A Novel Non-Collision Paths Planning Strategy For Multi-Manipulator Cooperate Manufacturing System

Chang Su (suchang@hust.edu.cn)
Huazhong University of Science and Technology - Main Campus: Huazhong University of Science and Technology

Xu Jianfeng
Huazhong University of Science and Technology - Main Campus: Huazhong University of Science and Technology

Research Article

Keywords: Multi-manipulator cooperate manufacturing system, PSO-BP neural network, Non-collision path planning, Sampling based Position Space Map Search

Posted Date: April 2nd, 2021

DOI: https://doi.org/10.21203/rs.3.rs-349956/v1

ORIGINAL ARTICLE

A novel non-collision paths planning strategy for multi-manipulator cooperate manufacturing system

Chang Su[1] · Jianfeng Xu[1]

Abstract
Analogy to the definition of Human-robot Interaction (HRI), the case of multiple manipulators with shared workspace, non-simultaneous manufacturing tasks and separate objects is named as multi-manipulator cooperation, which is becoming more widely employed in modern industrial manufacturing system and requiring non-collision path planning as a key issue in terms of safety and efficiency. In this paper, a novel method called Sampling based Position Space Map Search (SbPSMS) method which combines the map search method with the time-sampling based method will be proposed, including a minimum distance prediction method based on PSO-BP neural network for collision detection and two candidate position determination methods for search map establishing of all manipulators. After the specific search map simplification process, the local path fragments during each sampling time interval can be determined via cost function, which will be glued together to generate the final collision-free paths. The simulation results not only show that the PSO-BP hybrid algorithm has more accurate of nearly 2mm than the standard BP neural network in minimum distance prediction, but also demonstrated that our proposal can successfully achieve collision avoidance of dual manipulators system whilst meeting the real-time requirements for multi-manipulator co-operate assembling scenarios. The further satisfactory simulation results of triple manipulators suggesting our algorithm can be extended to applications of multiple manipulators cooperate manufacturing.

Keywords Multi-manipulator cooperate manufacturing system · PSO-BP neural network · Non-collision path planning · Sampling based Position Space Map Search

1 Introduction

At present, six degree-of-freedom (6-DOF) industrial manipulators play an increasingly important role in automated manufacturing, due to a number of advantages including the provision of tireless repetitive labour, faster moving speeds and higher accuracy performance [1]. Thus, tasks requiring multiple workers can undoubtedly be accomplished by multiple manipulators cooperation，which means that multiple manipulators respective manufacturing will not only work side by side, but as dyads and teams. Analogy to the definition of Human-robot Interaction (HRI) [2, 3], the case of dual or multiple manipulators manufacturing system can be divided into cooperation and collaboration according to the tasks arrangement (Fig.1). For the case of collaborate manufacturing system, all manipulators together with the executed mechanism constitute a complete multi-degree-of-freedom closed-loop or parallel manufacturing system. Thus, the non-collision manufacturing control strategy under this circumstance can be transformed into the currently matured internal obstacle avoidance strategy [4]. However, for the scenario of multiple manipulators cooperate manufacturing system (e.g. assembling as shows in the right figure of Fig.1), problems and limitations arise in small overlapping workspaces accommodating numerous cooperative manipulators, whereby collisions can occur if no related countermeasures are put in place. In this case, finding a reliable strategy of simultaneously collision-free paths planning for all the cooperative manipulators is a sensible choice to overcome the upper-mentioned limitations.

Fig.1 The schematic diagram of multi-manipulator cooperate and collaborate manufacturing system. Multi-manipulator collaborate manufacturing

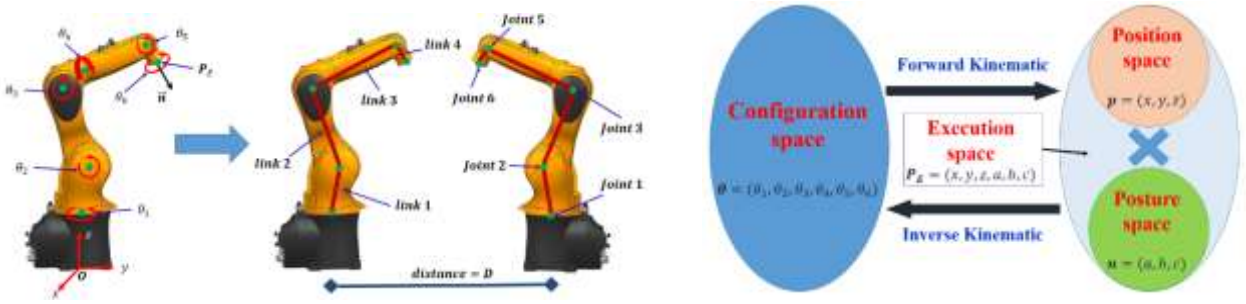
[1] Jianfeng Xu
jfxu@hust.edu.cn

[1] State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science & Technology, Wuhan 430074, China

system (Left): the case of multiple manipulators with shared workspace, simultaneous manufacturing tasks and common object; Multi-manipulator cooperate manufacturing system (Right): the case of multiple manipulators with shared workspace, non-simultaneous tasks and separate object;

1.1 Relative work

6-DOF industrial manipulator implies a serial chain robot with six revolving joints, which are structurally much more complex than mobile robot. For modelling of the 6-DOF industrial manipulator, a description of the position space, posture space, execution space, and configuration space are given as descripted in Fig.2.



(*a*) (*b*)

Fig. 2 Modelling of a 6-DOF industrial manipulator in (a): $\boldsymbol{Oxyz}$ is the base coordinate system of the manipulator; $\boldsymbol{n}$ is the direction vector of the central axis of the sixth joint; $\boldsymbol{P}_E$ is the position coordinate vector in $\boldsymbol{Oxyz}$. All position coordinates that the manipulator EEF can reach within the space set are gathered and designated as the **position space**. $\boldsymbol{n} = (a, b, c)$ is the direction vector of the central axis of the sixth joint, which can also be used to represent the pose of manipulator EEF. All possible direction vectors that manipulator EEF can reach are collected in a set, referred to as the **posture space**. The combination of the position space and posture space constitutes the **execution space**. The value of each joint can be denoted as $\theta_i (i = 1 \sim 6)$. Then $\boldsymbol{\theta} = (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6)$ can be recognized as a point in the 6-DOF configuration space (**C-space**). The relations between the execution space, C-space, position space, and posture space are shown in (b). Every point in the C-space can determine a only point in execution space via forward kinematic calculation, but every point in execution space can correspond to more than one point in the C-space for the inverse kinematic calculation of 6-DOF manipulator has multiple solutions.

Since single manipulator can be modelled as a particle in its own C-space, the particle-based path planning methods [5] can be used for single manipulator path-planning directly. However, considering the non-collision path planning of multi-manipulator cooperation scenarios, the main encumbrance that prevent us from directly applying particle-based path planning algorithms can be summarized in the following three points: (a) The dimensions of C-space are equivalent to the joint number of the manipulator. As a consequence, using particle-based path planning methods in high-dimensional space will greatly increase computation time and reduce the efficiency of the algorithm [6]. (b) The mapping relationship between C-space and execution space is nonlinear, which means the optimal path in C-space may not be optimal in execution space [7]. As the actual workspace of the manipulator is execution space, finding an optimal execution space path rather than C-space path is the final target of our exertion. (c) The cooperate manipulators are themselves regarded as obstacles to each other. Therefore, all manipulators must be projected into the same C-space. Due to the irregular structures of manipulators, this process is very complicated and time-consuming [8].

Limited progress has been made for non-collision path planning of multiple cooperating manipulators in C-space. Obstacles were modelled as simple geometries by Jia et al. [9], which will then be projected into the C-space of a manipulator. This method is not effective for multiple manipulators due to their complex structures. Following on from this, Yu et al. [10] divided the C-space into multiple layers based on certain dimensions, and implemented a rapidly created C-space grid map on each layer, however, due to the huge amount of elements in the maps, the whole algorithm tends to be time-consuming. Li et al. [11] planned collision-free paths for 2D horizontally articulated dual-arm robots by dividing the C-space of each robot into multiple blocks, marking the obstacle space and free space at different time intervals, and mapping the search method to seek out an optimal path in free space. The method was too computationally expensive for real-time applications, and moreover, it is difficult to project a 6-DOF manipulator body into the C-space of another 6-DOF manipulator. Harada et al. [12] proposed a general manipulation planner for a dual-arm industrial manipulator, which combined the C-space of the arms and obstacle space into one overall search map. The method mainly focused on the trajectory arrangement and movement order of each arm to allow a target object to pass from starting point to end point, however, the possibility of collision between the two working arms was not considered.

Hence, how to explore a simple, effective and reliable execution space path planning algorithm has become one of the more and more enthusiastic focuses in recent decades. An online collision-free trajectory generation algorithm for dual-arm robots was established by Lee et al. [13], by setting up a Virtual Road Map (VRM) in the execution space, and refreshing the map with a new collision-free path. However, the execution time is long due to difficulties in setting up the

VRM, and extending the method to multiple robots is challenging. Larsen et al. [14] divided the working area into different parts, and set up obstacle models within each part. A master-slave method was then used to choose a free path; however dynamic environments cannot be accommodated since the obstacle must be static. Cohen et al. [15] combined the C-space and execution space to build a "motion space", and used the Lazy Weighted A* method to plan a non-collision path in an environment containing N-manipulators, but only path planning for one robot is allowed at a time and the planning process is performed off-line. While the ARA* method was also tested to search for the optimal path in the workspace (execution space) of a manipulator, the study focused only on dual arms performing the same task and did not consider the coordination of different tasks.

Additional methods have been proposed for the non-collision path planning of multiple manipulators. Chiddarwar et al. [16] used the A* method to plan a collision-free path in C-space without considering the motion of other robots, and a Path Modification Sequences (PMS) method was then proposed to arrange the moving sequence of each robot, resulting in a much longer execution time. Another method proposed by Afaghani et al. [17] used a collision map for detecting the collisions between two robots in order to avoid deadlocks, which can occur if one robot becomes an obstacle to another. Unfortunately, the method simply delays the movement of the robot to avoid the collision. Rodríguez et al. [18] suggested an approach based on a variation of the Probabilistic Road Map, called the Probabilistic Road Map with Obstacles (PRMwO). The method does not exclude collision samples with removable objects, but instead classifies them as collided obstacle(s), and allows the search for free paths b highlighting which objects must be removed from the workspace to make a valid path. While this approach removes any obstacles along the path of the working manipulator, it does not enable the manipulator to actually bypass obstacles. Habibnejad et al. [19] used the Artificial Potential Field (APF) method to plan the paths of multiple cooperative manipulators on mobile bases, however the study focused solely on the path of the end effector (EEF) and did not consider the entire arm, thus lacking important systemic considerations.

A preliminary conclusion can be drawn from the above review that when it comes to the path planning of dual or multiple industrial manipulators, the negative influence, due to the above characteristics such as model complexity, high dimensionality and complex planning space, will become even more critical. Instantly multi-manipulator path planning algorithms can be divided into map search based method, time-sampling based method, mathematical model method and others [20]. Map search based method [21-23] is a global optimization algorithm which will cost huge time on planning map construction. While time-sampling based method [24-26] divide the whole planning period into several orderly but isolated intervals, and splice all the calculated outcomes of each interval into a complete result path or planning map in order.

1.2 Paper organization

This paper focus on the simultaneously non-collision path planning for the scenario of multi-manipulator cooperate manufacturing. Under this circumstance, a path planning algorithm that can skirt round the constraints of mechanical structure, simplify the planning space, and determine the paths of all manipulators participating in cooperation system in real time will be the kernel of this paper.

Thus, a novel method called Sampling based Position Space Map Search (SbPSMS) method which combines the map search method with the time-sampling based method will be proposed. Equivalently speaking, a local planning map in each time interval will be built and a local optimal path will be determined according to the local planning map, then all path fragments that be planned in chronological order are glued together to generate collision-free paths, which is very efficient at the cost of abandoning the global optimum.

The first step in building a planning map is modelling and collision detection. In order to further shorten the execution time of these two steps, we try to use neural network algorithm to merge these two steps into one step. Then, two different candidate position determination method as well as specific search map simplification process will be descripted in detail to set up the local planning map of all manipulators. After path planning based on map search algorithms, the remaining work is to repeat the same work during each sampling time interval until all manipulators reaches their respective targets.

In this paper, a brief path planning method review of multiple manipulator cooperation is shown in Section 1. Section 2 provides a detailed framework to describe how to predict the shortest distance between two industrial manipulators in real time based on the PSO-BP neural network. Section 3~5 are the detail content of SbPSMS method. In section 3, two candidate position points selection methods are determined for a single manipulator, providing the nodes of the search map for all the cooperative manipulators. In section 4, we describe the search map simplification method for dual and multiple manipulators. In section 5, formulae for the cost function are introduced as an evaluation index of the optimal path. Section 6 presents the simulation results and relevant analyses. Finally, in section 7, some conclusions are presented.

