Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105
Volume **9**, Number **4**, November **2024**

Deep Neural Network-Based Trajectory Computation for Collaborative Handling Robots

Zhou-Rong Li

College of Electronic Information
Chongqing Institute of Engineering, Chongqing 400056, P. R. China
cqie_li@163.com

Jin-Yu Xu*

School of Electrical and Information Engineering
Hubei University of Automotive Technology, Shiyan 442002, P. R. China
xujinyuhuat@163.com

Ya-Nan Qian

College of Electronic Information
Chongqing Institute of Engineering, Chongqing 400056, P. R. China
25898453@qq.com

Yong Zhang

College of Electronic Information
Chongqing Institute of Engineering, Chongqing 400056, P. R. China
757286191@qq.com

Sofia Wang

College of Information Technology
Waikato Institute of Technology, Hamilton 3240, New Zealand
vx9992@163.com

*Corresponding author: Jin-Yu Xu
Received January 3, 2024, revised April 12, 2024, accepted August 1, 2024.

Abstract. *In the process of handling bulky items, a single robot has limitations in motion control. Multiple robots, through collaboration, can perform complex operations that are difficult for a single robot to complete. However, multiple robots are affected by many nonlinear factors coupled with each other when they work together, and it is very difficult to establish its accurate mathematical model. Therefore, this paper proposes a deep neural network-based trajectory calculation method for cooperative handling robots. Firstly, the kinematics of industrial handling robots are analysed from the point of view of spatial description and transformation, including single and dual robots. Then, an improved RBF neural network based on Shuffled frog leaping algorithm is proposed. The RBF neural network weights are used as individual frogs, so as to randomly generate a frog population consisting of multiple weight combinations. Secondly, a collaborative dual-robot trajectory estimation method based on improved RBF is designed, and the trajectory calculation model and RBF neural network controller are given. The RBF neural network is used to approximate and compensate the four parameters in the dual-robot dynamics model individually, which reduces the model error while avoiding the influence of mutual coupling between the error functions. The simulation results show that the average tracking errors of joint 1 and joint 2 can be stabilised at 0.8285 ° and 0.7185 °. The average tracking errors of the SFLA-RBF neural network are lower than those of the RBF neural network throughout the iteration process.*

Keywords: Dual robot system; trajectory planning; cooperative handling; RBF neural network; dynamics modelling

1. **Introduction.** With the transformation and upgrading of the manufacturing industry, the application fields of industrial robots are getting bigger and bigger, and the robots are always better able to complete the tasks in workpiece handling, collaborative assembly, parts welding, and item sorting. However, with the expansion of application fields and the emergence of more and more complex tasks, many problems have arisen in the application of industrial robots [1, 2, 3]. For example, the handling of heavy or bulky objects, the need to process workpieces with complex trajectories and the need to move a large working range.

Research on multi-robot collaboration originated from the limitations of single robot capabilities and the need to achieve more efficient and intelligent task execution [4, 5]. In the field of factory automation, multi-robot collaboration can increase productivity, reduce production costs, and enable flexible production line layouts and task assignments. Multi-robot collaboration can also be applied to assist assembly, material handling and packaging [6, 7]. Multi-robot collaboration can be used in automated warehousing, material handling and order processing to improve logistics efficiency, reduce labour costs, and improve the precision and safety of warehouse management.

Multi-robot collaboration refers to the nature of multiple robots working together with each other when completing the same task [8. 9]. For complex tasks, such as handling irregularly shaped or larger and heavier objects, when a single robot is unable to complete the handling task due to its own limitations, multiple robots can replace a single robot to complete the task through collaborative work, which also improves the efficiency of the robotic system in the process of the operation, and enables the multi-robot system to solve more practical application problems.

The problem of determining the relationship between the base coordinate systems of two robots by some means or method is the problem of studying the calibration of the relationship between the base coordinate systems of a two-robot base coordinate system [10]. The relationship between the base coordinate systems of the robots is represented by a chi-square matrix containing position vectors and rotation matrices. Since the Z-axis of the robot's base coordinate system is generally located inside the robot, it is not

possible to obtain the relationship between the base coordinate systems of the two robots by direct measurements [11], and therefore it can only be done by indirect measurements or measurements with the help of advanced equipment.

Robot trajectory planning has a very important role in the control of industrial robots, which directly affects the rapidity and accuracy of robot control. The planning of the robot contains three parts in total [12]: task planning, action planning, and trajectory planning. Trajectory planning is to design the target motion trajectory according to the requirements of the robot operation task, and to describe its curve trajectory and motion line. If two robots or multiple robots instead of a single robot work together to complete a task, the problem can be solved with high quality and efficiency. However, multiple robots are affected by many non-linear factors coupled with each other when they work together, and it is very difficult to establish an accurate mathematical model for it.

With the advances in artificial intelligence, sensors, and communication technologies, research on multi-robot collaboration has been better supported [13, 14]. For example, the development of technologies such as distributed intelligence algorithms, machine learning, and sensor networks provides more possibilities for the realisation of multi-robot collaboration. Therefore, the research objective of this work is to propose a dual-robot collaborative handling trajectory computation method to improve the efficiency and accuracy of the robots, so as to solve the problem of precise synchronisation of dual-robot handling.

1.1. **Related Work.** trajectory planning for dual robotic systems is one of the more researched problems in robotic systems in recent years, where robots are controlled to complete complex tasks through trajectory planning and collaboration between robots.

