_3v \end{bmatrix} \tag{14}$$

The matrix eigenvalues of M were x_3, and the eigenvectors were $V = \begin{bmatrix} v \\ x_3v \end{bmatrix} / \left\| \begin{bmatrix} v \\ x_3v \end{bmatrix} \right\|$. θ_3 can be obtained from $\theta_3 = 2tan^{-1}(x_3)$.

In this paper, the elimination method is optimized and supplemented in three points.

1. The paper presented the specific process of Eq 6 obtained by eliminating Eq 14 by $e_1 \sim e_{14}$. First, the expression s_1, c_1 obtained from two equations e_3, e_6 was substituted into e_7, e_8, e_9, e_{14}, getting 4 equations. Two equations would be obtained by simplifying the e_1, e_2, e_4, e_5, e_{10}, e_{11}, e_{12}, e_{13} via Eqs (15)–(16). There were no specific simplified formulas in other literatures.

$$\frac{{\mu_1}^2}{2a_1}(e_{12} - \delta_1e_4 + \delta_2e_1) - \lambda_1\mu_1e_{10} + \mu_1we_2 - \mu_1(r - d_1)e_5 = 0 \tag{15}$$

$$\frac{{\mu_1}^2}{2a_1}(e_{13} - \delta_1e_5 + \delta_2e_2) - \lambda_1\mu_1e_{11} - \mu_1we_1 + \mu_1(r - d_1)e_4 = 0 \tag{16}$$

Where $\lambda_i = cos\alpha_i$, $\mu_i = sin\alpha_i$, $\delta_1 = p^2 + q^2 + (r - d_1)^2 - {a_1}^2$, $\delta_2 = 2(pu + qv + (r - d_1)w)$

2. In other documents, the final expression of the matrix A, B, C was not deduced, and the solution required to be made in several steps. For improving the solving speed, the paper adopted complex coefficient extraction and simplification to derive the final expression of matrix A, B, C. Maximized the simplification of the whole solution process for achieving one-step solution of the matrix M. This was also the greatest contribution of the paper to

eliminate optimization. The expression of A, B, C is shown in the appendix and refer to the uploaded matlab code. Refer to S1 Table. https://github.com/Wangxiaoqi1031/robot

3. In this paper, the solution formula in the simplest form of other joint angles θ_1, θ_2, θ_4, θ_5, θ_6 was introduced to integrate the whole method. Through a lot of experiments, taking the item with a large absolute value in the M matrix eigenvector as ratio, to get more accurate x_4, x_5. Via analysis, the maximum value may be v_1, v_3, v_{10}, v_{12}, and the formula θ_4, θ_5 obtained was shown as Eq (17).

$$\begin{cases} \theta_4 = tan^{-1}\left(\frac{v_1}{v_4}\right) & \theta_5 = tan^{-1}\left(\frac{v_1}{v_2}\right) & max(v_i) = |v_1| \\ \theta_4 = tan^{-1}\left(\frac{v_3}{v_6}\right) & \theta_5 = tan^{-1}\left(\frac{v_2}{v_3}\right) & max(v_i) = |v_3| \\ \theta_4 = tan^{-1}\left(\frac{v_7}{v_{10}}\right) & \theta_5 = tan^{-1}\left(\frac{v_{10}}{v_{11}}\right) & max(v_i) = |v_{10}| \\ \theta_4 = tan^{-1}\left(\frac{v_9}{v_{12}}\right) & \theta_5 = tan^{-1}\left(\frac{v_{11}}{v_{12}}\right) & max(v_i) = |v_{12}| \end{cases} \tag{17}$$

$$s_1 = \frac{(p_y - d_6 n_y)(c_4 c\beta s\beta + s_4 s_5 s\beta - c_4 c_5 c\beta s\beta) - c_4 d_5 n_y s\beta}{n_y p_x - n_x p_y}$$

$$s_2 = \frac{h_1 e_2 + h_2 e_1}{{h_1}^2 + {h_2}^2}$$

$$\theta_1 = sin^{-1}(s_1),\ \theta_2 = sin^{-1}(s_2) \tag{18}$$

$$x_6 = \frac{m \pm \sqrt{m^2 + n^2 - l^2}}{l + n}$$

$$\theta_6 = 2tan^{-1}(x_6) \tag{19}$$

Where $l = \frac{d_5 c\beta(n_y c_1 - n_x s_1) + d_6(n_y c_1 - n_x s_1) - p_y c_1 + p_x s_1}{d_5 s\beta}$, $n = m_y c_1 - m_x s_1$, $m = l_y c_1 - l_x s_1$

Optimized Gaussian damped least square method.. The Jacobian matrix pseudo-inverse method proposed by Whitney [3] is most widely used for solving robot inverse kinematics. The joint velocity of the robot is obtained from the speed of the end effector.

$$\dot{\theta} = J^+ \dot{X} \tag{20}$$

Where J^+ is the pseudo inverse of the robot Jacobian matrix J. This method gives the best possible solutions that minimize joint velocity norm $\|\dot{\theta}\|^2$ and the end-effector tracking error $\|\dot{X} - J\dot{\theta}\|^{.2}$. However, the method is unstable near singularities. SVD decomposing the Jacobian matrix to get

$$J = USV^T \tag{21}$$

S is a diagonal matrix formed by the singular values of J.which are arranged in descending order, $\sigma_1 \geq \sigma_2 \geq \cdots \sigma_6$. Hence, the joint speed could be computed as follows.

$$\dot{\theta} = J^{+}\dot{X} = \Sigma_{i=1}^{6}\frac{1}{\sigma_i}VU^{T}\dot{X} \tag{22}$$

While a robot approaching a singular configuration, the smallest singular value reaches 0 and it causes infinite joint velocities as implied in (22). Therefore, damped least squares (DLS) introduces a damping factor, balances the precision of the solution and the increase in joint velocities near singularities by minimizing $\|J\dot{\theta} - \dot{X}\|^{2} + \lambda^{2}\|\dot{\theta}^{*}\|^{2}$. The joint speed could be computed as follows.

$$\dot{\theta}^{*} = \Sigma_{i=1}^{6}\frac{\sigma_i}{\sigma_i^{2} + \lambda^{2}}VU^{T}\dot{X} \tag{23}$$