2 Minimum distance prediction based on PSO-BP neural network

2.1 Background

Usually, in order to determine the potential collision, a safety control strategy of collision detection by comparing the shortest distance between manipulators with threshold is ap-

plied. In the early stage, the minimum distance was calculated through the 3D geometric models and many efficient algorithms were well developed [27, 28]. But because of its computational cost, hierarchical bounding volume which employs geometrical primitives to approximate the objects is adopted to overcome the shortage in real application, including axis-aligned bounding box [29], object-oriented bounding box [30, 31], sphere-swept volume [32, 33], ellipsoid [34], sphere [35] and convex hull [36]. Basically, most bounding volumes are struggling to get a trade-off between the model accuracy and algorithm complexity, computation efficiency and the tightness of fitting the underlying object. Due to the rotation invariance, sphere is a popular choice of bounding volume. But the difficulties lie in the optimization of the radii and the number of the sphere in order to obtain a relative high accuracy. There are also other techniques that focus not only on the minimum distance between the objects, but also take the system state into consideration to evaluate risk index [34]. Through the methodology described above to approximate the objects, minimum distance is equivalent to calculate the shortest distance among geometrical primitives. But the algorithm calculating the shortest distance between two geometry objects is usually complex to implement and many different cases need to consider.

In the developed approach, to simplify the implementation of the algorithm, potential collision detection of the two manipulators is predicted by adopting PSO-BP neural network method. BP neural network is very powerful in nonlinear mapping and can approximates nonlinear function in any precision [37]. Besides it's popular in prediction, its drawback is easy to fall into local minimum. Nevertheless, PSO algorithm perform well in global optimization [38]. The improved PSO with two optimization stages integrating k-mean clustering to keep population variety and described in the subsequent section is applied before training BP neural network. With the optimized values as initial weights, the BP neural network can decrease the training time and generate a better training result of accuracy than the standard BP neural network with respect to given training and test data set, as investigated in [39, 40].

2.2 Modelling & Minimum distance calculation

The manipulators are composed of irregular and complex parts, which results in difficulties in computing the minimum distance. In this paper, sphere-swept volume is applied to approximate the linkages and joints in order to calculate the minimum distance between each two manipulators for the neural network training process, as shown in figure 3.

According to the modelling, the minimum distance calculation process is converted to determine the shortest distance between the surfaces of two capsule-like geometrical objects. Assume $\boldsymbol{S}_1, \boldsymbol{S}_2$ are the two points sets in Cartesian space,$\boldsymbol{P}_1(x_1, y_1, z_1) \in \boldsymbol{S}_1$,$\boldsymbol{P}_2(x_2, y_2, z_2) \in \boldsymbol{S}_2$, then the minimum distance between $\boldsymbol{S}_1, \boldsymbol{S}_2$ can be expressed as:

$$d_{min} = min\{\|\boldsymbol{P}_1 - \boldsymbol{P}_2\| : \boldsymbol{P}_1 \in \boldsymbol{S}_1, \boldsymbol{P}_2 \in \boldsymbol{S}_2\} \quad (1)$$

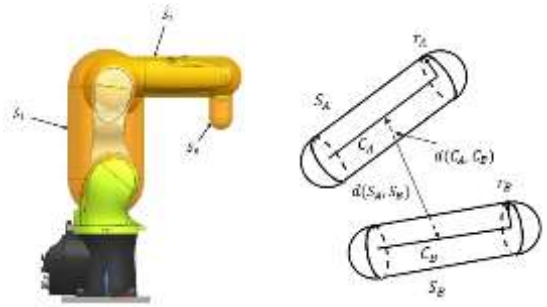


Fig. 3 Left: Manipulator model approximated by sphere-swept volume. Right: Minimum distance of two capsule models

In figure 3, define the surfaces of manipulator A and B by $\boldsymbol{S}_A^i \in R^3$, $\boldsymbol{S}_B^i \in R^3$ respectively; the center line segments of the cylinders and the center points of the spheres are defined by $\boldsymbol{C}_A^i \in R^3$, $\boldsymbol{C}_B^i \in R^3$; $\boldsymbol{r}_A^i, \boldsymbol{r}_B^i$ are the radiis of the hemispheres $(i \in (1,2,3,4))$.Thus, the minimum distance function can be summarized as [41]:

$$d_{min} = mi\,n\left(d\left(\boldsymbol{C}_A^i, \boldsymbol{C}_B^i\right) - \left(\boldsymbol{r}_A^i + \boldsymbol{r}_B^i\right)\right)(i \in 1,2,3,4) \quad (2)$$

This fact indicates that the complex calculation of two cylinder surfaces is replaced by the calculation of minimum distance of the two centre line segments of the cylinders and then subtracting the radii of hemispheres A and B [41]. Meanwhile, in order to determine the position and orientation of line segment, forward kinematic [42] is applied which is complex and time consuming. This paper try to propose an algorithm constructed by PSO-BP neural network [43] to predict the minimum distance with respect to the given joint values of all cooperated manipulators.

2.3 Detail of PSO-BP neural network

In this section, a hybrid method of PSO and BP neural network is developed to predict the shortest distance in a given position and posture. In the presented hybrid method, the PSO algorithm is utilized to optimize the initial connection weights of neurons and threshold values of network. With the introduction of an extra learning item evaluated by the fitness-distance-ratio [44] and k-mean clustering, the PSO can avoid premature. In this case, the hybrid algorithm combining the PSO and BP neural network has higher generalization and accuracy [39, 40].

1) PSO algorithm

PSO is a global optimization algorithm based on swarm intelligence and evolutionary computation, which searches the global optimization solution via the competition and learning among all particles. The original PSO formulas proposed by Kennedy and Eberhart [45] were modified [46] with inertial parameter which was empirically improve the overall performance.

The core idea of PSO is define every particle in the swarm with size of m as a potential solution problem in a d-dimensional space, with the i-th particle represented as$X_i =$

$(x_{i1}, x_{i2}, x_{i3}, \dots, x_{id})$. Each particle has two important properties of position and velocity, it preserves its previous best position $(pbest)$ and velocity $(vbest)$ along each dimension, represented as:

$$\begin{aligned} P_i &= (p_{i1}, p_{i2}, p_{i3}, \dots, p_{id}) \\ V_i &= (v_{i1}, v_{i2}, v_{i3}, \dots, v_{id}) \end{aligned} \tag{3}$$

In each generation, the $pbest$ of the particles with the best fitness named as *gbest* which can be denoted as $P_g = (p_{g1}, p_{g2}, p_{g3}, \dots, p_{gd})$, together with the *pbest* of the current particle, are used to adjust the velocities of each dimension. The updated velocities are then used to generate a new position for the particle. In mathematics, the position and velocity of each particle are computed by the following update formulae:

$$\begin{cases} v_{ij}^{k+1} = \omega v_{ij}^{k} + c_1 r_1 \left(p_{ij}^{k} - x_{ij}^{k}\right) + c_2 r_2 \left(p_{gj}^{k} - x_{ij}^{k}\right) \\ x_{ij}^{k+1} = x_{ij}^{k} + v_{ij}^{k+1} \end{cases} \tag{1}$$

where i is the serial number of the particle; j is the jth dimension of the particle; $\omega = \omega_{max} - \frac{\omega_{max} - \omega_{min}}{K_p} \times k$ is the inertial weight and dynamically changes with respect to iteration; $k = 1,2,3, \dots, K_p$(K_p is the maximum iteration of PSO) is the current iteration of the algorithm; c_1, c_2 are learning factors or acceleration coefficients determining the relative influence of the cognitive and social components, which are usually equal to 2; $r_1, r_2 \in [0,1]$ are random numbers.

2) Fitness-distance-ratio

In order to avoid the premature convergence observed in many applications of PSO, Peram T, Veeramachaneni K and Mohan C K [44] proposed FDR-PSO in which the particle moves towards nearby particles with higher fitness. By using the ratio of the fitness difference to the one-dimension distance, the velocity of the ith particle in nth dimension is also updated via a nearby best particle with the highest ratio, denoted as P_{in}:

$$\mathrm{P}_{in} = \max \frac{fitness(P_j) - fitness(X_i)}{|P_{jn} - X_{in}|} \tag{2}$$

In this case, the particle's velocity is influenced by three particles: previous best particle, global best particle and the nearby best particle. Considering the item of nearby particle, the formulae (4) can be changed into:

$$\begin{cases} v_{ij}^{k+1} = \omega v_{ij}^{k} + c_1 r_1 \left(p_{ij}^{k} - x_{ij}^{k}\right) + \omega v_{ij}^{k} + c_1 r_1 \\ \left(p_{ij}^{k} - x_{ij}^{k}\right) + c_2 r_2 \left(p_{gj}^{k} - x_{ij}^{k}\right) + c_3 r_3 \left(p_{nij}^{k} - x_{in}^{k}\right) \\ x_{in}^{k+1} = x_{in}^{k} + v_{in}^{k+1} \end{cases} \tag{3}$$

where c_3 is also a constant quantity representing the relative influence of the nearby best particle; $r_3 \in [0,1]$ is a random number.

3) K-mean clustering

K-mean clustering algorithm is an unsupervised learning algorithm that intends to divide the particles into k subgroups with the shortest Euclidean distance to the clustering centroids. K-mean clustering method is introduced in order to improve the globalization optimization and population diversity of PSO algorithm. This algorithm aims to minimize the cost function with respect to the cluster centroids using following formulas:

$$cost = \frac{1}{m} \sum_{i=1}^{m} \|X_i - u_k\|^2 \tag{4}$$

$$u_k = \frac{1}{m_k} \sum_{i=1}^{m_k} X_i \tag{5}$$

where $\|\dots\|$ denotes the Euclidean distance and m is the number of the particles; u_k represents the cluster centroid of the kth subgroup; m_k is the number of the particles in the kth group.

4) BP neural network

BP neural network is an artificial neural network that simulates the cerebrum information processing. BP neural network with hidden layer can approximate linear or nonlinear function with any precision. As the consequence of its powerful ability of nonlinear mapping, it has been widely used in pattern recognition, image processing, automatic control and some other field.

The basic structure of BP neural network is input layer, hidden layer and output layer. Input layer accepts the training data set, namely 12 joint values in this paper; hidden layer maps the input data to the output layer through connection weights and activation functions; output layer linear-combine the outputs of hidden layer to computes the predicted result. The model of BP neural network with l layers and one output is shown in figure 4.

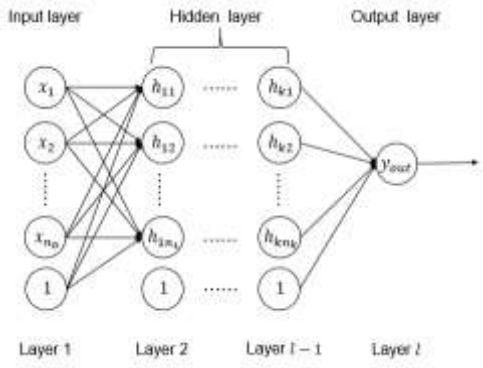


Fig.4 Model of BP neural network

For a neural network with one output, the mean square error of all training sample is used to evaluates the performance of neural network and can be expressed in formula (9).

$$E = \frac{1}{2m} \sum_{i=1}^{m} \left(y_{out,i} - y_i\right)^2 \tag{6}$$

In this learning algorithm, the training process of the network terminates when the network error E is less than the given tolerance or reach the maximum epoch. Back propagation (BP) algorithm and gradient descent are combined to update the connection weights of neuron throughout the whole training process. In the output layer, the error can be denoted as:

$$\delta^L = a^L - y \tag{7}$$

where δ^L is the error of the output layer; $a^L$is the output of the network and y is the training data. When the error backs propagation into the hidden layers, it needs to take the connection weights into consideration as shown in equation (11).

$$\delta^l = (\Theta^{l+1})^T \delta^{l+1} \circ g'_{(z^l)} \tag{8}$$

In the formula, δ^l refers to the error of the lth hidden layer($l = 2, \dots, L-1$); Θ^{l+1} is the weight matrix of the $(l+1)th$ layer; z^l is the input of lth hidden layer; $g'_{(z^l)}$ is the derivative of activation function with respect to z^l and $\circ$ means Hadamard product of matrix. With the gradient descent, the weights of neural network can be updated using the following formulas:

$$\partial E / \partial \theta^l = a_j^{l-1} (\delta^l)^T \tag{9}$$

$$\theta_{k+1}^l = \theta_k^l - \eta (\partial E / \partial \theta^l) \tag{10}$$

where η is the learning rate which has significant influence on the convergence speed and the accuracy of the network. But usually it is difficult to assign a specific value from the beginning in real application. Larger numbers will result in oscillations in the minimum value, while a small number will cause a long training time. Using adaptive learning rate [47] with respect to the error difference is a feasible solution of making a balance between these two aspects:

$$\eta(t+1) = \begin{cases} 0.75\eta(t), & E(t) > 1.04E(t-1) \\ 1.05\eta(t), & E(t) < E(t-1) \\ \eta(t), & other \end{cases} \tag{11}$$

5) PSO-BP neural network

Standard BP neural network uses gradient descent for training which indicates that the training result and accuracy are sensitive to the initial weight values and threshold. It is a feasible way to introduce PSO into BP neural network. The basic idea is to use the PSO algorithm to optimize the weight values, and then the optimized solution of the PSO is transferred to BP neural network as its initial weight. A PSO algorithm containing two stages is utilized to optimize the initial weights. In the first stage, all the particles that are divided into k subgroups with k-mean clustering search the global optimization together until the maximum iteration; in the second stage, PSO algorithm works on a new population that is composed of k global best particle of subgroups. After these two steps, the searching space of BP neural network narrows down and the hybrid method has better performance in prediction than standard BP neural network. The details are shown in the Fig.5.