Multi-robot cooperative problem is one of the hot issues in the field of robotics research, in the early 80's, the research of multi-robot cooperative was in the initial stage, how to maintain the movement of multi-robots is the key problem of multi-robot cooperative must be solved, Vergnano et al. [15] pointed out that in the multi-robot cooperative operation, the movement cooperation is the most basic form of operation, and the dynamics cooperation problem should be considered according to the needs of cooperative operation. However, there are few systematic studies on kinematic constraint analysis and path planning methods for complex cooperative motions. Antonelli and Astanin [16] investigated the kinematic constraints in cooperative welding and cooperative handling and demonstration methods for cooperative trajectory planning. During the robots' collaborative following motion, the schematic teaching method of the slave robot end motion path can be obtained based on the known master robot end motion path, and the results show that the master robot end pose remains consistent during the motion. This strategy lays a certain foundation for the development of next-generation robot controllers with collaborative functions. Čáp et al. [17] addresses the multi-robot motion coordination planning problem and proposes an asynchronous decentralised prioritized planning It highlights the limitations of existing decentralized prioritized planning algorithms, which contain synchronization points that all agents must pass synchronously, and demonstrates that the proposed method can converge faster than both synchronous decentralised and centralised algorithms. Ahmadzadeh and Masehian [18] presents a planning system that enables parallelisation of complex task and motion planning problems by iteratively It combines optimisation methods to jointly solve for manipulation constraints with a sampling-based bi-directional space-time path planner, allowing for cooperative multi-robot manipulation with arrival unknown. It combines optimisation methods to jointly solve for manipulation constraints with a sampling-based bi-directional space-time path planner, allowing for cooperative multi-robot manipulation with unknown arrival-times.

Guo et al. [19] proposed a real-time cooperative control of multiple robots using synchronised visual feedback controllers. Zhang et al. [20] proposed a distributed scheduling method based on a local priority strategy, which achieves rational scheduling and cooperative operation of individual robots by assigning time slots and principles to a swarm of robots. However, the robots need to perform predefined tasks in relatively fixed environments, which may have some limitations in practical applications. Jin et al. [21] proposed a novel multi-robot coordination method, which enables a group of distributed robots to reach a coherent state quickly by using a consensus algorithm. However, the approach focuses on robot coordination in a specific network environment, and thus may be problematic in the absence of a network or when communication links are broken.

Neural network control is a control method gradually developed at the end of the 20th century, its biggest advantage is that it can approximate any complex nonlinear system, and is often used as a controller or a discriminator, which is able to solve some control problems of complex systems.

1.2. **Motivation and contribution.** Deep neural networks can effectively learn and approximate complex, highly nonlinear system dynamics models [22, 23]. In dual-robot cooperative trajectory planning, the system often has a variety of complex nonlinear factors, and deep neural networks can better capture these factors and improve the accuracy and robustness of trajectory planning [24]. Therefore, in order to solve the problem of difficult control when dual robots work together, a deep neural network-based trajectory calculation method for cooperative handling robots is proposed. The main innovations and contributions of this paper include:

(1) The kinematic model of dual robot trajectory synchronisation is investigated as an example of dual robot cooperative handling task, and the mutual expression equations of dual robot cooperative motion are given.

(2) In order to improve the training efficiency and performance of Radial Basis Function (RBF) neural network [25], Shuffled Frog Leaping Algorithm (SFLA) [26] is used to optimise the weights of RBF neural network. The training of RBF neural network usually needs to be done by optimisation algorithms such as gradient descent to continuously adjust the weights, while the SFLA algorithm can find effective parameter combinations more quickly by means of group intelligence.

(3) A collaborative dual robot trajectory estimation method based on SFLA-RBF is designed, and the trajectory calculation model and RBF neural network controller are given. The SFLA-RBF neural network is used to approximate and compensate the four parameters in the dual-robot dynamics model individually, which reduces the model error while avoiding the influence of mutual coupling between the error functions.

2. Kinematic analysis of industrial handling robots.

2.1. **Spatial description and transformation.** In order to realise complex manipulation tasks, robots are usually composed of a series of connecting rods and corresponding kinematic attachments. Therefore, describing the relative motion relationships between robot links, as well as between them and the manipulated objects, plays an extremely important role in the study of the robot's motion and mode of operation. In describing the orientation relations of a rigid body, a coordinate system is first specified, relative to which the positions of points can be represented by three-dimensional column vectors. The orientation of a rigid body can be represented by a 3×3 rotation matrix. And a 4×4 chi-square transformation matrix can unify the orientation description of the rigid body. Until now, a uniform description of the robot's position has been given in the form of a

uniform chi-square coordinate matrix, on the basis of which subsequent kinematic studies and analyses have been realised.

1) Location Description.

The right-angle coordinate system $\sum_i$ is used to describe the position, and the position of any point P in space can be represented by a 3×1 column vector ${}^{A}P$, i.e., the position vector.

$$ {}^{A}P = \begin{pmatrix} P_x \\ P_y \\ P_z \end{pmatrix} \tag{1} $$

where P_x, P_y, P_z are the three coordinate components of point P in the coordinate system $\sum_i$.

2) Orientation description.

Set up a right-angled coordinate system $\sum_A$ connected to a rigid body A in space. Use the three principal vectors X_A, Y_A, Z_A of the coordinate system $\sum_A$ to form a 3×3 matrix.

$$ {}^{B}_{A}R = \left({}^{B}X_A \; {}^{B}Y_A \; {}^{B}Z_A\right) = \begin{pmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{pmatrix} \tag{2} $$

where ${}^{B}_{A}R$ is called the rotation matrix and the superscript B represents the selected reference coordinate system $\sum_B$.

Since the three column vectors X_A, Y_A, Z_A are unit principal vectors and are perpendicular to each other, the nine elements of the rotation matrix satisfy the following six constraints (also called orthogonality conditions):

$$ \begin{cases} {}^{B}X_A \cdot {}^{B}X_A = {}^{B}Y_A \cdot {}^{B}Y_A = {}^{B}Z_A \cdot {}^{B}Z_A = 1 \\ {}^{B}X_A \cdot {}^{B}Y_A = {}^{B}Y_A \cdot {}^{B}Z_A = {}^{B}Z_A \cdot {}^{B}X_A = 0 \end{cases} \tag{3} $$

Thus, the rotation matrix ${}^{B}_{A}R$ is unit-orthogonal and the inverse of ${}^{B}_{A}R$ is the same as its transpose with a determinant value of 1. Typically, position vectors are used to describe the position of a point, while rotation matrices are used to describe the orientation of an object.