Multiple methods are generated from damping least square method and selecting strategies based on different damping factors. A constant damping factor was proposed in DLS method. The relative relation between the end-effecter position and the target position was considered in SDLS method, and a piecewise function was adopted to determine the damping factor. Gauss damping least square (GDLS) method refers to determine the damping factors with the following functions, based on the Gaussian distribution of the damping factor.

$$\lambda_i = \lambda_{max} \cdot e^{-(\sigma_i/\varepsilon)^{2}} \tag{24}$$

Where λ_{max} is the maximum value of the damping factor and ε is a scalar quantity indicating region of singularities and σ_i is singular value. GDLS is more effective than other methods that select constant and piecewise function damping factors. On one hand, each singular value corresponds to a damping factor, so that damping only act on singular vectors, to prevent the introduction of unnecessary damping and reduce the errors. On the other hand, when the robot is in close proximity to singular configuration, the method allows to make the joint velocity continuously change from undamped to damped, for more stable motion. The λ_{max}, ε of the robot configuration in this paper was determined based on the genetic algorithm in the literature [8] and obtained $\lambda_{max} = 0.071$, $\varepsilon = 0.051$. The Jacobian matrix pseudo-inverse method J^{T}, DLS,SDLS and GDLS were used to analyze the inverse kinematics when close to the singular configuration and compare the performance of the four methods, as shown in Fig 4.

Fig 4 showed that when the robot approaches a singular configuration, compared with the Jacobian matrix pseudo-inverse method, the damped least square method prevented oscillation and effectively controlled the joint velocity, making the more stable motion. SDLS had similar effects as GDLS had, which reduced unnecessary damping and errors, compared with DLS.

DLS series methods including GDLS can be used to solve the IK problem of robot, and can ensure the stability of motion by controlling the joint speed. However, hundreds of iterations are needed to obtain the solution meeting the accuracy requirement in the solving process, which cannot achieve the goal of high-speed to solve IK. In the iteration process, it is found that the speed of approaching the iteration convergence can be improved through multiplying the iterative increment $d\theta$ by a coefficient k, achieving faster iteration convergence. The change of the number of iterations with k can be usually obtained as shown in Fig 5 in the process of solving with GDLS method. Therefore, we introduce the concepts of enhanced step length coefficient and optimal enhanced step length coefficient.

Definition: During the iteration, change $\theta_{cur} = \theta_{cur} + d\theta$ into $\theta_{cur} = \theta_{cur} + kd\theta$, to make the iteration converge faster, and k is called the enhanced step length coefficient. The number of iterations reached the least, when k was assigned a value, and the value was defined as the optimal enhanced step length coefficient *k_optimal.*

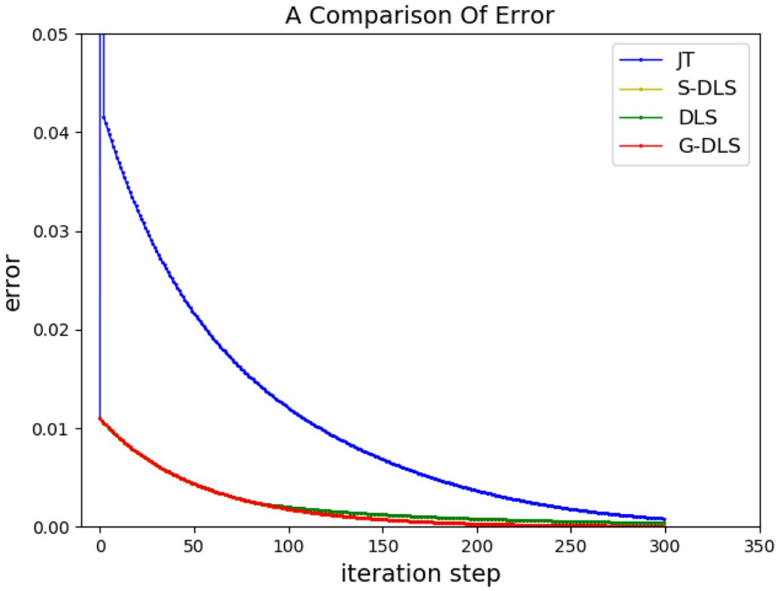

Fig 4. Comparison of four methods.

https://doi.org/10.1371/journal.pone.0230790.g004

To study the distribution of *k_optimal* and the range of the minimum number of iterations. We selected randomly 10^5 points and controlled the initial and target position within a certain

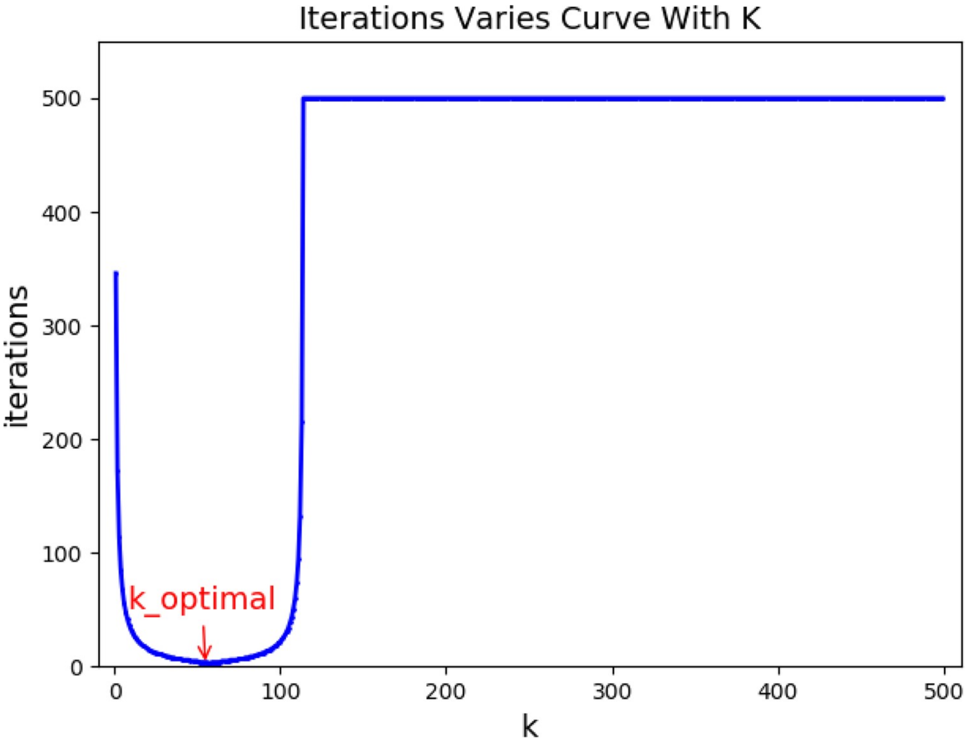

Fig 5. Number of iterations varies with k.

https://doi.org/10.1371/journal.pone.0230790.g005

range. The probability distribution of *k_optimal* and the minimum number of iterations was obtained and shown in Fig 6. Distribution statistics data of *k_optimal* were shown in Table 4.