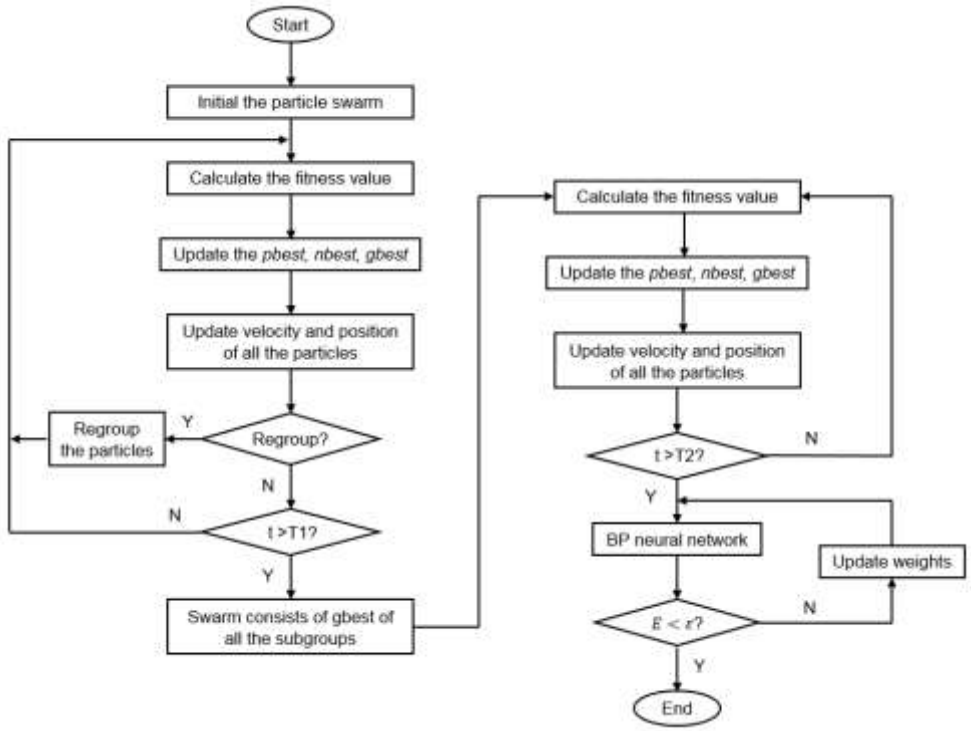


Fig.5 Flowchart of PSO-BP neural network for collision detection

3 Candidate position determination for single manipulator

The candidate positions for a single manipulator consist of all possible positions of the EEF that the manipulator can reach within the position space during the next scanning period. Determining these candidate positions is the cornerstone of proposal method, and the key to building a simple and effective position space search map (PSSM).

During the movement process of an industrial manipulator, the absolute value of EEF speed $\boldsymbol{v}_t$ is generally held uniform throughout the entire movement period to maintain stability of the system, which means the step length of the ma-

nipulator EEF L_t^{step} in every scanning interval must be constant. Therefore, the set of all candidate positions of a single manipulator constitutes a spherical surface, in which the current space position point of the EEF, $\boldsymbol{P}_t^{EEF}$, is the sphere center and L_t^{step} is the radius, and we call this set as Search Feasible Region (SFR). Theoretically, there can be an infinite number of candidate position points within SFR, however infinite continuous points cannot be used to build a node map, thus selecting a specific number of points is necessary. Here, we propose finding three non-coplanar vectors as the reference direction vectors and determining all candidate positions of a single manipulator, referring to the method of establishing a "unit cell" in chemistry. In this section, two separate methods are proposed to find the three reference direction vectors: the first is based on the base coordinate system of each manipulator, and the second considers the current velocity of each manipulator EEF.

However, before talking about how to determine the three reference direction vectors, the concepts of main movement axis, secondary movement axis, displacement axis, and main movement plane are proposed.

Definition: Assuming that a point moves from start position $\boldsymbol{P_{start}}$ to endpoint $\boldsymbol{P_{end}}$ under a coordinate system $\boldsymbol{Oxyz}$ as shown in Fig. 4, and the coordinate system $\boldsymbol{Oxyz}$ can be arbitrary, the vector from $\boldsymbol{P_{start}}$ to $\boldsymbol{P_{end}}$ is $\boldsymbol{P_{disp}} = (x_{disp}, y_{disp}, z_{disp})$, which indicates the size and direction of the movement. Displacements in the positive $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{z}$ directions are x_{disp}, y_{disp}, and z_{disp}. The values $\|x_{disp}\|$, $\|y_{disp}\|$, and $\|z_{disp}\|$ can be compared and sorted in a descending order, such that the axis corresponding to the minimum value is set as the displacement axis. Moreover, the main movement plane is defined as the plane composed of the other two axes, whereby the main axis of the plane is the axis corresponding to the maximum number, and the remaining axis is the secondary movement axis.

As an example scenario in Fig.6, if $\|x_{disp}\| \geq \|y_{disp}\| \geq \|z_{disp}\|$ then the $\boldsymbol{x}$-axis is the main movement axis, the $\boldsymbol{y}$-axis is the secondary movement axis, the $\boldsymbol{z}$-axis is the displacement axis, and the $\boldsymbol{xoy}$-plane is the main movement plane.

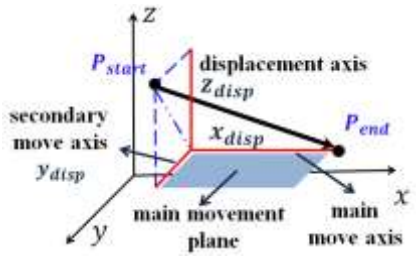


Fig. 6 Diagram of displacement axis, secondary movement axis, main movement axis, and main movement plane of a moving path from $\boldsymbol{P_{start}}$ to $\boldsymbol{P_{end}}$ under $\boldsymbol{oxyz}$ coorsinate system.

3.1 Coordinate system based method

The principle of coordinate system based method is to project the base coordinate system of the single manipulator onto the global coordinate system to establish the three reasonable reference direction vectors by using the motion attribute of the manipulator while moving along a given path in Cartesian space. In a multi-manipulator system, the motion of every single manipulator can be calibrated conveniently by each manipulator's own base coordinate system, but the relative motion among multiple manipulators can only be calibrated under the global coordinate system. Besides, when applying the PSMS method on the non-collision path planning of multi-manipulator system, we need to combine the motion of every single manipulator and that of all the manipulators together to judge how to reduce the number of nodes in the PSSM in order to increase the efficiency of the algorithm.

The common base coordinate system for a single manipulator is shown as Fig.7. When the manipulator is in its initial position (which means that all the values of six joints are zero), the $\boldsymbol{x-y}$ plane of the base coordinate system is the bottom surface of the manipulator base, the origin point of base coordinates system is the center point of the bottom surface of the base. From the origin point, the $\boldsymbol{z}$ axis of the base coordinates system is the axis perpendicular to the bottom surface of the base and point to the direction of the manipulator mechanical arm, the $\boldsymbol{y}$ axis is the axis in the $\boldsymbol{x-y}$ plane and point to the position of EEF, and then the $\boldsymbol{x}$ axis can be determined by the right-hand rule. Then the base coordinate system of the manipulator $\boldsymbol{Oxyz}$ is built up. $\boldsymbol{O_g x_g y_g z_g}$ is the global coordinate system.

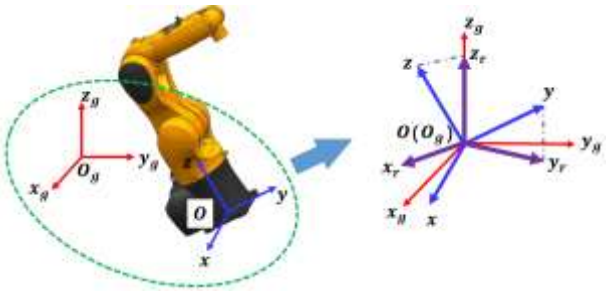


Fig. 7 Diagram of building up the three reference direction vectors via coordinate system base method. $\boldsymbol{O_g x_g y_g z_g}$ is the global coordinate system as well as $\boldsymbol{Oxyz}$ is the base coordinate system. The posture of base coordinate system is arbitrary. Take the scenario of Fig.4 for example, the reference direction vector $\boldsymbol{z_r}$ is the projection vector of $\boldsymbol{z}$ axis on the displacement axis $\boldsymbol{z_g}$, $\boldsymbol{y_r}$ is the projection vector of $\boldsymbol{y}$ axis on the main movement plane $\boldsymbol{O_g x_g y_g}$, and $\boldsymbol{x_r}$ **is** close to $\boldsymbol{x}$ axis and perpendicular to both the reference direction vector $\boldsymbol{z_r}$ and $\boldsymbol{y_r}$.

The execution steps of the coordinate system based method are shown as follows:

Step 1: determine the main movement axis, secondary movement axis, displacement axis, and main movement plane of the manipulator under the global coordinate system;

Step 2: move the origin point of the global coordinate system to the origin point of the base system coordinate system. Since the base coordinate system can be arbitrary, the base coordinate system and the global coordinate system may not coincide;

Step 3: project the $\boldsymbol{z}$ axis of the base coordinate system on the displacement axis to be the reference direction vectors $\boldsymbol{z_r}$, project $\boldsymbol{y}$ axis of the base coordinate system onto the main movement plane to be the reference direction vectors $\boldsymbol{y_r}$, then the reference direction vectors $\boldsymbol{x_r}$ will be determined via the following formula:

$$\boldsymbol{x_r} = \boldsymbol{y_r} \times (\boldsymbol{x} \times \boldsymbol{y_r}) \tag{15}$$

which means the reference direction vector $\boldsymbol{x_r}$ is close to the $\boldsymbol{x}$ axis and perpendicular to both the $\boldsymbol{z_r}$ and $\boldsymbol{y_r}$.

However, in some cases like the $\boldsymbol{z}$ axis of the base coordinate system will in the main movement plane which means the $\boldsymbol{z}$ axis is perpendicular to the displacement axis, or the $\boldsymbol{y}$ axis coincides with the displacement axis which means the projection of $\boldsymbol{y}$ axis on the main movement plane is the origin point itself, the aforementioned process will lose efficacy. In these cases, we only need to interchange the $\boldsymbol{y}$ axis and the $\boldsymbol{z}$ axis of the base coordinate system, and then the three reference direction vectors can be got according to the processes aforementioned.

It is obvious that these three reference direction vectors ($\boldsymbol{x_r}$,$\boldsymbol{y_r}$, $\boldsymbol{z_r}$) are perpendicular to each other as the base coordinate system and global coordinate system do.

3.2 Velocity based method

According to particle kinematics and dynamics, when the manipulator EEF passes point $\boldsymbol{P}$ at time instant t with speed $\boldsymbol{v}$, it will move at a speed which is as close as possible to $\boldsymbol{v}$ in the next time instant according to the principle of minimum energy. When the current velocity of the EEF is not parallel to any of the three axes of the manipulator base coordinate system, the direction of the velocity vector, as determined by the base coordinate method, is not necessarily consistent with the current velocity direction. Therefore, we propose a new method of three reference direction vectors determination based on the current velocity.

Firstly, a new coordinate system called velocity coordinate system $\boldsymbol{O_c xyz}$ will be set up to calibrate the motion of every single manipulator, in which $\boldsymbol{O_c}$ is the current position point of manipulator EEF. As shown on the right half side of Fig. 6, the velocity of the manipulator EEF at point $\boldsymbol{P}$ is $\boldsymbol{v_A}$ at the current time instant. Thus, the direction of vector $\boldsymbol{v_A}$ can be taken as the direction of the reference $\boldsymbol{x}$-axis. Then, the EEF position $\boldsymbol{P}$ and the base of the manipulator $\boldsymbol{O}$ can be connected to construct the space vector $\overrightarrow{\boldsymbol{PO}}$, which is projected onto the bottom surface of the manipulator base in the same coordinate system and serves as our reference $\boldsymbol{y}$-axis. Finally, the $\boldsymbol{z}$ -axis of the candidate position points can be calculated from the $\boldsymbol{x}$- and $\boldsymbol{y}$-axis according to:

$$\boldsymbol{z} = \boldsymbol{x} \times \boldsymbol{y} \tag{16}$$

Then how to determine the three reference direction vectors are the same as those presented in Section 4.1.