In order to describe the position and attitude of the robot end-effector, a reference coordinate system $\sum_A$ is chosen. The end-effector is defined to be solidly connected to a coordinate system, called the end-effector coordinate system $\sum_B$. The Z-axis is located close to the object and the X-axis is determined according to the right-hand rule. Thus, the orientation rotation matrix of the end-effector is shown below:

$$ {}^{A}_{T}R = (n, o, a') \tag{4} $$

where a' is the approach vector, n is the normal vector, and o is the orientation vector. We use the position vector P to describe the position of the end-effector, then the end-effector's bit position can be described by four vectors (n, o, a', P).

2.2. **Kinematics modelling of single robot.** Any robot configuration can be modelled using the D-H method. In addition, it can be used to represent any possible combination of robot joints and linkages.

The total transformation matrix of the robot is obtained by connecting all the transformations of the robot starting from the bottom base until the last joint to each other in

order [27]. Figure 1 represents the three joints of the linkage (joints n, $n+1$, and $n+2$) in relation to each other, each of which is rotatable or translatable.

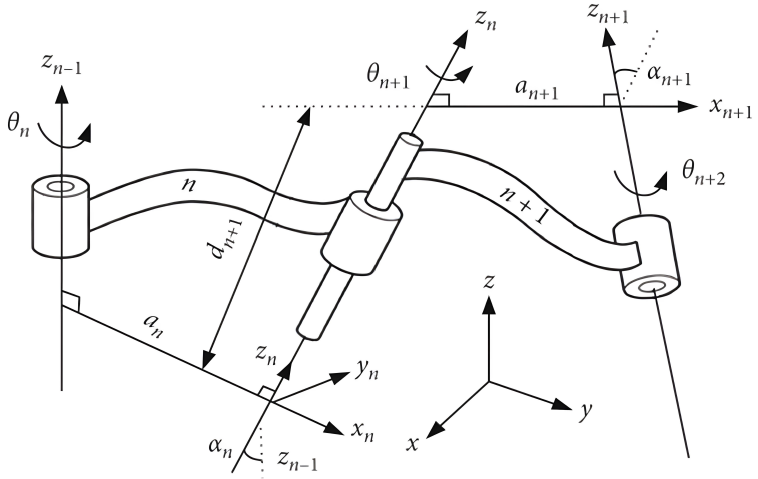

Figure 1. D-H representation of the universal joint-link combination

In order to model the robot linkages and joints using the D-H algorithm, a local reference coordinate system needs to be specified for each joint. Therefore, for each linkage joint of the robot, a z-axis and an x-axis must be specified, and generally the y-axis is not specified because the y-axis can be derived from the right-hand rule and the direction is always perpendicular to the z-axis and the x-axis. The transformation matrix A is represented as six right-multiplication matrices. Since all transformations are transformations with respect to the current coordinate system, all matrices are right-multiplied. On the base at the bottom of the robot, the first joint is transformed sequentially to the end joints and finally to the robot end-effector by means of matrix positional transformations. If each matrix transformation is defined as A_{n+1}, many A matrices representing the transformations are obtained. The total transformation between the bottom base of the robot and the arm is shown below:

$$^{R}T_{H} =^{R} T_1^{-1}T_2^2T_3, \ldots, T_n = A_1A_2A_3, \ldots, A_n \tag{5}$$

where n is the number of robot joints. For a six-axis robot, there are six A matrices.

In summary, the end spatial attitude of the six-axis robot is obtained by kinematic orthogonal solution of the six-axis robot and coordinate transformation with given D-H coordinate parameters, and any position in the range of the robot's workspace can be represented by the end attitude, which lays a foundation for the subsequent dual-robot trajectory planning and collaboration.

Robot inverse kinematics is the basis for robot motion planning and trajectory control. Since solving the kinematic equations is a nonlinear problem, it is difficult to study the inverse kinematic solution problem, and the existence of its solutions and multiple solutions must be considered. Algebraic solution method in robot inverse kinematics equations is convenient for real-time control, fast computation and high efficiency, so algebraic solution method is used as an example to solve the problem. In the process of robot inverse kinematics calculation, it is learnt that when the robot's end position is fixed and kept unchanged, the solution is the existence of multiple solutions, in other words, the robot inverse kinematics has multiple solutions. A solution closest to the current robot can be selected as the optimal solution.

2.3. Kinematics analysis of dual-robot. In the process of dual-robot synchronous motion operation, there must exist certain constraint relationship between the two robots, when the two robots start to move at the same time, the relative positions and attitudes of their end-effector remain unchanged, so a strong synchronisation between the two robots is required.

For example, when a two-robot system is handling a larger and heavier box, in order to keep the relative position constraints at the end of the two robot's constant, the two robots need to keep their motions in a synchronised state. In this paper, we will study the dual-robot synchronous kinematics model as an example of the dual-robot cooperative handling task. Figure 2 shows an example of two-robot cooperative and synchronous handling of a box.