Fig 6 and Table 4 showed that the *k_optimal* was mainly distributed between [30,90] and centrally distributed between [55,60). If proper *k* was selected, the number of iterations can be controlled within 20 times.

To study the influencing factors of *k_optimal*, the iterative process was analyzed and the supposed possible factors to be: The difference value *dif* between current position T_{cur} and terminal position T_{end}, iterative convergence error E_{rr}, initial iteration value θ_i and singularity. Via the analysis on the singularity in section 3.1, the minimum singular value σ_m was adopted to represent the singularity. When other conditions remained unchanged, we got and the variations of *k_optimal* and the minimum number of iterations with *dif*, E_{rr}, θ_i, σ_m were shown in Fig 7.

From Fig 7, when $E_{rr} \leq 0.01$, there was almost no influences of E_{rr} on *k_optimal*. When *dif* changed over a larger scale, it did affect *k_optimal*. θ_3 and θ_5 had greater influence on *k_optimal* when the robot was far away from the singular configuration. When the robot approached the singular configuration, θ_2, θ_3, θ_4 and θ_5 had greater influence on *k_optimal*. After repeated experiments, we get that not every group of data could produce the same change rule and curve, while this section was intended to determine the influence factors of *k_optimal*, so the curve shape can be neglected. The singularity had no direct effect on *k_optimal*. Therefore, conclusion comes that the initial iteration value θ_i has a great impact on *k_optimal*.

Based on the analysis on the influencing factors of *k_optimal*. The initial iteration value θ_i was used as input and *k_optimal* as the output, and the *k_optimal* of each iteration was determined by the classification and regression models of machine learning. We selected 10^5 random positions as research data and changed the initial iteration value θ_i to get *k_optimal* when other conditions remained unchanged.

- Classification method

According to the distribution of *k_optimal* in Table VI, we divided the values of k_optimal into four classes [30,55), [55,60], (60,65), [65,90]. It can be seen from Fig 5 that the number of

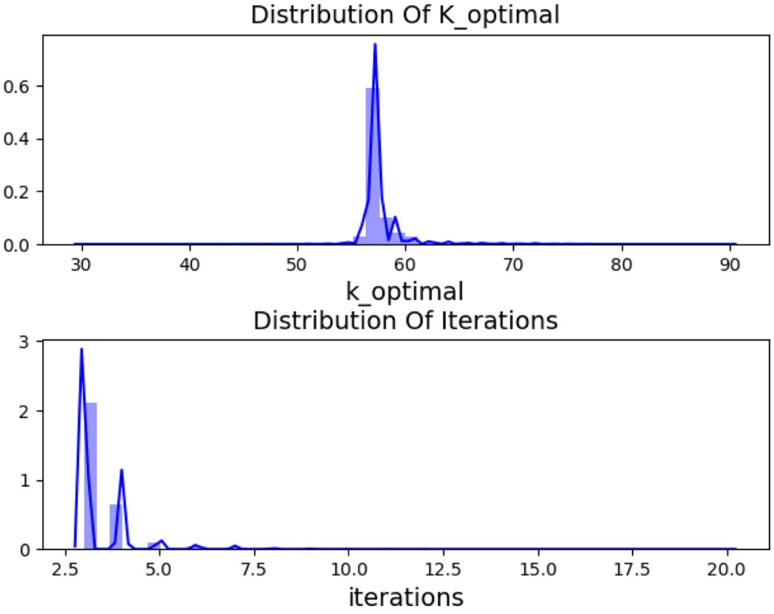

Fig 6. Distribution of *k_optimal* and iterations number.

https://doi.org/10.1371/journal.pone.0230790.g006

Table 4. Probability of *k_optimal* in different intervals.

k_optimal	Probability(%)
[30,55)	0.602
[55,60]	94.614
(60,65)	2.782
[65,90]	2.002

https://doi.org/10.1371/journal.pone.0230790.t004

iterations changes with *k* continuously and gradually decreases when *k* approaches *k_optimal*. Therefore, a slight difference between the final selected *k* value and *k_optimal* is acceptable, and the number of iterations will be close to the minimum. The initial iteration value θ_i was used as input and the classification of *k_optimal* used as the output, and the classification models of machine learning were used for classifying. The performance was evaluated by OA (overall accuracy), ECA (each class accuracy), AA (average accuracy) and TT (training time). Seven models with better performance were selected from various models for comparison, as in Table 5.

Table 5 showed that the overall and the average accuracy of the DT model were the highest. The ECA obtained from DT and GBDT were relatively high. NB model featured the shortest training time. On the whole, DT model had the best classification result. After classification, in the iteration process, the median of each range is taken for the value of *k_optimal* of each class, which is *k_optimal* = 42.5,57.5,62.5,77.5.