After setting up the three reference direction vectors, the determination of candidate positions for a single manipulator will be an easy job. As shown in Fig. 8, we assume the coordinate of the center point (black point) of the EEF at time t is $\boldsymbol{P} = (x_t, y_t, z_t)$, and $\Delta\boldsymbol{P} = (\Delta x, \Delta y, \Delta z)$ is the step gain required for the EEF to reach the target position along the three reference direction vectors, where $\Delta x = \Delta y = \Delta z$. That is to say, if the $\boldsymbol{x}$ axis coordinate of point $\boldsymbol{P}$ at time instant t is x_t, a possible coordinate for x_{t+1} must be one of the following: $x_t - \Delta x$, x_t, and $x_t + \Delta x$. The same can be applied for y_{t+1} and z_{t+1}.

According to the assumptions presented above, 27 different coordinate combinations can be obtained using the following formula:

$$\begin{aligned} \boldsymbol{P_{cad}^t} &= (x_{t+1}, y_{t+1}, z_{t+1}) \\ &= \boldsymbol{\Theta}\left\{\begin{pmatrix} x_t - \Delta x \\ x_t \\ x_t + \Delta x \end{pmatrix} \times \begin{pmatrix} y_t - \Delta y \\ y_t \\ y_t + \Delta y \end{pmatrix} \times \begin{pmatrix} z_t - \Delta z \\ z_t \\ z_t + \Delta z \end{pmatrix}\right\} \end{aligned} \tag{17}$$

where $\boldsymbol{\Theta}$ is the set flag and the operator "×" denotes "combination".

These coordinate combinations can set up as 27 points (marked $\boldsymbol{P_1}$~$\boldsymbol{P_{27}}$; yellow points in Fig. 8) in Cartesian space, and form a cube with side length $2\Delta x$ about point P. The 27 points are referred to as the candidate positions of point P. However, inconsistencies exist between the displacement values from point P to the 27 candidate positions, which results in difficulties in setting up the cost function.

In order to make every displacement fixed, an inscribed sphere of radius Δx is placed into each cube, as shown in Fig. 8. From point P to $\boldsymbol{P_1}$~$\boldsymbol{P_{27}}$, there will be 27 radials intersecting the inscribed sphere resulting in 27 intersection points marked as $\boldsymbol{P'_1}$~$\boldsymbol{P'_{27}}$ (green points in Fig. 8). The intersection points are the candidate positions with constant displacement in Cartesian space for a single manipulator during the next scanning period.

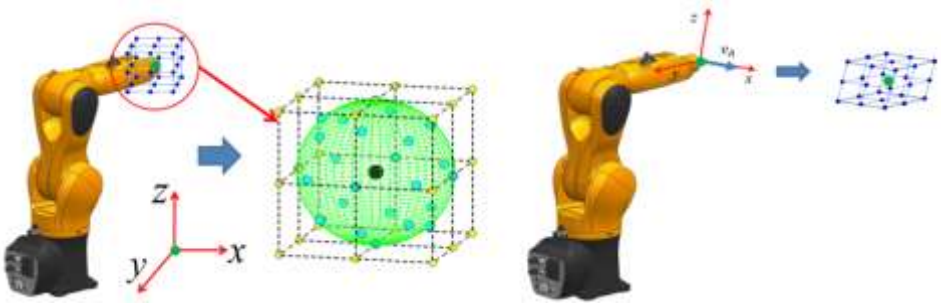


Fig. 8 Candidate positions of single manipulator EEF in Cartesian space. The left hand side is a diagram of candidate positions as determined by coordinate system based method; and the right hand side are the candidate positions determined using the velocity based method.

After the process above, 27 choices are available for one 6-DOF industrial manipulator, and can be uniformly marked in matrix $\boldsymbol{P_{cad}^{t}}$. The 27 candidate positions ($\boldsymbol{P_{cad}^{t}}$) also constitute the PSSM of a single manipulator at time t.

4 Search map simplification process

For multi-manipulator system, after setting up the "unit cell" structure of a single manipulator, the next step is to set up a reasonable PSSM for the entire system. Since the number of nodes in the "unit cell" can be reduced according to the physical and kinematic characteristics of the multi-manipulator system, a system of dual manipulators and one with no less than three manipulators will have different physical space position features. Thus, the search map simplification (SMS) of these two systems must be described separately.

4.1 SMS for dual-manipulator system

The two manipulators are marked as A and B in Fig. 9. The global coordinate system of the dual-manipulator is $\boldsymbol{T} = (x, y, z)$, while $\boldsymbol{T_A} = (x_A, y_A, z_A)$ and $\boldsymbol{T_B} = (x_B, y_B, z_B)$ are the base coordinate systems of manipulators A and B, respectively. Since the two manipulators are mounted on the same plane, we can define the $\boldsymbol{z}$-axis of each base coordinate system in the same direction as the $\boldsymbol{z}$-axis of the global coordinate system.

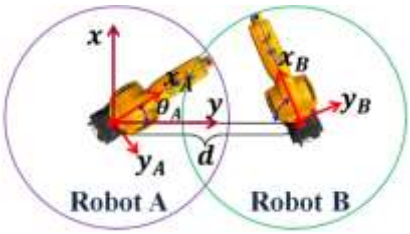


Fig. 9 Diagram of universal dual-manipulator cooperation situation

As mentioned in Section 4, the candidate positions of manipulators A and B can be expressed as $\boldsymbol{P_{Acad}^{t}}$ and $\boldsymbol{P_{Bcad}^{t}}$, with each containing 27 elements in their own three reference direction vectors. Every element represents a position point under global coordinate system, $\boldsymbol{P_{Acad}^{t}}$ can be set as the horizontal coordinate and $\boldsymbol{P_{Bcad}^{t}}$ as the vertical coordinate, in order to compose a PSSM of 27×27 nodes. Finding an optimal node in this map can provide an optimal solution for the dual-manipulator cooperation in the next step. If the time variable is taken into consideration, the time dimension can be added into the PSSM to produce a time-position space map (TPSM). Thus, planning of a collision-free path can be converted into finding an optimal path in the TPSM (Fig.10).

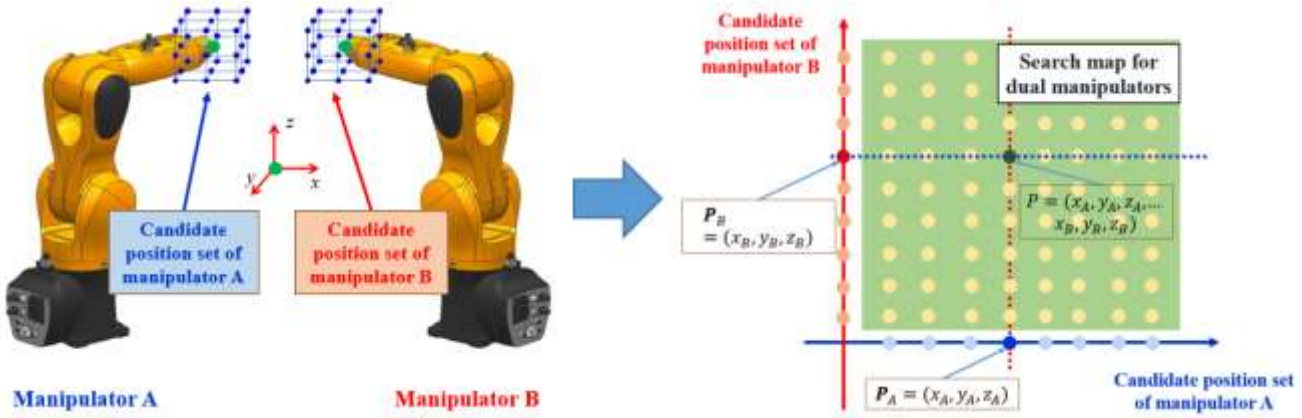


Fig.10 Relation diagram between set of candidate positions and PSSM of dual-manipulator system. The left hand side is the diagram of candidate positions of the two manipulators, and the right hand side is the diagram of PSSM for dual manipulators. In PSSM, the candidate positions of manipulators A constitute the discrete points on horizontal coordinate and those of manipulators B constitute the discrete points on the vertical coordinate.

However, a PSSM with 27×27 nodes is not efficient enough to meet real-time requirements. Hence, additional measures must be taken to reduce the number of nodes. One possible method is to reduce the number of candidate positions of a single manipulator.

Every element of the candidate positions for a single manipulator consists of 3 axis coordinates according to their own reference direction vectors, in which ($\boldsymbol{x_r}$, $\boldsymbol{y_r}$, $\boldsymbol{z_r}$) represents three non-coplanar reference direction vectors.

The above example can be used to demonstrate how to reduce the elements of the PSSM. The search choices can be reduced along the $\boldsymbol{x_r}$-, $\boldsymbol{y_r}$-, and $\boldsymbol{z_r}$-axis based on the following movement analyses:

a) Along $\boldsymbol{x_r}$-axis

Based on the previous assumption, the coordinate value of point $\boldsymbol{P}$ along the $\boldsymbol{x_r}$-axis at time t is x_t and the physical meaning of three possible values of x_{t+1} ($x_t - \Delta x$, x_t and $x_t + \Delta x$) represent three choices: "step backward towards the starting point", "remain stationary " and "step forward towards the target". Since the "step backward to the starting point" option does not promote efficient movement of a manipulator, this choice can be abandoned. Thus, movement along main movement axis can be reduced from 3 choices to

2.

b) Along $\boldsymbol{y_r}$-axis

The movement in the main movement plane of manipulator A (Fig. 6) is shown in Fig.11.

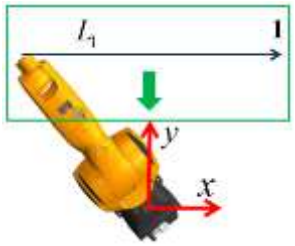


Fig. 11 Movement in the main movement plane of manipulator A

"Movement towards the manipulator base" (green arrow) and "remain stationary" options are the only two reasonable avoidance movements along $\boldsymbol{y_r}$-axis, since movement towards the base of the other manipulator would increase the possibility of collision. In the example scenario, manipulator A abandons moving along positive $\boldsymbol{y_r}$-axis, while manipulator B abandons along the negative $\boldsymbol{y_r}$-axis. In a similar way, the choice of movement along the secondary movement axis can also be reduced from 3 to 2 choices.

c) Along $\boldsymbol{z_r}$-axis

In the example scenario, the $\boldsymbol{z_r}$-axes of dual manipulators are the same and both the same with the $\boldsymbol{z_g}$-axis of global coordinate system, the coordinate values at time t for the two manipulators under the global coordinate system are z_A^t and z_B^t, respectively. Two boundary values ε and $-\varepsilon$ are set, where $\varepsilon > 0$. If $z_A^t - z_B^t > \varepsilon$, the reasonable movements of manipulators A and B along their respective $\boldsymbol{z_g}$-axes are in the positive (towards the target point) and negative (away from the target point) directions, respectively. If $z_A^t - z_B^t < -\varepsilon$, the reasonable movements of manipulators A and B along their respective $\boldsymbol{z_g}$-axes are in the negative and positive directions, respectively. If $\|z_A^t - z_B^t\| \leq \varepsilon$, manipulators A and B only need to move in the main movement plane of each manipulator.

In summary, movement choices along the displacement axis can be reduced from 3 to either 2 or 1, depending on the actual position coordinates.

The results of the choice reduction for manipulators A and B of the example scenario are summarized in Table 1.

Table 1 Results of choice reduction compared to original number of choices for dual manipulators*

			Manipulator A	Manipulator B	Nodes of PSSM
Before reduction	$\boldsymbol{x_r}$**-axis**		**3**: $(x_A^t - \Delta x,\ x_A^t,\ x_A^t + \Delta x)$	**3**: $(x_B^t + \Delta x,\ x_B^t,\ x_B^t - \Delta x)$	27×27
	$\boldsymbol{y_r}$**-axis**		**3**: $(y_A^t - \Delta y,\ y_A^t,\ y_A^t + \Delta y)$	**3:** $(y_B^t + \Delta y,\ y_B^t,\ y_B^t - \Delta y)$	
	$\boldsymbol{z_r}$**-axis**		**3**: $(z_A^t - \Delta z,\ z_A^t,\ z_A^t + \Delta z)$	**3** :$(z_B^t + \Delta z,\ z_B^t,\ z_B^t - \Delta z)$	
After reduction	$\boldsymbol{x_r}$**-axis**		**2**:$(x_A^t - \Delta x,\ x_A^t)$ or $(x_A^t,\ x_A^t + \Delta x)$	**2**:$(x_B^t + \Delta x,\ x_B^t)$ or $(x_B^t,\ x_B^t - \Delta x)$	7×7 or 3×3
	$\boldsymbol{y_r}$**-axis**		**2**: $(y_A^t - \Delta y$ or $y_A^t)$	**2**:$(y_B^t + \Delta y$ or $y_B^t)$	
	$\boldsymbol{z_r}$**-axis**	$\boldsymbol{z_A^t > z_B^t}$	**2**: $(z_A^t + \Delta z$ or $z_A^t)$	**2**: $(z_B^t - \Delta z$ or $z_B^t)$	
		$\boldsymbol{z_A^t < z_B^t}$	**2**: $(z_A^t - \Delta z$ or $z_A^t)$	**2**: $(z_B^t + \Delta z$ or $z_B^t)$	
		$\boldsymbol{z_A^t = z_B^t}$	**1**: $(z_A^t\ )$	1: $(\ z_B^t)$	

** "–" in the table means moving away from the target point; "+" means moving towards the target point. Here the movement choice of stationary on all the three reference direction vectors can be abandoned in the reduction process.*

However, if the $\boldsymbol{z_r}$-axes of dual manipulators are both along the a same axis but the axis is not the $\boldsymbol{z_g}$-axis of global coordinate system, then we take the along-axis of the global coordinate system as a new "$\boldsymbol{z_g}$-axis" and do as step c; if the $\boldsymbol{z_r}$-axes of dual manipulators are not along the a same axis, then manipulator A and B only need to move in the main movement plane of each manipulator.