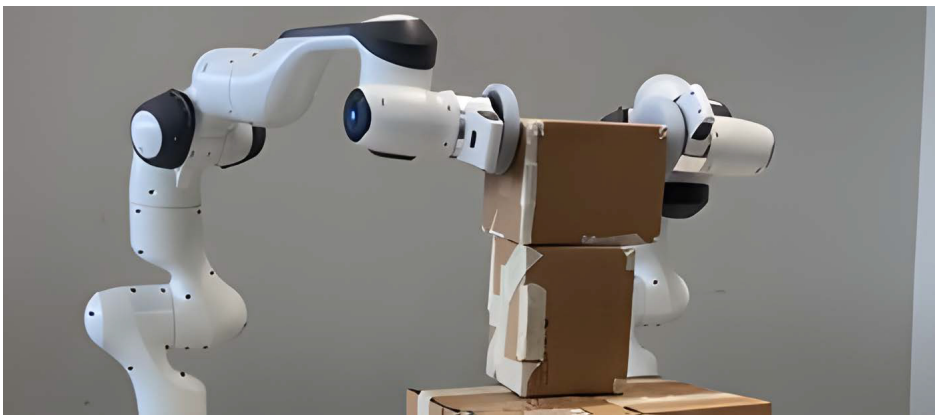

Figure 2. Example of synchronised handling dual-robot

After modelling the kinematics of a single robot, a kinematic model of a dual robot is required to study the cooperative motion of the dual robot. When modelling the kinematics of the two-robot system, only the relationship between the base coordinate systems of the two robots needs to be investigated as the kinematic model of each single robot has been established previously. As the two six-axis robots are handling the box, the positional relationship of their end-effector needs to be represented in their respective base coordinate systems, which will increase the computational complexity. Therefore, the kinematic equations between the two robots need to be established on the basis of the forward kinematics as well as the inverse kinematics of a single robot, so that the computational relationship between the two is determined in the same base coordinate system. It is assumed that the end-effector attitude of the robot under study does not change.

The world coordinate system is the absolute coordinate system of the system independent of the base position. The coordinate transformation of the base coordinate system of master robot ① with respect to the world coordinate system is indicated by A_1^E, and the coordinate transformation of the base coordinate system of slave robot ② with respect to the world coordinate system is indicated by A_2^E.

$$O_E = A_1^E \cdot O_1 \tag{6}$$

$$O_E = A_2^E \cdot O_2 \tag{7}$$

When the world coordinate system is selected to coincide with O_1, the relative position relationship between robot ① and robot ② can be obtained.

$$O_1 = A_2^E \cdot O_2 \tag{8}$$

$$A_2^{-1} = O_1 \cdot O_2^{-1} \tag{9}$$

When the base coordinate system of the master and slave robots is known, Equation (9) can be used to find out the position transformation array A_2^{-1} of the base coordinate system of the slave robot relative to the base coordinate system of the master robot. In order to distinguish it from the transformation array of the joint coordinate system, the transformation array of the base coordinate system is denoted as T_{12}. Solving T_{12}, the relative motion relationship between robot ① and robot ② is established, which completes the mutual expression of the cooperative motion of the two robots.

3. Improved RBF neural network design.

3.1. RBF neural network.

RBF neural network is an important type of deep neural network. RBF neural network is characterised by a radial basis function, which simulates nonlinear relationships through a series of nonlinear transformations. The network usually consists of three layers [27]: an input layer, a hidden layer and an output layer, as shown in Figure 3.

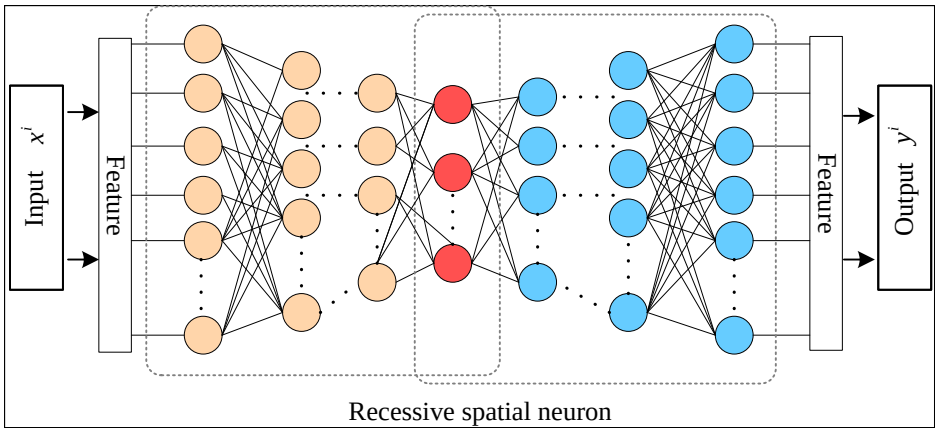

Figure 3. Structure of the RBF

Compute the Euclidean distance between the input vector x and each centre vector c_j of the network.

$$D_j = \|x - c_j\| \tag{10}$$

where D_j is the distance between the input and the centre of the j-th RBF neuron.

This distance is transformed using a radial basis function to get the output of that neuron. The most commonly used radial basis function is the Gaussian function. The activation function for the j-th RBF neuron is shown as follows:

$$h_j = \exp(-\beta_j \cdot D_j^2) \tag{11}$$

where β_j is the width parameter of this neuron.

The output of the RBF neural network can be obtained by summing the weights of the output layer and the outputs of all the neurons in the hidden layer. The output of the RBF neural network is shown as follow:

$$y = \sum (w_j * h_j) \tag{12}$$

Unlike many other types of neural networks, RBF neural networks are typically trained in a two-stage approach. The centre vector and width parameters are fixed first, and then the output weights are trained. For example, we can train the weights using the least squares method with the objective function shown as follow:

$$E = \sum (t - y)^2 \tag{13}$$

The minimum error is obtained by taking the derivative of the objective function and making the derivative equal to zero. Using the mean square error as the loss function.

$$E = \frac{1}{2} \sum_{k=1}^{K} (t_k - o_k)^2 \tag{14}$$

where t_k is the target output of the training sample.

Once the centre c_j and the width parameter β_j are fixed, the minimisation of the objective function E can be achieved simply by adjusting the weights w_j of each neuron, thus completing the training process of the RBF neural network.

RBF neural network is a neural network based on radial basis functions [28], which models nonlinear relationships through nonlinear transformations, and has better approximation performance and nonlocal generalisation ability for pattern recognition and prediction in a variety of problems.

3.2. **Shuffled frog leaping algorithm.** The SFLA is a new heuristic population evolutionary algorithm [29] with efficient computational performance and excellent global search capability.

SFLA divides the population into m subpopulations $S = S_1, S_2, ..., S_m$, each subpopulation is of size n, then the population size is $m \times n$. The position of each frog is updated in the following way:

$$x_{ij} = x_{ij} + rand(-1, 1) * (x_{ij} - x_{kj}) \tag{15}$$

where x_{ij} denotes the position of the ith frog in the j-th dimension, x_{kj} denotes the position of the other selected frog in the corresponding dimension, and $rand()$ is a random number.