- Regression method

The initial iteration value θ_i was used as input and the *k_optimal* was used as the output, and the regression models of machine learning were used for analysis. The paper used three commonly used statistical indicators including root mean square error (RMSE), coefficient of

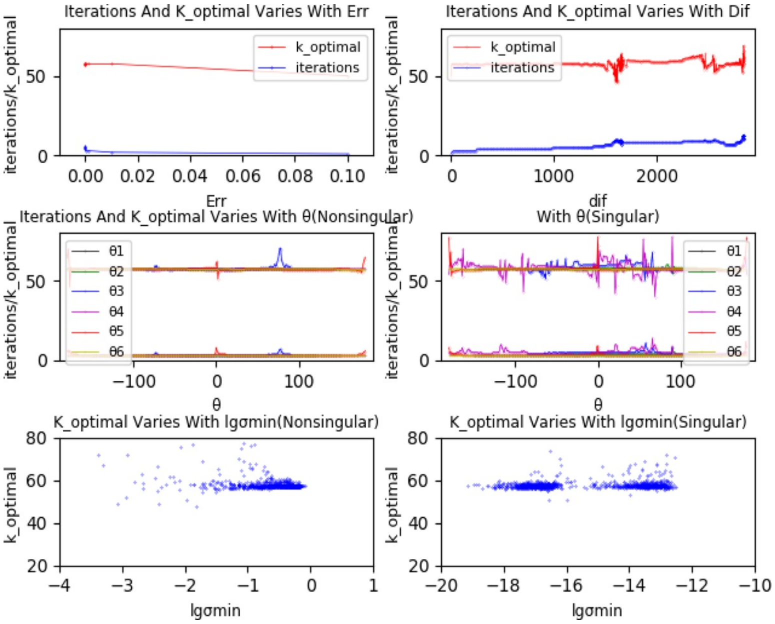

Fig 7. Influencing factors of *k_optimal*.

https://doi.org/10.1371/journal.pone.0230790.g007

Table 5. Performance comparison of classification models.

Model	OA	ECA	AA	TT
KNN	0.89	[0.73 0.91 0.38 0.70]	0.68	0.115
NB	0.88	[0.69 0.89 0.00 0.73]	0.58	**0.016**
LR	0.87	[0.82 0.87 0.00 0.75]	0.61	0.182
RF	0.97	[0.95 0.97 0.83 0.93]	0.92	0.069
DT	**0.99**	[0.98 **0.99 0.95 0.97**]	**0.97**	0.021
GBDT	0.97	[**1.00** 0.98 0.88 0.94]	0.95	4.792
SVM	0.85	[0.00 0.85 0.00 0.00]	0.21	69.87

https://doi.org/10.1371/journal.pone.0230790.t005

determination (R^2) and mean absolute error (MAE) to evaluate the performance of the model.

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}(Y_{i,m} - Y_{i,e})^2} \tag{25}$$

$$R^2 = \frac{\Sigma_{i=1}^{n}(Y_{i,m} - Y_{i,e})^2}{\Sigma_{i=1}^{n}(Y_{i,m} - \bar{Y}_{i,m})^2} \tag{26}$$

$$MAE = \frac{1}{n}\Sigma_{i=1}^{n}|Y_{i,m} - Y_{i,e}| \tag{27}$$

Three indicators were calculated according to Eqs (25)–(27). Five models with better performance were selected from various models for comparison, as shown in Table 6, of which four with better results were selected and 1000 groups of data were taken at random to generate the regression fitting error curve, as shown in Fig 8.

Table 6 showed that the DT model featured the best results for both training and testing samples. It could also be seen intuitively from Fig 8 that DT model had the best fitting effect. The DT model adopted cost complexity pruning (CCP) method and adjusted relevant parameters to avoid overfitting. The flow chart of the optimized Gauss damped least squares method are as follows.

The flow chart of the optimized algorithm is shown in Fig 9.

Results and discussion

Examples of orthogonal non-spherical wrist robot

The D-H parameters are $a_1 = 0.3$, $a_2 = 0.85$, $a_3 = 0.844$, $d_4 = 0.165$, $d_5 = 0.141$, $d_6 = 0.093$. We set $\theta_1 = 30^\circ$, $\theta_2 = 70^\circ$, $\theta_3 = 60^\circ$, $\theta_4 = 140^\circ$, $\theta_5 = -40^\circ$, $\theta_6 = 40^\circ$. Then we can get

Table 6. Performance comparison of regression models.

Model	Training			Testing		
	***R*²**	***RMSE***	***MAE***	***R*²**	***RMSE***	***MAE***
DT	**1.000**	**0.000**	**0.000**	**0.982**	**0.340**	**0.064**
RF	0.988	0.253	0.072	0.945	0.599	0.198
Bagging	0.988	0.256	0.071	0.949	0.576	0.193
ETR	0.999	0.038	**0.000**	0.933	0.656	0.101
XBGboost	0.702	1.282	0.456	0.561	1.690	0.572

https://doi.org/10.1371/journal.pone.0230790.t006

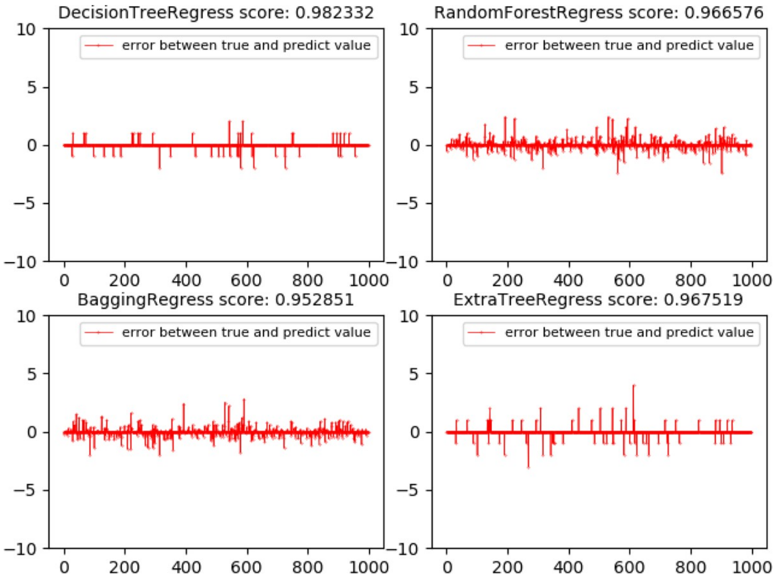

Fig 8. Regression fitting error curve of four models.