The above analysis shows that the number of elements in the PSSM can be reduced from 27×27 to 7×7, or in some cases 3×3.

4.2 SMS for multiple manipulators

According to the analysis in Section 5.1, additional processing procedures should be performed to simplify the PSSM in order to meet the requirements for real-time calculation. In this section, the PSMS method is applied to the cooperation of multiple manipulators in the same way it can be applied to dual manipulator system.

N manipulators are marked as A, B, …, N. The transformation relationships of the position coordinates for each coordinate system are shown in Fig. 12.

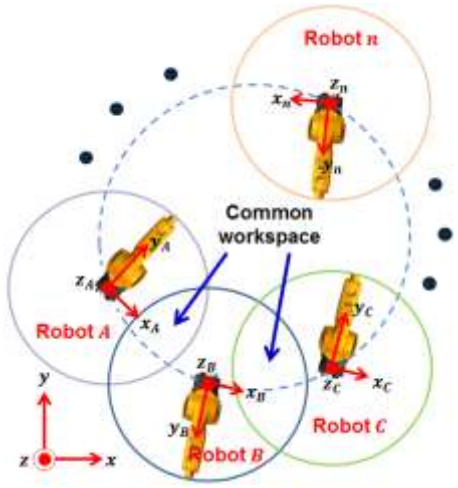


Fig. 12 Schematic of position and coordination transformation relationships for cooperation of multiple manipulators. Note that the base coordinate of each manipulator can be arbitrary, and we do not require that all the workspaces of the manipulators overlapped

In the schematic, $\boldsymbol{T_A} = (x_A, y_A, z_A)$,…,$\boldsymbol{T_N} = (x_n, y_n, z_n)$ are the base coordinate systems of manipulators A to N, individually, and $\boldsymbol{T_g} = (x_g, y_g, z_g)$ is the global coordinate system. Moreover, $\boldsymbol{T_{Ag}}$ is the transformation matrix of coordinate system $\boldsymbol{T_A}$ in relation to the global coordinate system $\boldsymbol{T_g}$, and similarly, $\boldsymbol{T_{Bg}}$ to $\boldsymbol{T_{Ng}}$. According to Section 4, one manipulator has 27 candidate position points. Hence, for N manipulators, the whole system produces a PSSM with 27^N nodes, which undoubtedly hinders practical application.

If the coordinates for the starting points $\boldsymbol{P_A^{start}}, \boldsymbol{P_B^{start}}, \ldots, \boldsymbol{P_N^{start}}$ and target points $\boldsymbol{P_A^{targ}}, \boldsymbol{P_B^{targ}}, \ldots, \boldsymbol{P_N^{targ}}$ of each manipulator under the global coordinate system are known, the main movement axis, secondary movement axis, and displacement axis of each manipulator can be identified. Thus, the number of elements in the PSSM can be reduced following the same logic used in Section 4.1.

Upon obtaining the coordinates for the starting points and target points of all manipulators, the displacement axes of all robots can be determined, and all manipulators with the same displacement axis can be collected as a set. For example, all manipulators with the same $\boldsymbol{z_g}$ displacement axis are collected as the set of "$\boldsymbol{z_g}$ displacement axis manipulators" (whose total number is i), and similarly, the set of "$\boldsymbol{x_g}$ displacement axis manipulators" (whose total number is j) and "$\boldsymbol{y_g}$ displacement axis manipulators" (whose total number is $(N - i - j)$) can be collected.

For each manipulator set, the candidate positions of each manipulator are defined following a certain set of rules:

(a) Along the $\boldsymbol{x_r}$-axis there are two candidate positions ("remain stationary" or "move one step towards its target").

(b) Along the $\boldsymbol{y_r}$-axis there are two candidate positions ("remain stationary" or "move one step towards its base").

(c) Along the $\boldsymbol{z_r}$-axis is slightly more complex and must be classified based on the real-time situation. All the $\boldsymbol{z_r}$-axis coordinate values under the global coordinate system are sorted in descending order and be divided into three subsets, the "positive set", "negative set", and "zero set". Two boundary values are set, ε and $-\varepsilon$ where $\varepsilon > 0$, such that values greater than ε are grouped into the "positive set", values lower than $-\varepsilon$ are gathered to form the "negative set", and the remaining values constitute the "zero set". Then, every manipulator in the "positive set" has two candidate positions, remain stationary or move one step towards the positive direction; every manipulator in the "zero set" has one candidate position, remain stationary; and every manipulator in the "negative set" has two candidate positions, remain stationary or move one step in the negative direction.

Since remaining stationary along all the three reference direction vectors is not desirable in terms of maximizing manipulator movement efficiency, the nodes represented by remain stationary along all three reference direction vectors are removed. The above process can thus reduce the number of elements from 27^N to 7^N, or in some cases even 3^N.

For manipulators in different sets, reductions in the number of candidate positions are independent. However, the above rules for element reduction may not be applicable under special conditions, for instance when one manipulator moves to its workspace boundary. In this situation, it is necessary to expand its number of candidate positions from 3 or 7 to 27 in order to find an optimal result for every manipulator during its next movement.

Next, we display the reduction results of the number of PSSM nodes for the cooperation of three manipulators. Assuming all $\boldsymbol{x_r}$-axes are $\boldsymbol{x_g}$-axes, all $\boldsymbol{y_r}$-axes are $\boldsymbol{y_g}$-axes, and all $\boldsymbol{z_r}$-axes are $\boldsymbol{z_g}$-axes under the global coordinate system, the results of the element reduction are given in Table 2.

Table 2 Results of the node reduction for triple manipulator cooperation*

		Robot A	Robot B	Robot C	Nodes of PSSM
Before reduction	$\boldsymbol{x_r}$**-axis**	**3**: $(x_A^t - \Delta x_A,\ x_A^t,\ x_A^t + \Delta x_A)$	**3**: $(x_B^t + \Delta x_B,\ x_B^t,\ x_B^t - \Delta x_B)$	**3**: $(x_C^t + \Delta x_C,\ x_C^t,\ x_C^t - \Delta x_C)$	$27 \times 27 \times 27$
	$\boldsymbol{y_r}$**-axis**	**3**: $(y_A^t - \Delta y_A,\ y_A^t,\ y_A^t + \Delta y_A)$	**3:** $(y_B^t + \Delta y_B,\ y_B^t,\ y_B^t - \Delta y_B)$	**3:** $(y_B^t + \Delta y_C,\ y_C^t,\ y_C^t - \Delta y_C)$	
	$\boldsymbol{z_r}$**-axis**	**3**: $(z_A^t - \Delta z_A,\ z_A^t,\ z_A^t + \Delta z_A)$	**3** :$(z_B^t + \Delta z_B, z_B^t,\ z_B^t - \Delta z_B)$	**3** :$(z_C^t + \Delta z_C,\ z_C^t,\ z_C^t - \Delta z_C)$	

	x_r-axis	**2**: $(x_A^t,\ x_A^t + \Delta x_A)$	**2** :$(x_B^t + \Delta x_B,\ x_B^t)$	**2** :$(x_C^t + \Delta x_C,\ x_C^t)$		
	y_r-axis	**2**: $(y_A^t - \Delta y$ or $y_A^t)$	**2** :$(y_B^t - \Delta y_B$ or $y_B^t)$	**2** :$(y_B^t - \Delta y_B$ or $y_B^t)$		$7 \times 7 \times 7$ or $7 \times 7 \times 3$ or $3 \times 3 \times 3$
After reduction	z_r-axis: $\boldsymbol{z_A^t > z_B^t > z_C^t}$	**2**: $(z_A^t + \Delta z$ or $z_A^t)$	**1**: $(\ z_B^t)$	**2**: $(z_C^t - \Delta z$ or $z_C^t)$	**2, 2, 1**	
	$\boldsymbol{z_A^t > z_C^t > z_B^t}$	**2**: $(z_A^t + \Delta z$ or $z_A^t)$	**2**: $(z_B^t - \Delta z$ or $z_B^t)$	**1**: $(\ z_C^t)$		
	$\boldsymbol{z_B^t > z_A^t > z_C^t}$	**1**: $(\ z_A^t)$	**2**: $(z_B^t + \Delta z$ or $z_B^t)$	**2**: $(z_C^t - \Delta z$ or $z_C^t)$		
	$\boldsymbol{z_A^t > z_B^t = z_C^t}$	**2**: $(z_A^t + \Delta z$ or $z_A^t)$	**2**: $(z_B^t - \Delta z$ or $z_B^t)$	**2**: $(z_C^t - \Delta z$ or $z_C^t)$	**2, 2, 2**	
	$\boldsymbol{z_B^t = z_C^t > z_A^t}$	**2**: $(z_A^t - \Delta z$ or $z_A^t)$	**2**: $(z_B^t + \Delta z$ or $z_B^t)$	**2**: $(z_C^t + \Delta z$ or $z_C^t)$		
	$\boldsymbol{z_A^t = z_B^t = z_C^t}$	**1**: (z_A^t)	**1**: (z_B^t)	**1**: (z_C^t)	**1, 1, 1**	

**: "–" in the table means moving away from the target point; "+" means moving towards the target point. Here, all cases are not enumerated all, but still represent all possible scenarios for number of element of the PSSM of three cooperating manipulators*

In Table 2, the current positions of the manipulators are $\boldsymbol{P_A^t} = (x_A^t, y_A^t, z_A^t)$, $\boldsymbol{P_B^t} = (x_B^t, y_B^t, z_B^t)$, and $\boldsymbol{P_C^t} = (x_C^t, y_C^t, z_C^t)$, under each reference direction vectors.

5 Cost function and optimal solution selection

Formula representing the cost function for multiple manipulators scenarios are presented in this section. Assuming there exist N manipulators, for the j-th manipulator, the number of nodes in the PSSM at the current time instant is m and the configuration of the i-th node at current time instant t are

$$\boldsymbol{\theta_i^t} = (\theta_{1i}^t, \theta_{2i}^t, \theta_{3i}^t, \theta_{4i}^t, \theta_{5i}^t, \theta_{6i}^t) \tag{18}$$

The configuration of the i-th node at time instant $(t+1)$ is

$$\boldsymbol{\theta_i^{t+1}} = (\theta_{1i}^{t+1}, \theta_{2i}^{t+1}, \theta_{3i}^{t+1}, \theta_{4i}^{t+1}, \theta_{5i}^{t+1}, \theta_{6i}^{t+1}) \tag{19}$$

The EEF position of the i-th node at time instant $(t+1)$ under global coordinate system is

$$\boldsymbol{P_i^{t+1}} = (x_i^{t+1},\ \ y_i^{t+1},\ \ z_i^{t+1}) \tag{20}$$

The target position of the i-th node under global coordinate system is

$$\boldsymbol{P_i^g} = \left(x_i^g,\ \ y_i^g,\ \ z_i^g\right) \tag{21}$$

Finally, the cost function using the classical A* algorithm is

$$F(t) = H(t) + G(t) \tag{22}$$

where $F(t)$ is the total cost value of the current node in the search map at the current time instant, $H(t)$ is the cost value from the current node to candidate node, and $G(t)$ represents the cost value from the current node to target node. The motion of a manipulator can be expressed in both C-space and execution space, therefore the cost value can also be calculated in these two spaces. However, the distances between the candidate nodes and the current node are all the same in the execution space, such that $H(t)$ in the configuration space and $G(t)$ in execution space can be calculated individually. The cost value for the i-th node in the PSSM of the j-th manipulator can be calculated as

$$\begin{aligned} F_i^j(t) &= \omega_i^H \times H_i(t) + \varphi_i^G \times G_i(t) \\ &= \omega_i^H \times \left\|\boldsymbol{\theta_i^{t+1}} - \boldsymbol{\theta_i^t}\right\| + \varphi_i^G \times \left\|\boldsymbol{P_i^g} - \boldsymbol{P_i^{t+1}}\right\| \end{aligned} \tag{23}$$

where $\|\boldsymbol{A}\|$ is the secondary norm of vector $\boldsymbol{A}$, and ω_i^H and φ_i^G are cost coefficients, which can be predefined. The total cost value of the i-th node for N manipulators at the current time instant t is

$$F_i(t) = \sum_{j=1}^{N} c_j \times F_i^j(t) \tag{24}$$

where $\boldsymbol{C}_t = (c_1, c_2, \ldots, c_N)$ are the Boolean values (0 or 1) that determine whether the corresponding manipulator will execute the proposed path planning.