By doing this, the frog's position is updated based on the position of another frog and a random factor is introduced to increase the diversity of the algorithm. For each subpopulation S_i, the frog's fitness is calculated as shown as follow:

$$fitness_i = f(x_i) \tag{16}$$

The fitness of each frog is calculated based on the objective function $f(x)$ defined by the problem. The fitness is used in the SFLA algorithm to measure how well the frogs fit the optimisation problem, which in turn affects the frog's position update and selection in the next iteration. By introducing the global optimal information to guide the local search to achieve the global search, the hybrid frog hopping algorithm combines the advantages of population global search and local search.

3.3. **SFLA-RBF.** There are two main ways to optimise RBF neural networks, one way is parameter optimisation. The structure and performance of the RBF network can be optimised by adjusting the parameters such as the number and distribution of the centre vectors, the width of the basis function, and the weights from the hidden layer to the output layer. Commonly used methods include feed-forward RBF networks, regularisation techniques, Bayesian training methods, etc.

Another way is structural optimisation. The performance can be optimised by changing the connection structure of RBF network, such as adding shortcut connections, using cascade structure, etc. These structural optimisations can reduce the training error and prevent overfitting. These structural optimisations can reduce the training error and prevent overfitting. In this work, parameter optimisation of RBF neural network is chosen. The steps of the proposed SFLA-RBF neural network are as follows:

1) Initialising a group of frogs: first, we need to initialise a group of frogs, each representing a possible combination of weights in the RBF neural network. These initial weights can be randomly generated or a set of weights chosen based on experience and prior knowledge.

2) Calculate the fitness: next, for each frog, we need to calculate its fitness based on the performance metrics of the RBF neural network. This can be done by calculating the prediction error of the neural network or other performance metrics.

3) Position update: The position update of frogs can be performed by Equation (15), which increases the diversity of the algorithm and the global search capability.

4) Selective updating: at the end of each iteration, the weights are selectively updated according to the frogs' adaptability, and the frogs with high adaptability are kept and their positions are updated. This step-by-step iteration can finally get the optimised RBF neural network weight combination.

The pseudo-code of the proposed SFLA-RBF neural network is shown in Algorithm 1.

Algorithm 1 SFLA-RBF Neural Network

Input: Number of nodes in RBF input layer, number of nodes in hidden layer, number of nodes in output layer; Population size, subpopulation size, and number of iterations for SFLA

Output: Optimized RBF network weights (center vector, variance, and weights from hidden layer to output layer)

```
Encode the RBF network weights into a frog (position vector)
Initialize the position of the n frogs in the SFLA population
Divide the population into m subpopulations, each with n/m frogs
while iterations not reaching maximum number of iterations do
    for each subgroup do
        Calculate the fitness of each frog (error of the corresponding RBF network)
        Find the best and worst frogs within the subpopulation X_best, X_worst
        for each frog do
            Generate new positions according to Equation (15)
            if X_new is better than X_old then
                Replace X_old with X_new
            end if
        end for
    end for
    Regroup populations and disrupt sequences
end while
Return the best frog position, i.e., the optimal weights of the RBF network
```

4. Improved RBF-based computation of trajectory estimation for collaborative dual robots.

4.1. **Establishment of trajectory calculation model.** Based on the mathematical model of dual robot kinematics in Equation (8) and Equation (9), the mathematical model of dual robot kinematic position can be introduced as follows:

$$M(q)\ddot{q} + C(\dot{q})\dot{q} + D(\dot{q})\dot{q} + g(q) = \tau - \tau_d \tag{17}$$

where $q(t)$ denotes the real-time position of the underwater robot; $M(q) \in \mathbb{R}^{6\times6}$ denotes the matrix of inertia coefficients of the dual robot; $C(\dot{q}) \in \mathbb{R}^{6\times6}$ denotes the matrix of centripetal force coefficients; $D(\dot{q}) \in \mathbb{R}^{6\times6}$ denotes the matrix of damping forces; $g(q) \in \mathbb{R}^{6\times6}$ denotes the combined moments of gravitational and frictional forces; $\tau \in \mathbb{R}^{6\times1}$ denotes the output control moments; τ_d denotes the external disturbance.

Define the expected value of the dual robot running trajectory at the moment t as $q_d(t)$ and the actual value of the dual robot running trajectory as $q(t)$, then the trajectory error $e(t)$ is shown as follow:

$$e(t) = q_d(t) - q(t) \tag{18}$$

$$f = M(q)(\ddot{q}_d + \Delta e) + C(\dot{q})(\dot{q}_d + \Delta e) + D(\dot{q})(\dot{q}_d + \Delta e) + g(q) \tag{19}$$

The four parameters $M(q)$, $C(\dot{q})$, $D(\dot{q})$ and $g(q)$ are often unknown in real engineering, so in order to estimate the sub-error functions in the model more accurately, this work adopts four RBF neural networks to approximate each of the four parameters in f individually, which improves the accuracy of the motion control model while avoiding the mutual coupling among the sub-error functions of the model effectively.

4.2. **SFLA-RBF Neural Network Controller Design.** In general, Gaussian function is chosen for the basis function of the RBF neural network and its approximation algorithm is shown as follow:

$$h_i = g\left(\frac{\|x - c_i\|^2}{b_i^2}\right), \quad i = 1, 2, \ldots, n \tag{20}$$

$$y = w^T h(x) \tag{21}$$

where x, y are the inputs and outputs of the RBF neural network; h_i denotes the output of the i-th hidden layer unit; c_i denotes the centre of the Gaussian function; b_i denotes the width of the Gaussian function; $h(x) = [h_1, h_2, \ldots, h_n]^T$ is a column vector consisting of the outputs of each hidden layer unit; w denotes the weights of $h(x)$ mapped to the output layer.