https://doi.org/10.1371/journal.pone.0230790.g008

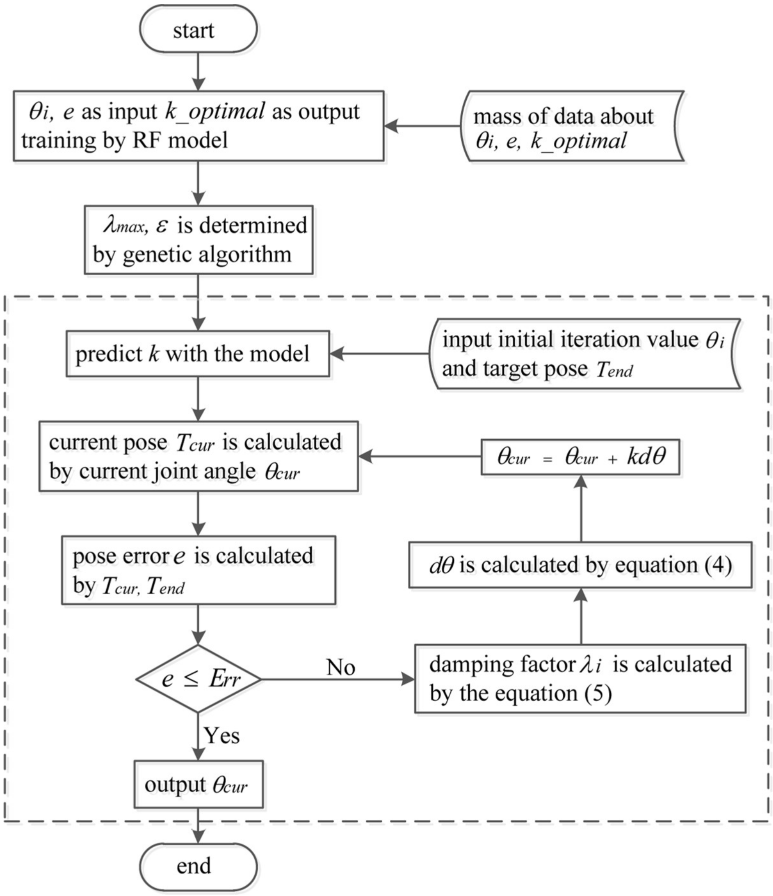
start
θi, e as input $k\_optimal$ as output training by RF model
mass of data about θi, e, $k\_optimal$
λ_{max}, ε is determined by genetic algorithm
predict k with the model
input initial iteration value θ_i and target pose T_{end}
current pose T_{cur} is calculated by current joint angle θ_{cur}
$\theta_{cur} = \theta_{cur} + kd\theta$
pose error e is calculated by T_{cur}, T_{end}
$d\theta$ is calculated by equation (4)
$e \leq Err$
No
damping factor λi is calculated by the equation (5)
Yes
output θ_{cur}
end

Fig 9. The flowchart of the optimized method.

https://doi.org/10.1371/journal.pone.0230790.g009

Table 7. Results that meet the accuracy requirements.

	θ_1(°)	θ_2(°)	θ_3(°)	θ_4(°)	θ_5(°)	θ_6(°)	err
1	-41.778451	64.145049	72.666641	1.7309941	76.140837	-61.947867	1.4993e^{-12}
2	-41.778451	138.49267	-72.666641	72.716655	76.140837	-61.947867	1.8000e^{-12}
3	**30.000000**	**69.999999**	**60.000000**	**139.99999**	**-40.000000**	**39.999999**	**3.8979e^{-13}**
4	30.000000	131.31962	-60.000000	198.68037	-40.000000	39.999999	5.9746e^{-13}

https://doi.org/10.1371/journal.pone.0230790.t007

$$T_{end} = [T_1, T_2, T_3, T_4]^T.\text{Where } T_1 = [-0.80287234, -0.45682599, 0.38302222, 14.723871];$$

$$T_2 = [0.10504046, -0.74084306, -0.66341395, -264.28809];$$

$$T_3 = [-0.586824089, 0.492403877, -0.642787610, 1416.60239]; T_4 = [0, 0, 0, 1]$$

According to the Eqs (2)–(7), we can get $\theta_1 \sim \theta_6$. All the results that meet the accuracy requirements are as shown in Table 7.

For similar configurations, the formulas are universal.

Examples of oblique non-spherical wrist robot

Examples and properties of optimized elimination method.. The D-H parameters of the configuration were $a_1 = 0.25$, $a_2 = 0.95$, $a_3 = 0.3$, $d_4 = 1.55$, $d_5 = 0.114$, $d_6 = 0.123$, $\beta = 60^\circ$, we set $\theta_1 = 14^\circ$, $\theta_2 = 29.7^\circ$, $\theta_3 = -45^\circ$, $\theta_4 = 71^\circ$, $\theta_5 = -63^\circ$, $\theta_6 = 100^\circ$, and let the terminal position matrix be $T_{end} = [T_1, T_2, T_3, T_4]^T$.

Where $T_1 = [-0.53060777 - 0.71765135, 0.4510343, 1.047652];$

$$T_2 = [-0.79429474, 0.23523209, -0.56014439, 0.21160551];$$

$$T_3 = [0.29589063, -0.65547114, -0.69484266, -1.2686177]; T_4 = [0, 0, 0, 1]$$

The matrix A, B,C can be directly obtained from the formula in the appendix. The matrix M was obtained from the formula (14) and the eigenvalues and eigenvectors of the matrix were obtained. Real eigenvalue of M matrix was [-61.880055, 39.835519, -6.2262997, -5.0896338, -0.41421356, -0.34282825, -0.21892882, -0.17065825]. Then other joint variables were obtained from the equation (17)–(19). All solutions that meet the accuracy requirements, as shown in Table 8.

For analyzing the algorithm performance, we selected 10^6 positions of the robot in the reachable workspace at random, to find a solution by the above optimized elimination method. And the testing environment was (Intel(R) Xeon CPU2.8GHz, python). The statistical results

Table 8. Results that meet the accuracy requirements.