Further, the selection process of the optimal solutions can be defined as the selection of optimal nodes in the PSSM for all manipulators, and summarized as the following mathematical model:

$$W_{t+1} = min[F_1(t), F_2(t), \ldots, F_i(t), \ldots, F_m(t)] \tag{25}$$

6 Simulation and experiments

6.1 Simulation of minimum distance prediction

6.1.1 Simulation set up

In the simulation, the joint values of the two manipulators are used as input data to calculate the minimum distance. Therefore, the training data set has 12 inputs and 1 output. BP neural network constructed by MATLAB toolbox has two hidden layers with 20 and 6 neurons in each layer respectively, and the activation functions in hidden layers are tansig function:

$$f(x) = (1 - e^{-x})/(1 + e^{-x}) \tag{26}$$

while in the output layer the activation function is pure line function $f(x) = x$.

The dimension of each particle depending on the number of the connection weights and threshold can be calculated by the formula: $D = n_0 n_1 + n_1 + n_1 n_2 + n_2 + n_2 n_3 + n_3$.

And the population size of the particles is set to 60 which are divided into 3 subgroups to keep the swarm variety. The constant values in formula (6) are set to $c_1 = 2.8, c_2 = 1.3, c_3 = 2$. As for the iteration of the PSO, in order to get a tradeoff between accuracy and computational cost, the values of T_1 and T_1 are set to 3,000 and 1,000 respectively.

In order to eliminate variable type difference and saturation caused by large range of the input, the input data should be normalized into interval [-1,1] before training. The following formula is employed to pre-process the data.

$$y = 2 * \frac{x - x_{min}}{x_{max} - x_{min}} - 1 \quad (27)$$

where x_{min}, x_{max} are the minimum and maximum value of the inputs with respect to the exactly same feature, namely the values of the same joint in simulation and x represents the training data.

6.2.2 Simulation result

A training data set with size of 300,000 samples in which each sample is composed of 12 values of joints and the minimum distance is used for training PSO-BP neural network, and conduct 10 trials to eliminate the random factors for a general result. The optimization process with respect to the two stages of PSO is showed in figure 13 and the training of BP neural network with optimized weights is presented in figure 14.

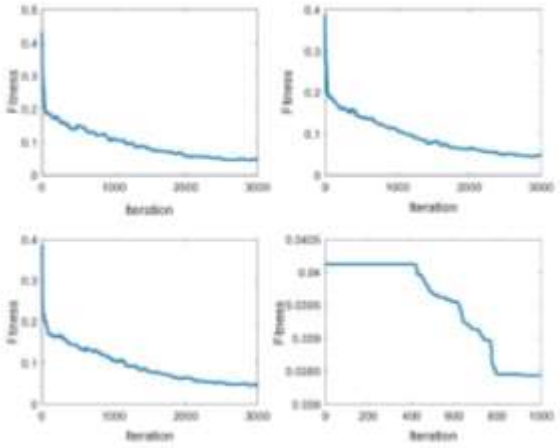


Fig.13 Fitness-iteration of the PSO. (a)(b)(c) correspond to three subgroups in the first optimization stage and (d) is the second optimization stage of the hybrid method. All of the fitness values are average of the ten trials.

In figure 13, it can be seen that the fitness of the particle decreased with the iteration increasing which shows a convergence process. The subfigure (a)(b)(c) corresponding to the three subgroups of the particles in the first optimization stage show that there's vibration of the fitness that does not appear in the subfigure (d). Because the particle with best fitness in a subgroup may be grouped into another one, and thus the best fitness of one of the subgroups might change and be larger than the original one which indicates the k-mean clustering algorithm works well in keeping the population diversity of the particles. In order to get a better optimization solution, the particles with best fitness in the subgroups form a new swarm used in the second stage. Subfigure (d) demonstrates that a relative optimal result is obtained in the second stage.

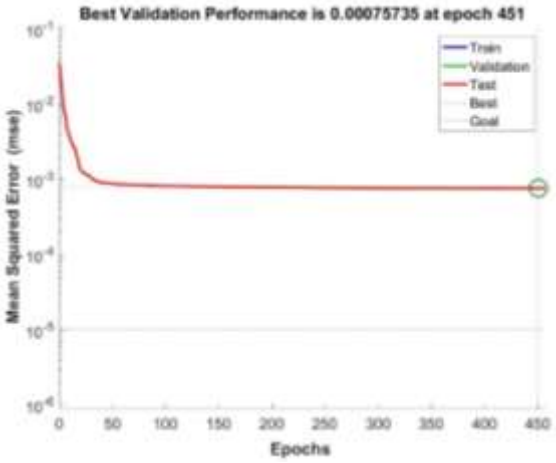


Fig.14 Training of BP neural network

The performance of training results is evaluated on a test set with a size of 5,000 through MSE and MAE from statistic perspective. Table 3 lists the MSE and MAE values of each trial respectively.

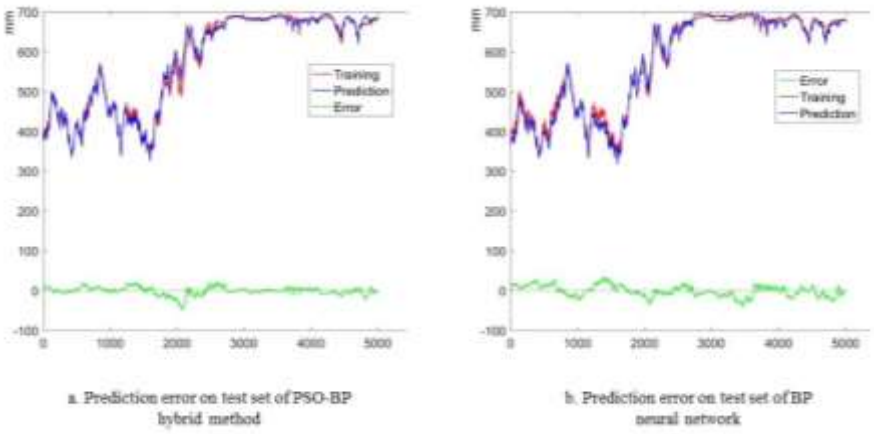
Table 3. Comparison of the prediction results

Trial	PSO-BP hybrid method		BP neural network	
	MAE	MSE	MAE	MSE
1	10.53	189.82	15.80	418.25
2	17.11	448.71	13.88	293.92
3	7.16	101.10	20.23	713.56
4	20.32	622.01	10.42	204.61
5	10.48	171.55	15.45	351.21
6	9.47	132.49	9.13	124.08
7	10.81	183.23	14.70	323.86
8	8.07	109.71	9.57	148.07
9	13.82	308.00	11.41	204.36
10	9.36	138.31	13.63	272.42
Average	11.72	240.49	13.43	305.43

Table 3 demonstrates that the PSO-BP hybrid method gets a more accurate result than the standard BP neural network with $1.71mm$ less average MAE and $64.94mm^2$ less average MSE in ten trials. Considering the computational cost, if a larger size of population of particles and more iterations are applied in PSO, a better solution is supposed to attain in a high dimensional problem which is 393 in this paper.

The prediction results of the best trials which is the 3rd trial in hybrid method and 6th trial in BP neural network are selected to showed in figure 15. And it can be seen obviously that errors in (b) are larger and oscillate more intensely than those in (a). No matter what the algorithm is, it is impossible

to predict the distance exactly without any error. Since when the line segments are in different relative position and orientation the computation formula is different in mathematics, the learning algorithm needs to compromise different situation. Besides, the different radii of sphere-swept volume approximated the manipulator can lead to error while the learning algorithm intends to make general prediction, in which it needs to get a tradeoff among different radii.



a. Prediction error on test set of PSO-BP hybrid method

b. Prediction error on test set of BP neural network

Fig.15 Prediction error on test set of PSO-BP method and BP neural network

6.2 Simulation of path planning

In this paper, two different methods were proposed to set up the PSSM for systems with multiple cooperating manipulators. In this section, the two methods are first tested for the dual manipulator scenario, and the simulation results are compared. Subsequently, the scenario of a triple manipulator system is used as an example to show how the method works in the situation of multiple manipulators. Finally, the simulation results are analyzed and some conclusions are drawn from the results. Moreover, the execution time of each method is assessed to determine whether it is possible meet real-time requirements.

Assuming N manipulators in a system of multiple manipulator cooperating, all 6-DOF industrial manipulators have common D-H parameters, which are presented in Table 4.

Table 4 D-H parameters for a 6-DOF manipulator

Link j	Link offset d_j	Link length a_{j-1}	Twist angle of link α_{j-1}	Rotation angle of link θ_j
1	0	0	0	θ_1
2	0	a_2^i mm	-90°	θ_2
3	0	a_3^i mm	0	θ_3
4	d_4^i mm	a_4^i mm	-90°	θ_4
5	0	0	90°	θ_5
6	0	0	-90°	θ_6

The simulation results for dual-manipulator and triple-manipulator system will be analysed individually. For more than three manipulators, the results were relatively similar to those obtained for the triple-manipulator system, except for increases in number of the PSSM elements and execution time. All the algorithms were programmed in C++ and run on a 64-bit Windows operation system with an i7-4790 CPU GHz with 8 GB RAM.

6.2.1 Simulation of dual-manipulator cooperate manufacturing system

For manipulators A and B, we can define $a_2^i = 25$ mm, $a_3^i = 315$ mm, $a_4^i = 35$ mm, and $d_4^i = 365$ mm (i=1,2), where A and B were given the same parameters and positioned face-to-face at a distance of $D = 800$ mm. The minimum tolerance for the distance between the two manipulators was set as $D_{min} = 20$ mm. The global coordinate system originated at the base of B, which means the coordinates of the base of B were $(0,0,0)$ and for base of A were $(0,-800,0)$. The step length of each manipulator and boundary values were predefined as $l_s = 10$ mm and $\varepsilon = 40$ mm, respectively.

Three simple simulation scenarios were established as follows:

(a) Manipulator A moves from $\boldsymbol{P_A^{start}} = (400,-350,600)$ to $\boldsymbol{P_A^{end}} = (-400,-350,600)$, while manipulator B simultaneously moves from $\boldsymbol{P_B^{start}} = (-400,-450,600)$ to $\boldsymbol{P_B^{end}} = (400,-450,600)$. If no collision occurs, A and B will move from start point to target point in a straight line of displacement only along the main movement axis ($\boldsymbol{x}$-axis).

(b) Manipulator A moves from $\boldsymbol{P_A^{start}} = (400,-300,580)$ to $\boldsymbol{P_A^{end}} = (-400,-500,580)$, while manipulator B simultaneously moves from $\boldsymbol{P_B^{start}} = (-400,-300,600)$ to $\boldsymbol{P_B^{end}} = (400,-500,600)$. If no collision occurs, A and B will move from start point to target point in a straight line of displacement along the main movement axis and secondary movement axis ($\boldsymbol{x}$- and $\boldsymbol{y}$-axis).

(c) Manipulator A moves from $\boldsymbol{P_A^{start}} = (400,-350,400)$ to $\boldsymbol{P_A^{end}} = (-400,-350,600)$, while manipulator B simultaneously moves from $\boldsymbol{P_B^{start}} =$

$(-400, -450, 400)$ to $\boldsymbol{P_B^{end}} = (400, -450, 600)$. If no collision occurs, A and B will move from start point to target point in straight line of displacement along the main movement axis and displacement axis ($\boldsymbol{x}$- and $\boldsymbol{z}$-axis).

1) Results of simulation using coordinate system based method

According to the principles of the coordinate system based method, the main movement axis, secondary movement axis, displacement axis, and main movement plane in the above three scenarios are the $\boldsymbol{x}$-axis, $\boldsymbol{y}$-axis, $\boldsymbol{z}$-axis and $\boldsymbol{xoy}$-plane.

Results of the simulation using the coordinate system based method for each of the above scenarios are presented in Fig. 16.