From the approximation algorithm of the RBF neural network, it can be seen that the input layer of the RBF neural network and the hidden layer mainly rely on the Gaussian function to achieve a nonlinear mapping relationship. The hidden layer and the output layer mainly rely on the linear sum relationship implemented by the weights, and this linear relationship greatly accelerates the learning speed of the network. In addition, the parameter w is adjustable, so there always exists an ideal vector of weights, which makes the error value of the RBF neural network approximating the continuous function $f(\cdot)$ converge to a very small number.

$$\max \left\| f(\cdot) - \hat{f}(\cdot) \right\| \leq \varepsilon_0 \tag{22}$$

$$f = w^T h + \eta(x) \tag{23}$$

where η denotes the neural network estimation error.

In the proposed SFLA-RBF adaptive controller, assuming the existence of ideal weights W, the ideal network output is shown as follow:

$$f = W^T h \tag{24}$$

The estimated output of the SFLA-RBF neural network is shown as follow:

$$\hat{f} = \hat{W}^T h \tag{25}$$

where $\hat{W}$ is the estimated weights, the estimation error of SFLA-RBF neural network $\xi(x) = |f - \hat{f}|$.

For the mathematical model in Equation (17), the estimated output of the error function of the SFLA-RBF neural network is shown as follow:

$$\begin{cases} M(q) = M_0(q) + E_M \\ C(\dot{q}) = C_0(\dot{q}) + E_C \\ D(\dot{q}) = D_0(\dot{q}) + E_D \\ g(q) = g_0(q) + E_g \end{cases} \tag{26}$$

where E_M, E_C, E_D and E_g are the approximation errors of the SFLA-RBF neural network for the four parameters, respectively. The calculations for estimating the four parameters using RBF neural network are shown as follow:

$$\begin{cases} \hat{M}(q) = \left[\hat{W}_M\right]^T \cdot h_M(q) \\ \hat{C}_0(q) = \left[\hat{W}_C\right]^T \cdot h_C(q) \\ \hat{D}_0(q) = \left[\hat{W}_D\right]^T \cdot h_D(q) \\ \hat{g}_0(q) = \left[\hat{W}_g\right]^T \cdot h_g(q) \end{cases} \tag{27}$$

where W_M, W_C, W_D, and W_g denote the ideal weights of the approximation error subfunctions of the SFLA-RBF neural network; $h_M(q)$, $h_C(q)$, $h_D(q)$, and $h_g(q)$ are the output matrices of the implicit layer; and $\{\hat{W}_M\}$, $\{\hat{W}_C\}$, $\{\hat{W}_D\}$, and $\{\hat{W}_g\}$ are the estimated values.

5. Cooperative handling simulation tests.

5.1. **Experimental setup.** In order to verify the performance of SFLA-RBF neural network for trajectory control of collaborative dual robots, simulation tests were carried out as an example of collaborative handling of boxes by dual robots.

The SFLA-RBF neural network is used to implement the cooperative dual robot trajectory control to achieve the horizontal handling of rectangular box. The sampling frequency of the experimental data is set to 500 ms, the length of time is set to 60 s, and the linear accuracy of displacement sensor is 0.1%. The roadmap of the horizontal handling of the box by the dual robots is shown in Figure 2. The size of the box is 150 mm × 100 mm × 60 mm.

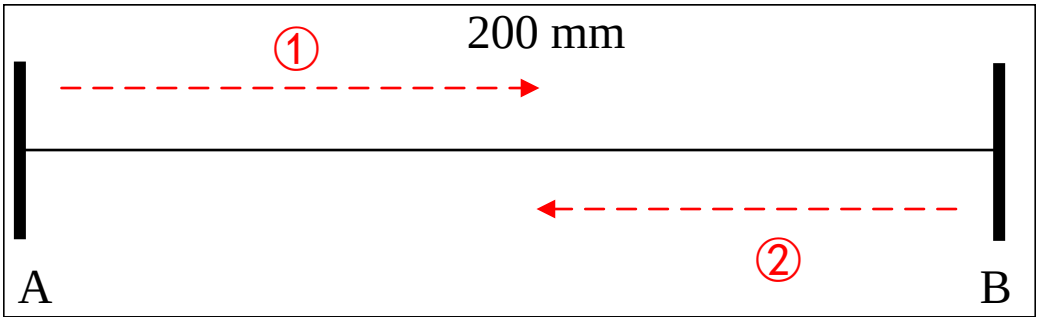

Figure 4. Roadmap of horizontal handling box motion for dual-robots

Let the target angles of joint 1 and joint 2 be $q_{1d}(t)$ and $q_{2d}(t)$, respectively, as shown in Equation (28) and Equation (29). Derive the above 2 target angles to obtain the target angular velocities $\dot{q}_{1d}(t)$ and $\dot{q}_{2d}(t)$ respectively.

$$q_{1d}(t) = \begin{cases} 0.5\cos(0.1t), & 0 < t \leq 20\pi \\ 0.5\cos(0.3t), & 20 < t \leq 30\pi \\ 0.5\cos(0.5t), & 30 < t \leq 40\pi \end{cases} \tag{28}$$

$$q_{2d}(t) = \begin{cases} 0.8\cos(0.1t), & 0 < t \leq 20\pi \\ 0.8\cos(0.3t), & 20 < t \leq 30\pi \\ 0.8\cos(0.5t), & 30 < t \leq 40\pi \end{cases} \tag{29}$$

5.2. **Tracking accuracy simulation.** RBF neural network and SFLA-RBF neural network are used to simulate the target angle of joint 1 and joint 2 of the robot respectively, and the tracking error is used as the evaluation criterion. The simulation results are shown in Table 1. From Table 1, it can be seen that with the increase of network size, the

Table 1. Tracking angle error of dual-robots motion track

Nodal	**Number of hidden layer neurons**	**Maximum tracking error/°**	**Average tracking error/°**
1	3	7.0884	3.2416
	5	4.1350	2.1248
	8	2.3387	1.3311
	10	1.9375	0.8285
	15	1.9371	0.8285
2	3	7.0475	3.3424
	5	4.6347	2.0562
	8	2.3460	1.1134
	10	1.1381	0.7185
	15	1.1386	0.7185

angular tracking error of robot node 1 and node 2 gradually decreases and the tracking accuracy increases. Moreover, when the number of hidden layer neurons reaches 10, the average tracking error degree reaches a stable value and no longer decreases with the increase of the number of neurons, this is because the maximum angle tracking error degree of the two nodes varies very little when the number of hidden layer neurons is 10 and 15, respectively. Eventually, when the average tracking error no longer varies, it reaches a stable value of 0.8285 and 0.7185 respectively.