	θ_1	θ_2	θ_3	θ_4	θ_5	θ_6	err
1	-162.89741	156.18595	-178.14833	-76.120407	-48.798102	47.154357	3.03917e^{-14}
2	-163.75277	154.00112	177.12399	85.181957	49.016271	-168.81112	2.64275e^{-14}
3	20.011962	-142.53364	-161.7514	140.31013	-63.624205	8.6364123	4.26276e^{-14}
4	14.018839	-140.47799	-157.76849	-71.20786	62.629725	144.74019	2.01718e^{-14}
5	**14.000000**	**29.7000000**	**-45.000000**	**71.000000**	**-63.00000**	**100.000000**	**2.51878e^{-11}**
6	19.565127	27.505394	-37.846326	-133.49287	58.583668	-117.21089	1.86615e^{-11}
7	-166.21313	-68.60891	-24.697729	111.37032	88.933046	109.22533	5.38294e^{-11}
8	-158.97354	-69.890768	-19.369394	-17.484555	-91.606294	-9.1357937	7.86953e^{-11}

https://doi.org/10.1371/journal.pone.0230790.t008

Table 9. Performance analysis of two methods of machine learning.

method	Average number of iterations	Average forecast time(s)	Iteration more than 10	Iteration more than 20
classification	6.55	7.962e-07	1.68%	0.08%
regression	3.6414	9.8465e-07	0.29%	0.05%

https://doi.org/10.1371/journal.pone.0230790.t009

showed that the average solving speed was reduced from 11-13ms before optimization to 1.48ms, and the solving speed was reduced to 20% of the original, the average accuracy of solution reached 2.3205e^{-13}. During solving by elimination, the proportion of the points that cannot be solved accurately since the *A* matrix was singular or the robot approached the singular configuration was 0.086%, which can be solved by the following solutions.

1. In the case of A is singular, finding inverse is changed to find pseudo-inverse.
2. According to the singularity analysis in Section 3.1, there is no singular regions in trajectory planning.
3. Points that cannot be solved accurately can be processed by iteration method.

Examples and properties of optimized Gaussian damped least square method. For comparing the solving performance of the classification and regression, we selected 10^5 positions at random and adopted DT model to predict the *k* value separately, and solved it in accordance with the above steps. The resulting statistics, as shown in Table 9.

Table 9 gave that the regression method featured less average number of iterations. The two methods had almost the same average prediction time, and the average prediction time was negligible compared with the solution time. The regression method had a lower proportion of iterations over 10 times and over 20 times, less than 0.1%. On the whole, the regression method had better effects than classification.

For comparing the solving performance of the methods before and after optimization, the Gauss damped least square method and the optimized method in the paper were used for solving separately. A complete experimental process was: we randomly selected 10^5 data points as the original data set, each data point contains the initial iteration value and *k_optimal*. The collected data set was used to train a DecisionTree model through supervised learning to obtain a model that can be used to predict *k_optimal*. Then 10^4 data points were randomly selected in the robot workspace, that is, the initial iteration values were randomly selected, and the IK was solved by the GDLS method before and after optimization. Then we counted the number of iterations and the solution time. The above experiment was repeated 10 times and the statistics were obtained. The best, worst, and average cases in the statistical repeat experiments were shown in Table 10.

Table 10 showed that the average iteration times of the optimized method was obviously reduced, and the average solution time was reduced to 5% of the original, the average solution time of optimized method was close to the solution time of the analytic solution of the general configuration. The optimized method in this paper was very effective for improving the solving speed.

Table 10. Performance analysis before and after optimization.

method	Number of iterations			Solution time (ms)		
	Maximum	Minimum	Average	Maximum	Minimum	Average
GDLS	137.1	119.8	125.3	3.813	3.309	3.491
Optimized GDLS	4.143	2.918	3.641	0.1467	0.1132	0.1298

https://doi.org/10.1371/journal.pone.0230790.t010

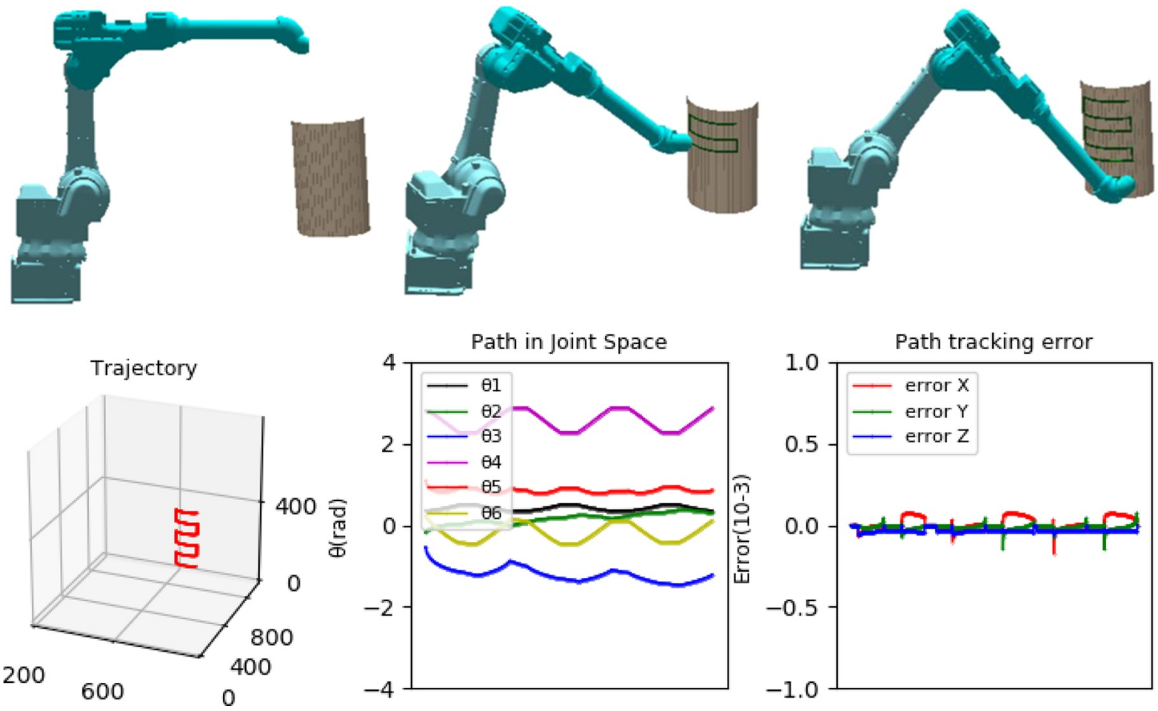

Fig 10. Simulation verification of robot.

https://doi.org/10.1371/journal.pone.0230790.g010

A simulation platform was developed, based on Qt integrated opengl, QML and C ++ language. It was superior in cross-platform, high flexibility and high reusability, etc. Combined with the interpolation algorithm, a trajectory was planned. The algorithms in the paper were adopted, the robot model moved the planned trajectory on a curved surface in the simulation platform, and the collecting data were analyzed, as shown in Fig 10.