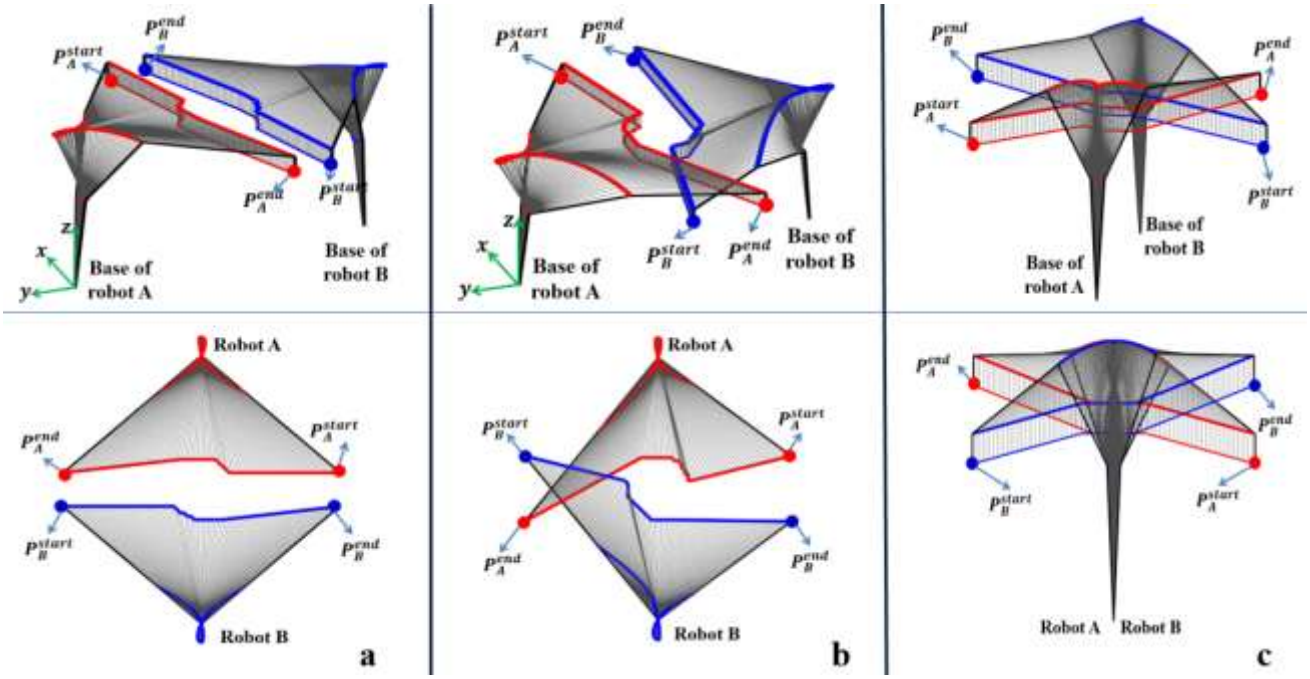


Fig. 16 Simulation results of path planning for dual-manipulator cooperation using coordinate system based method， where a, b and c correspond to the simulation results under scenarios *a*, *b*, and *c*, respectively. The paths of each joint of A and B in Cartesian space are represented by the red and blue curves, respectively. The upper images show the 3D views, and the lower images are the top view.

Movements along the $\boldsymbol{x}$- $\boldsymbol{y}$-, and $\boldsymbol{z}$-axis for both manipulators are shown in Fig.17.

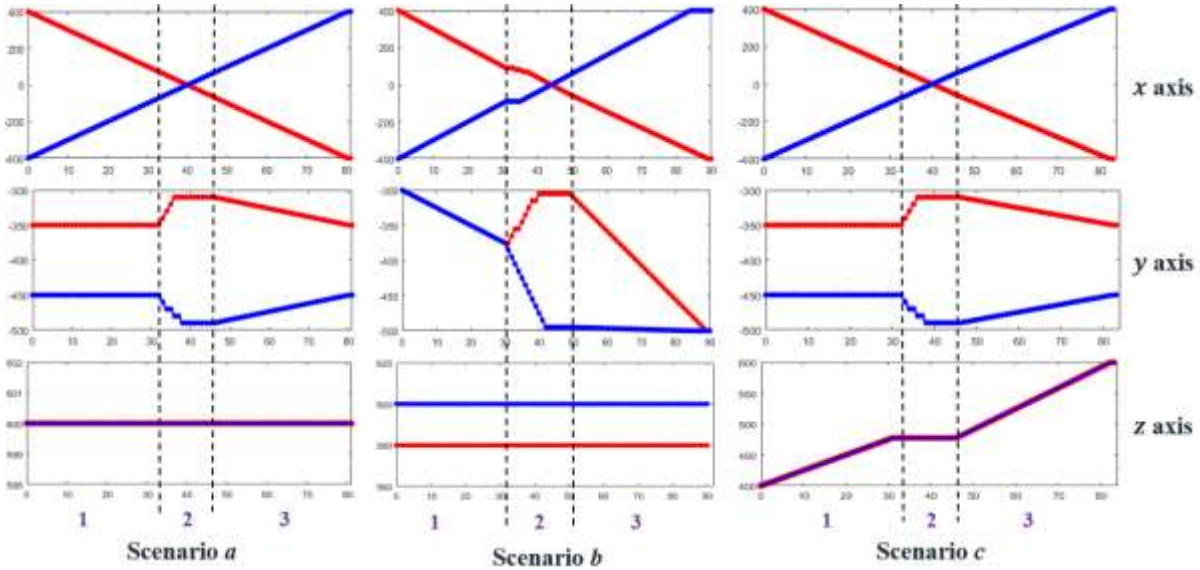


Fig. 17 Movements along $\boldsymbol{x}$-, $\boldsymbol{y}$-, and $\boldsymbol{z}$-axis of manipulators A and B for scenarios *a*, *b*, and *c* using coordinate system based method. The red and blue curves represent the movements of manipulators A and B, respectively. For each image, the horizontal axes represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

Moreover, the paths of both manipulators are clearly divided into three distinct parts, marked as 1, 2, and 3 (Fig. 17) and named the “original path period”, “avoidance period”, and “forward period”. The “original path period” indicates

the manipulator moved along its original path without deviation, the "avoidance period" is the case where a manipulator path was re-planned in order to avoid a collision, and the "forward period" is the last situation for which the manipulator moved toward its target along a straight line.

For any scenario, it can be observed that when a collision is about to occur, it will be avoided by manipulator A in the $-\boldsymbol{y}$ direction and avoided by B in the $+\boldsymbol{y}$ direction, indicating that both manipulators move toward their respective bases. In this way, the two manipulators bypass the Possible Collision Area (PCA) and achieve the goal of collision avoidance. However, the coordinate values of the $\boldsymbol{z}$-axis stay constant during the "avoidance period" for all the three scenarios. The reason for this is that for each of the three scenarios, when the two manipulators moved into the PCA, the coordinate values of the $\boldsymbol{z}$-axis of the two manipulators $\boldsymbol{Z}_A^t$ and $\boldsymbol{Z}_B^t$ met the condition of $\|z_A^t - z_B^t\| \leq \varepsilon$, as described in Section 4.1. Here, simulation d was set up to observe how the coordinate values of the $\boldsymbol{z}$-axis would change when the condition $\|z_A^t - z_B^t\| \leq \varepsilon$ is untenable.

(d) Manipulator A moves from $\boldsymbol{P_A^{start}} = (400, -350, 400)$ to $\boldsymbol{P_A^{end}} = (-400, -350, 600)$, while manipulator B simultaneously moves from $\boldsymbol{P_B^{start}} = (-400, -450, 500)$ to $\boldsymbol{P_B^{end}} = (400, -450, 500)$. If no collision occurs, A will move from the start point to target point in a straight line of displacement along the main movement axis and displacement axis ($\boldsymbol{x}$- and $\boldsymbol{z}$-axis), while B will move from the start point to target point in a straight line of displacement along the main movement axis ($\boldsymbol{x}$-axis).

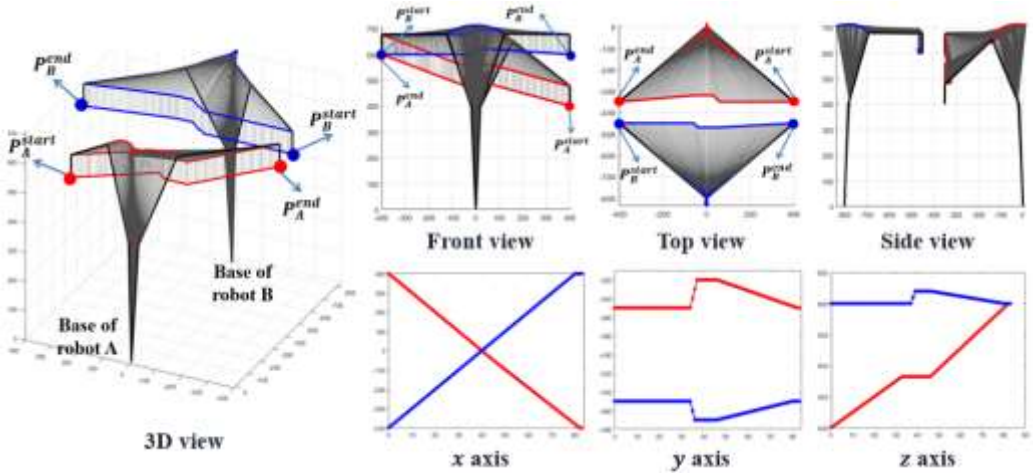


Fig. 18 Results of path planning simulation and movements along $\boldsymbol{x}$-, $\boldsymbol{y}$-, and $\boldsymbol{z}$-axis for dual-manipulator cooperation under scenario *d* via coordinate system based method

According to the results presented in Fig. 18, when the two manipulators move into the PCA, $\|z_B^t - z_A^t\| > 100 > \varepsilon$, therefore B has the choice of "$+\boldsymbol{z}$" and "remain stationary" and A has the choice of "$-\boldsymbol{z}$" and "remain stationary", according to the principles of the PSMS method. The results show that z_A^t remained stationary while z_B^t changed along $+\boldsymbol{z}$ direction in the "avoid period".

Real-time execution is also a key assessment index of our algorithms. In order to get the per step execution time of the PSMS method, manipulator A was driven move from $\boldsymbol{P_A^{start}} = (400, -450, 600)$ to $\boldsymbol{P_A^{end}} = (-400, -450, 600)$ and B was moved from $\boldsymbol{P_B^{start}} = (-400, 450, 600)$ to $\boldsymbol{P_B^{end}} = (400, 450, 600)$ in the base coordinate system of each manipulator, simultaneously. Then, the base distance D was changed from 1000 mm to 900 mm to 800 mm, and each change was tested 5 times. The simulation results are summarized in Fig. 19.

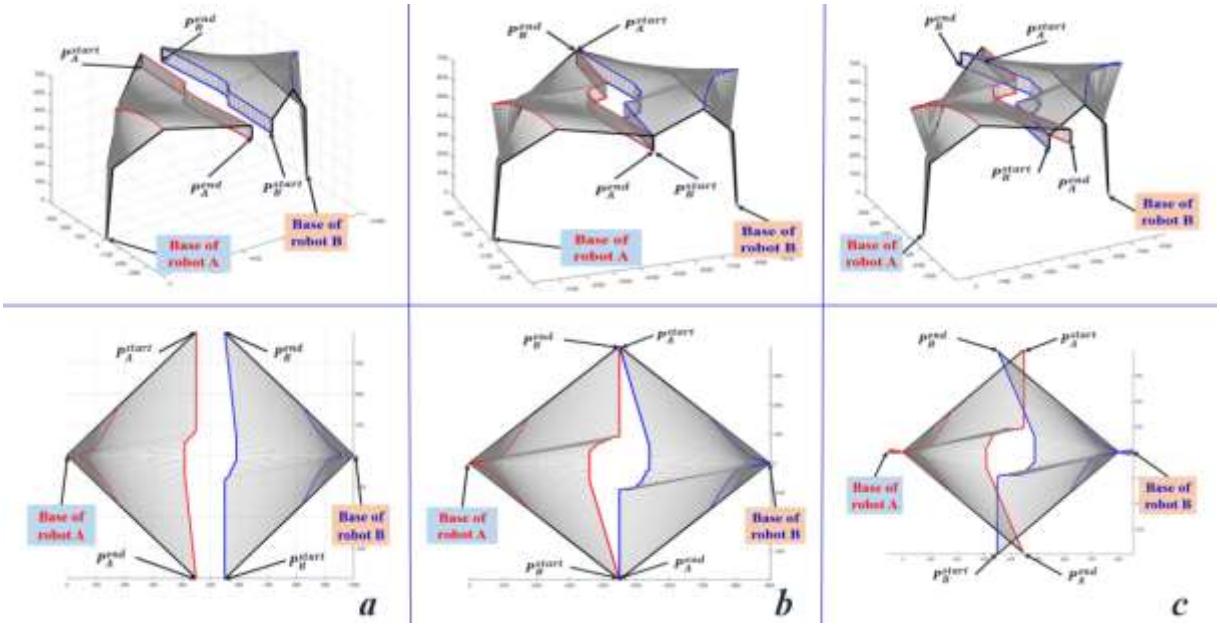


Fig. 19 Results of path planning simulation for dual-manipulator cooperation based on coordinate system based method with decreasing distance between the bases of two manipulators, where a, b and c are the simulations with distances of 1000 mm, 900 mm and 800 mm, respectively.

The test results of execution time of whole path planning period are summarized in Table 5.