The following trajectory tracking simulations are performed for RBF and SFLA-RBF, respectively, with the number of hidden layer neurons set to 10, and the error tracking results are shown in Figure 5.

As can be seen in Figure 5, the two curves represent the variation of the average tracking error with the number of iterations for the RBF neural network and the SFLA-RBF neural network, respectively. The red solid line represents the RBF neural network, and it can be seen that this curve decreases rapidly from some higher value and smoothes out to converge to some lower level. The black dashed line represents the SFLA-RBF neural network, and this curve starts from the same starting point, but declines more rapidly than the red curve, and quickly stabilises to a level lower than the red curve. It can be seen that the average tracking error of the SFLA-RBF neural network decreases faster at the

beginning of the iteration but slower at the later stages, while the average tracking error of the RBF neural network decreases slower throughout the iteration. This indicates that the SFLA-RBF neural network has a faster learning speed at the beginning but slower at the later stages, while the RBF neural network learns slower throughout the learning process.

It can also be seen that the average tracking error of the SFLA-RBF neural network is lower than the average tracking error of the RBF neural network throughout the iterations. This indicates that SFLA-RBF neural network has better tracking accuracy. In summary, under the same neural network size, SFLA-RBF neural network has better tracking accuracy and faster learning speed, which is a more suitable network for collaborative dual robot motion planning.

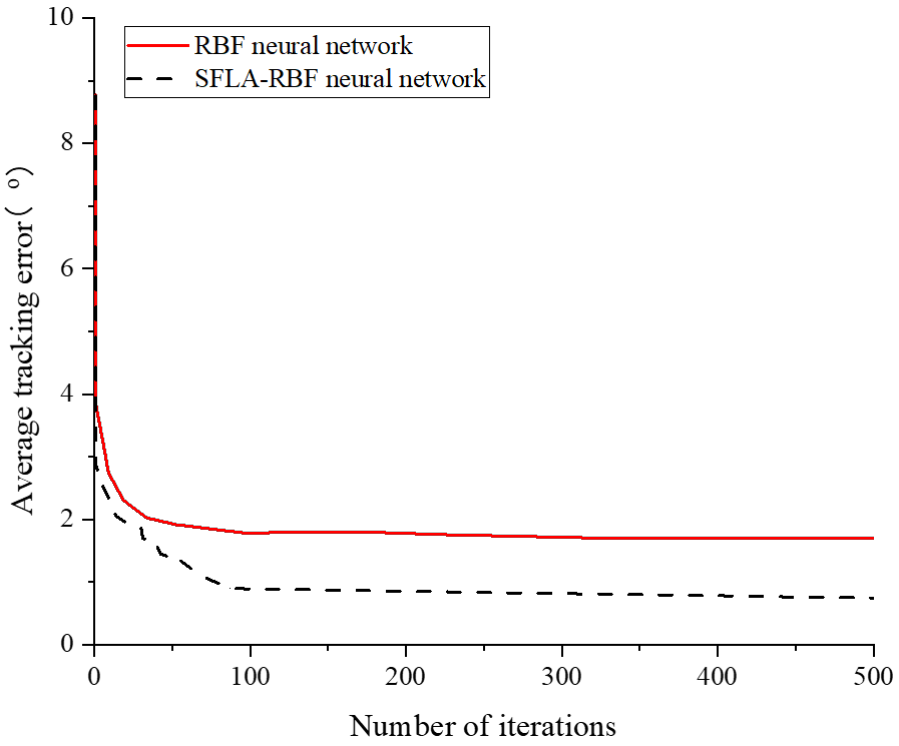

Figure 5. Roadmap of horizontal handling box motion for dual-robots

6. **Conclusion.** In order to solve the problem of difficult control when dual robots work together, a deep neural network-based trajectory calculation method for cooperative handling robots is proposed. Taking the two-robot cooperative handling task as an example, the kinematic model of two-robot trajectory synchronisation is investigated, and the mutual expression equations of two-robot cooperative motion are given. SFLA is used to optimise the weights of the RBF neural network, and the SFLA algorithm finds effective combinations of weight parameters more quickly by means of group intelligence. The SFLA-RBF based trajectory estimation method for collaborative dual robots is designed, and the trajectory calculation model and RBF neural network controller are given. The SFLA-RBF neural network is used to approximate and compensate the four parameters in the dual robot dynamics model individually. The simulation results show that the SFLA-RBF neural network has better tracking accuracy and faster learning speed under the same neural network size, and is a more suitable network for cooperative dual robot motion planning.

Acknowledgement. This work was supported by Science and Technology Youth Fund Project of Chongqing Municipal Commission of Education (No. KJQN202301916), and Science and Technology Youth Fund Project of Chongqing Municipal Commission of Education (No. KJQN202301918).

REFERENCES

[1] M. Slamani, A. Nubiola, and I. Bonev, "Assessment of the positioning performance of an industrial robot," *Industrial Robot: An International Journal*, vol. 39, no. 1, pp. 57-68, 2012.

[2] J. Arents and M. Greitans, "Smart industrial robot control trends, challenges and opportunities within manufacturing," *Applied Sciences*, vol. 12, no. 2, 937, 2022.