According to Fig 10, the planned trajectory of a robot model moving on a curved surface was accurate and the robot moved smoothly. The joint angle changed smoothly in the joint space, and the trajectory tracking errors were up to grade. The optimized method could ensure the stability and accuracy of the motion.

There was the experimental platform of oblique non-spherical wrist robot in the lab as shown in Fig 11. Same as the configuration of robot studied in the paper, the structure parameters were $a_1 = 0.135$, $a_2 = 0.7$, $a_3 = 0.1$, $d_4 = 0.433$, $d_5 = 0.115$, $d_6 = 0.174$, $\beta = 60^\circ$. The two methods optimized in this paper were encoded and implemented on the DSP end of the controller, to control the motion of the robot at high speed and smoothly. The effectiveness of the methods were verified.

Conclusions

For the orthogonal non-spherical wrist configuration, the simplest analytical solution was given in this paper. For the oblique non-spherical wrist configuration, elimination method and Gauss damped least squares method were selected to optimize the solution. For the elimination method, this paper mainly made three points optimization, gave the expression of equality simplification, deduced the expressions of final A, B and C matrix to improve the speed of solution, and gave the simplest expression of all joint variables to complement the integrity of elimination method. Elimination method was mainly time-consuming in solving

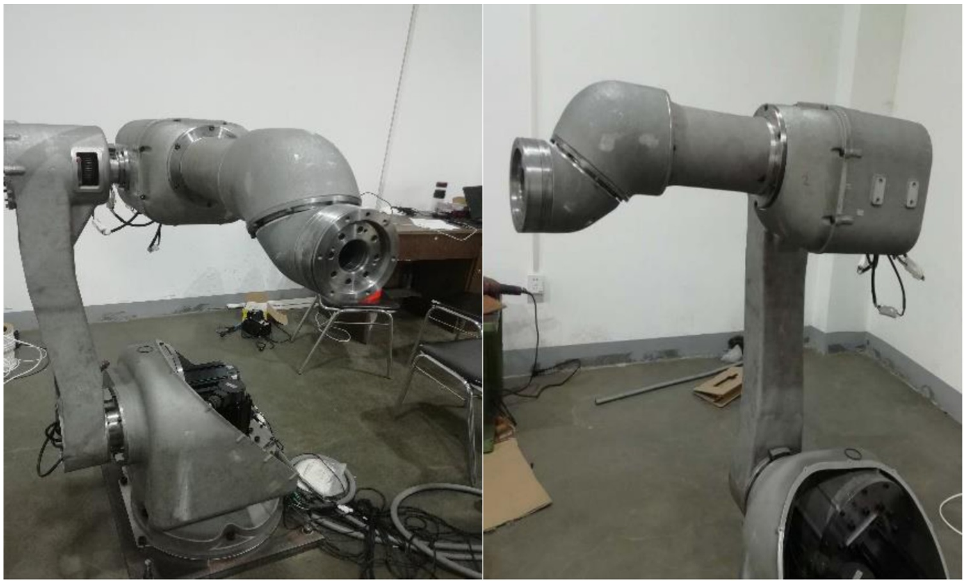

Fig 11. Experimental platform.

https://doi.org/10.1371/journal.pone.0230790.g011

eigenvalues and eigenvectors of *M* matrix. Increasing the speed of this step implies the total cost of the algorithm will be decreased. For the Gauss damped least squares method, the enhanced step length coefficient was proposed. By analyzing the influencing factors of the coefficient, the correlation between the coefficient and the initial iteration value was determined. The concept of optimal enhanced step length coefficient was introduced, and studied using the classification and regression methods of machine learning. By comparing the performance indicators, it was determined that the Decision Tree algorithm had better performance for the study of this problem. Follow-up research expects to obtain more accurate coefficients approaching the optimal enhanced step length coefficients through other methods, so that the number of iterations for each solution can be minimized.

Supporting information

S1 Table. Matlab code.
(M)

S1 Appendix.
(DOCX)

Author Contributions

Conceptualization: Xiaoqi Wang.

Data curation: Heyu Hu.

Formal analysis: Xiaoqi Wang.

Funding acquisition: Jianfu Cao.

Methodology: Xiaoqi Wang.

Project administration: Jianfu Cao.

Software: Lerui Chen.

Supervision: Jianfu Cao.

Validation: Lerui Chen, Heyu Hu.