[3] T. Brogårdh, "Present and future robot control development—An industrial perspective," *Annual Reviews in Control*, vol. 31, no. 1, pp. 69-79, 2007.

[4] E. Abele, M. Weigold, and S. Rothenbücher, "Modeling and identification of an industrial robot for machining applications," *CIRP Annals*, vol. 56, no. 1, pp. 387-390, 2007.

[5] A. Lopes and F. Almeida, "A force–impedance controlled industrial robot using an active robotic auxiliary device," *Robotics and Computer-Integrated Manufacturing*, vol. 24, no. 3, pp. 299-309, 2008.

[6] Z. Pan, J. Polden, N. Larkin, S. Van Duin, and J. Norrish, "Recent progress on programming methods for industrial robots," *Robotics and Computer-Integrated Manufacturing*, vol. 28, no. 2, pp. 87-94, 2012.

[7] M. Švaco, B. Šekoranja, F. Šuligoj, and B. Jerbić, "Calibration of an industrial robot using a stereo vision system," *Procedia Engineering*, vol. 69, pp. 459-463, 2014.

[8] G. Ferretti, G. Magnani, P. Putz, and P. Rocco, "The structured design of an industrial robot controller," *Control Engineering Practice*, vol. 4, no. 2, pp. 239-249, 1996.

[9] D. Massa, M. Callegari, and C. Cristalli, "Manual guidance for industrial robot programming," *Industrial Robot: An International Journal*, vol. 42, no. 5, pp. 457-465, 2015.

[10] K. Young and C. G. Pickin, "Accuracy assessment of the modern industrial robot," *Industrial Robot: An International Journal*, vol. 27, no. 6, pp. 427-436, 2000.

[11] T. Brogårdh, "Robot control overview: An industrial perspective," *Modeling, Identification and Control*, vol. 30, no. 3, 167, 2009.

[12] K. Wang, Z. Chen, X. Dang, X. Fan, X. Han, C.-M. Chen, W. Ding, S.-M. Yiu, and J. Weng, "Uncovering Hidden Vulnerabilities in Convolutional Neural Networks through Graph-based Adversarial Robustness Evaluation," *Pattern Recognition*, vol. 143, 109745, 2023.

[13] F. Zhang, T.-Y. Wu, and G. Zheng, "Video salient region detection model based on wavelet transform and feature comparison," *EURASIP Journal on Image and Video Processing*, vol. 2019, pp. 1-10, 2019.

[14] F. Zhang, Y. Wang, and C. Wu, "An automatic generation method of cross-modal fuzzy creativity," *Journal of Intelligent & Fuzzy Systems*, vol. 38, no. 5, pp. 5685-5696, 2020.

[15] A. Vergnano, C. Thorstensson, B. Lennartson, P. Falkman, M. Pellicciari, F. Leali, and S. Biller, "Modeling and optimization of energy consumption in cooperative multi-robot systems," *IEEE Transactions on Automation Science and Engineering*, vol. 9, no. 2, pp. 423-428, 2012.

[16] D. Antonelli and S. Astanin, "Qualification of a collaborative human-robot welding cell," *Procedia CIRP*, vol. 41, pp. 352-357, 2016.

[17] M. Čáp, P. Novák, A. Kleiner, and M. Selecký, "Prioritized planning algorithms for trajectory coordination of multiple mobile robots," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 3, pp. 835-849, 2015.

[18] H. Ahmadzadeh and E. Masehian, "Modular robotic systems: Methods and algorithms for abstraction, planning, control, and synchronization," *Artificial Intelligence*, vol. 223, pp. 27-64, 2015.

[19] J. Guo, X. Jin, S. Guo, and Q. Fu, "A vascular interventional surgical robotic system based on force-visual feedback," *IEEE Sensors Journal*, vol. 19, no. 23, pp. 11081-11089, 2019.

[20] H. Zhang, H. Luo, Z. Wang, Y. Liu, and Y. Liu, "Multi-robot cooperative task allocation with definite path-conflict-free handling," *IEEE Access*, vol. 7, pp. 138495-138511, 2019.

[21] L. Jin, Y. Qi, X. Luo, S. Li, and M. Shang, "Distributed competition of multi-robot coordination under variable and switching topologies," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 3575-3586, 2021.

[22] W. Liu, Z. Wang, X. Liu, N. Zeng, Y. Liu, and F. E. Alsaadi, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11-26, 2017.

[23] D. Bau, J.-Y. Zhu, H. Strobelt, A. Lapedriza, B. Zhou, and A. Torralba, "Understanding the role of individual units in a deep neural network," *Proceedings of the National Academy of Sciences*, vol. 117, no. 48, pp. 30071-30078, 2020.
[24] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1-15, 2018.
[25] Z. Majdisova and V. Skala, "Radial basis function approximations: comparison and applications," *Applied Mathematical Modelling*, vol. 51, pp. 728-743, 2017.
[26] B. Amiri, M. Fathian, and A. Maroosi, "Application of shuffled frog-leaping algorithm on clustering," *The International Journal of Advanced Manufacturing Technology*, vol. 45, pp. 199-209, 2009.
[27] Z. Fu, J. Pan, E. Spyrakos-Papastavridis, X. Chen, and M. Li, "A dual quaternion-based approach for coordinate calibration of dual robots in collaborative motion," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4086-4093, 2020.
[28] B. Benaissa, N. A. Hocine, S. Khatir, M. K. Riahi, and S. Mirjalili, "YUKI Algorithm and POD-RBF for Elastostatic and dynamic crack identification," *Journal of Computational Science*, vol. 55, 101451, 2021.
[29] Z. Chen, "Research on internet security situation awareness prediction technology based on improved RBF neural network algorithm," *Journal of Computational and Cognitive Engineering*, vol. 1, no. 3, pp. 103-108, 2022.
[30] M. Karpagam, K. Geetha, and C. Rajan, "A modified shuffled frog leaping algorithm for scientific workflow scheduling using clustering techniques," *Soft Computing*, vol. 24, no. 1, pp. 637-646, 2020.