References

1. Raghavan M, Roth B. Kinematic analysis of the 6R manipulator of general geometry. International symposium on robotics research. 1990; 314–320.
2. Manocha D, Canny J F. Efficient inverse kinematics for general 6R manipulators. IEEE transactions on robotics and automation. 1994; 10(5): 648–657. https://ieeexplore.ieee.org/abstract/document/326569
3. Whitney D E. Resolved motion rate control of manipulators and human prostheses. IEEE Transactions on man-machine systems. 1969; 10(2): 47–53. https://ieeexplore.ieee.org/abstract/document/4081862
4. Kim J S, Chirikjian G S. Inverse kinematic solutions of 6-DOF biopolymer segments. Robotica. 2016; 34(8): 1734–1753. https://www.cambridge.org/core/journals/robotica/article/inverse-kinematic-solutions-of-6dof-biopolymer-segments/594B81D84ACD96DE727885F406D287D3
5. Wampler C W. Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. IEEE Transactions on Systems, Man, and Cybernetics. 1986; 16(1): 93–101. https://ieeexplore.ieee.org/abstract/document/4075580
6. Chiaverini S, Siciliano B, Egeland O. Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. IEEE Transactions on control systems technology. 1994; 2(2): 123–134. https://ieeexplore.ieee.org/abstract/document/294335
7. Buss S R, Kim J S. Selectively damped least squares for inverse kinematics. Journal of Graphics tools. 2005; 10(3): 37–49. https://www.tandfonline.com/doi/abs/10.1080/2151237X.2005.10129202
8. Phuoc L M, Martinet P, Lee S, et al. Damped least square based genetic algorithm with Ggaussian distribution of damping factor for singularity-robust inverse kinematics. Journal of Mechanical Science and Technology. 2008; 22(7): 1330–1338. https://link.springer.com/article/10.1007%2Fs12206-008-0427-4?LI=true
9. Olsen A L, Petersen H G. Inverse kinematics by numerical and analytical cyclic coordinate descent. Robotica. 2011; 29(4): 619–626. https://www.cambridge.org/core/journals/robotica/article/inverse-kinematics-by-numerical-and-analytical-cyclic-coordinate-descent/28CDABA8F4BBD306F18794DDC9E86EB7
10. Fu Z, Yang W, Yang Z. Solution of inverse kinematics for 6R robot manipulators with offset wrist based on geometric algebra. Journal of mechanisms and robotics. 2013; 5: 031010. https://asmedigitalcollection.asme.org/mechanismsrobotics/article/5/3/031010/474846/Solution-of-Inverse-Kinematics-for-6R-Robot
11. Groh F, Groh K, Verl A. On the inverse kinematics of an a priori unknown general 6R-Robot. Robotica. 2013; 31: 455–463. https://www.cambridge.org/core/journals/robotica/article/on-the-inverse-kinematics-of-an-a-priori-unknown-general-6rrobot/768C385146F831380828E8935655A443
12. Dulęba I, Opałka M. A comparison of Jacobian-based methods of inverse kinematics for serial robot manipulators. International Journal of Applied Mathematics and Computer Science. 2013; 23: 373–382.
13. Karpińska J, Tchoń K, Janiak M. Approximation of Jacobian inverse kinematics algorithms: differential geometric vs. variational approach. Journal of Intelligent & Robotic Systems. 2012; 68: 211–224. https://link.springer.com/article/10.1007/s10846-012-9679-4
14. Tørdal S S, Hovland G, Tyapin I. Efficient implementation of inverse kinematics on a 6-dof industrial robot using conformal geometric algebra. Advances in Applied Clifford Algebras. 2017; 27: 2067–2082. https://link.springer.com/article/10.1007/s00006-016-0698-2
15. de Oliveira A S, De Pieri E R, Moreno U F, et al. A new approach to singularity-free inverse kinematics using dual-quaternionic error chains in the davies method. Robotica. 2016; 34: 942–956. https://www.cambridge.org/core/journals/robotica/article/new-approach-to-singularityfree-inverse-kinematics-using-dualquaternionic-error-chains-in-the-davies-method/F62C6555A2A7FCF660C630F5A00533AF
16. Qiao S, Liao Q, Wei S, et al. Inverse kinematic analysis of the general 6R serial manipulators based on double quaternions. Mechanism and Machine Theory. 2010; 45: 193–199. https://www.sciencedirect.com/science/article/pii/S0094114X09001098
17. An H S, Seo T W, Lee J W. Generalized solution for a sub-problem of inverse kinematics based on product of exponential formula. Journal of Mechanical Science and Technology. 2018; 32: 2299–2307. https://link.springer.com/article/10.1007/s12206-018-0441-0

18. Chen Q, Zhu S, Zhang X. Improved inverse kinematics algorithm using screw theory for a six-DOF robot manipulator. International Journal of Advanced Robotic Systems. 2015; 12: 140. https://journals.sagepub.com/doi/full/10.5772/60834

19. Zhao R, Shi Z, Guan Y, et al. Inverse kinematic solution of 6R robot manipulators based on screw theory and the Paden–Kahan subproblem. International Journal of Advanced Robotic Systems. 2018; 15: 1729881418818297. https://journals.sagepub.com/doi/full/10.1177/1729881418818297

20. El-Sherbiny A, Elhosseini M A, Haikal A Y. A comparative study of soft computing methods to solve inverse kinematics problem. Ain Shams Engineering Journal. 2017. https://www.sciencedirect.com/science/article/pii/S2090447917300965

21. KöKer R I. A genetic algorithm approach to a neural-network-based inverse kinematics solution of robotic manipulators based on error minimization. Information Sciences. 2013; 222: 528–543. https://www.sciencedirect.com/science/article/pii/S0020025512005233

22. Köker R, Cakar T, Sari Y. A neural-network committee machine approach to the inverse kinematics problem solution of robotic manipulators. Engineering with Computers. 2014; 30: 641–649. https://link.springer.com/article/10.1007/s00366-013-0313-2

23. Almusawi A R J, Dülger L C, Kapucu S. A new artificial neural network approach in solving inverse kinematics of robotic arm (denso vp6242).s Computational intelligence and neuroscience. 2016. https://www.hindawi.com/journals/cin/2016/5720163/abs/

24. Lopez-Franco C, Hernandez-Barragan J, Alanis A Y, et al. A soft computing approach for inverse kinematics of robot manipulators. Engineering Applications of Artificial Intelligence. 2018; 74: 104–120. https://www.sciencedirect.com/science/article/pii/S0952197618301325

25. Zhou Z, Guo H, Wang Y, et al. Inverse kinematics solution for robotic manipulator based on extreme learning machine and sequential mutation genetic algorithm. International Journal of Advanced Robotic Systems. 2018; 15: 1729881418792992. https://journals.sagepub.com/doi/full/10.1177/1729881418792992

26. Toshani H, Farrokhi M. Real-time inverse kinematics of redundant manipulators using neural networks and quadratic programming: a Lyapunov-based approach. Robotics and Autonomous Systems. 2014; 62: 766–781. https://www.sciencedirect.com/science/article/pii/S0921889014000360

27. Kotsiantis S B. Decision trees: a recent overview. Artificial Intelligence Review. 2013; 39: 261–283. https://link.springer.com/article/10.1007/s10462-011-9272-4

28. Soucy P, Mineau G W. A simple KNN algorithm for text categorization. Proceedings 2001 IEEE International Conference on Data Mining. 2001;647–648.https://ieeexplore.ieee.org/abstract/document/989592

29. Breiman L. Random forests. Machine learning. 2001; 45: 5–32. https://link.springer.com/article/10.1023/A:1010933404324

30. Friedman J H. Stochastic gradient boosting. Computational statistics & data analysis. 2002; 38: 367–378. https://www.sciencedirect.com/science/article/pii/S0167947301000652

31. Chen T, He T, Benesty M, et al. Xgboost: extreme gradient boosting. R package version 0.4–2. 2015: 1–4.