Table 5 Execution time for simulation based on dual manipulators (unit: \μs) for different distances

Simulation Num.	D = 1000 mm 81 steps	D = 900 mm 86 steps	D = 800 mm 91 steps
1	44881.44	67200.45	75222.38
2	44856.35	67678.86	74210.83
3	45633.26	69841	74384.46
4	44125.07	68584	76169.49
5	44267.05	65112.65	74865.43
Average	44752.63	67683.4	74970.52

As previously stated, the entire movement period can be divided into the "original period", "avoidance period" and "forward period", and a different path planning algorithm is required for each period. The execution times of each period are defined by the variables T_{op}, T_{ap}, and T_{fp}, and by default, the three variables remain constant for each period. Fig.20 shows the movements of manipulators A and B (red and blue, respectively) along the y- axis for the previously defined scenarios, and each of the three periods is demarcated.

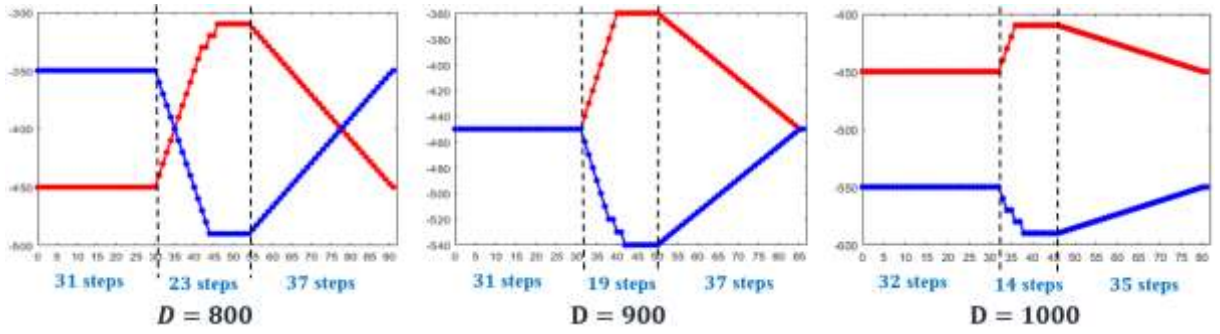


Fig. 20 Simulation results of movements along axis y in the scenarios of Fig.19. The red and blue curves represent the movements of manipulators A and B,

respectively. For each image, the horizontal axes represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

Using a timer program embedded in the algorithm, it was possible to obtain average values of T_{op}, T_{ap} and T_{fp}:

$T_{op} = 69.48\mu s;\ \ T_{ap} = 2893.4\mu s;\ \ T_{fp} = 78.28\mu s.$

Then

$$32 \times T_{op} + 14 \times T_{ap} + 35 \times T_{fp} = 45470.76$$
$$31 \times T_{op} + 19 \times T_{ap} + 37 \times T_{fp} = 60024.84$$
$$31 \times T_{op} + 23 \times T_{ap} + 37 \times T_{fp} = 71598.44$$

Comparing the above results to those presented in Table 5, we can see that the errors are within the allowable range. The results suggest that path planning using the PSMS method for dual manipulators can be executed in less than 3 ms, and can therefore fully meet real-time requirements.

2) Results of simulation using velocity based method

The simulation results for path planning, using the current velocity based method under scenarios a, b, and c, are presented in Fig. 21.

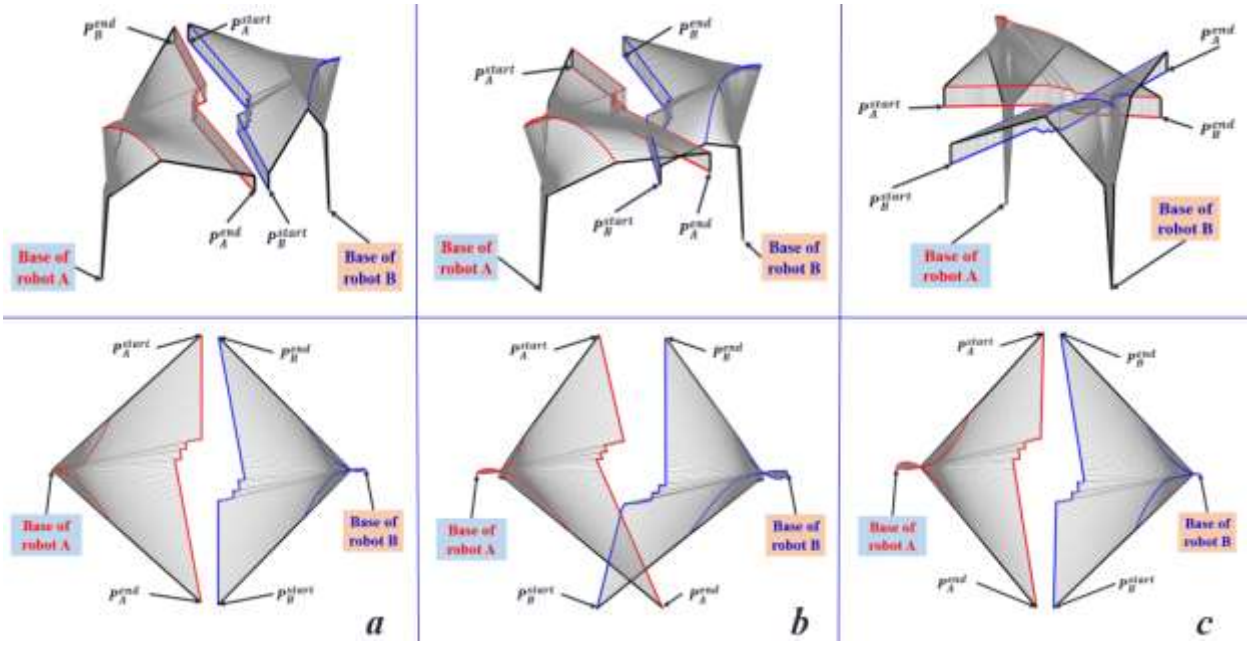


Fig. 21 Results of path planning simulation for dual-manipulator cooperation using velocity based method where a, b and c correspond to results for scenarios *a*, *b* and *c*, respectively. The paths of all joints in A and B in Cartesian space are represented by the red and blue curves, respectively. The upper images show 3D views of each scenario, while the lower images are the top view.

The movements along the x-, y-, and z-axis for both manipulators are shown in Fig. 22. From the three images shown in the lower half of Fig. 21, it can be observed that regardless of scenario, when a collision is about to occur, both A and B can plan a new path with different geometries to those presented in Fig. 16. However, A will usually avoid a collision by moving in the $-y$ direction, while B will move in the $+y$ direction, indicating both manipulators will move towards their own base. In this way, the two manipulators bypass the PCA and achieve the goal of collision avoidance. The coordinate values of the z-axis remained constant during the "avoidance period" for all three scenarios, which were the same as those in the coordinate system based method. The simulation results (Fig. 23) show movements along x-, y-, and z-axis under scenario d using the velocity-based method, clearly shown to be different along the z-axis to those using the coordinate system based method (Fig. 18), mainly due to differences in the principles of each method.

To examine the per step execution time using the velocity based method, the same simulation scenario was established as for the coordinate system based method (Fig. 19). The simulation results are summarized in Fig. 24.

The results for execution time over the whole path-planning period are summarized in Table 6.

Table 6 Execution time for simulation based on dual manipulators (unit: \μs) for different distances.

Num.	$D = 1000\ mm$ /86 steps	$D = 900\ mm$ /91 steps	$D = 800\ mm$ /99 steps
1	38614.28	58685.52	90809.24
2	39114.92	57553.66	89090.92
3	38230.82	59227.22	90670.96
4	39623.54	58248.74	88377.31
5	38254.76	59687.66	89026.49
Average	38767.66	58680.56	89594.98

Using a timer program embedded in the algorithm, it was possible to obtain average values of T_{op}, T_{ap}, and T_{fp}

given by

$$T_{op} = 76.34\mu s;\ \ T_{ap} = 3350.2\mu s;\ \ T_{fp} = 89.41\mu s.$$

Then

$$33 \times T_{op} + 9 \times T_{ap} + 44 \times T_{fp} = 36605.06$$
$$31 \times T_{op} + 15 \times T_{ap} + 45 \times T_{fp} = 56642.99$$
$$31 \times T_{op} + 25 \times T_{ap} + 43 \times T_{fp} = 89966.17$$

Comparing the results to those from Table 6, errors were within the allowable range. The results suggest that path planning using the PSMS method for dual manipulators can be executed in less than 4 ms, which can fully meet real-time requirements.

1) Comparison of simulation results

Based on the analysis and results of the previous sections, both methods can be used to plan non-collision paths for dual-manipulator cooperation systems in real time. However, a number of similarities and differences can be observed upon carefully comparing of the two methods:

(a) The simulated paths of both methods are not smooth throughout, and there were "twists and turns" in the geometries. One reason may be that all nodes in the search map based on the PSMS method are special positions within the SFR. Another may be the influence of step length which will be simulated and analysed in Fig.26.

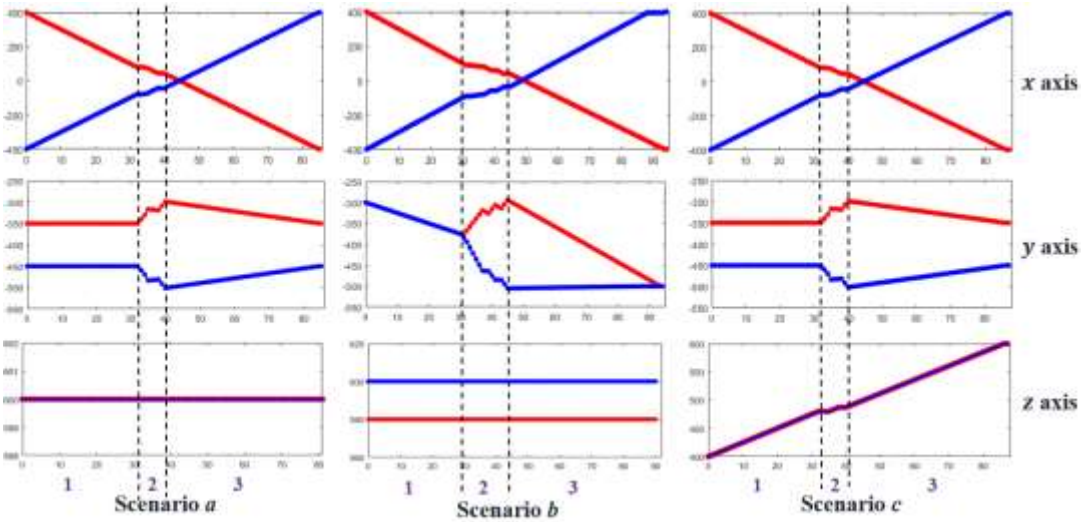


Fig. 22 Movements of manipulators A and B along x-, y-, and z-axis for scenarios *a*, *b* and *c* using velocity based method. The red and blue curves represent the movements of manipulators A and B, respectively. For each image, the horizontal axes represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

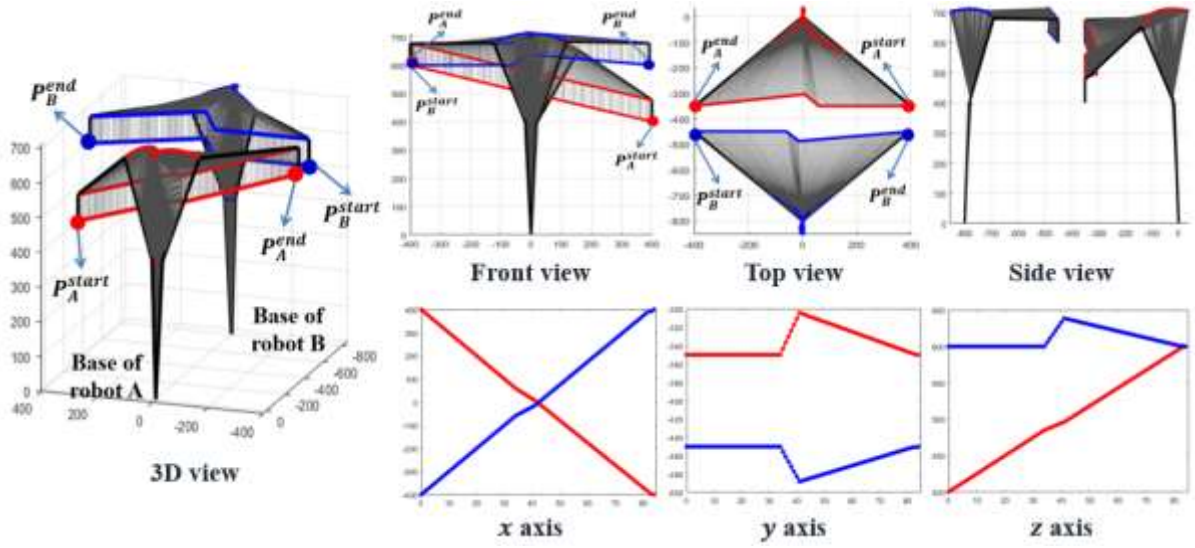


Fig. 23 Results of path planning simulation and movements along x-, y-, and z-axis dual-manipulator cooperation under scenario *d* via velocity based method

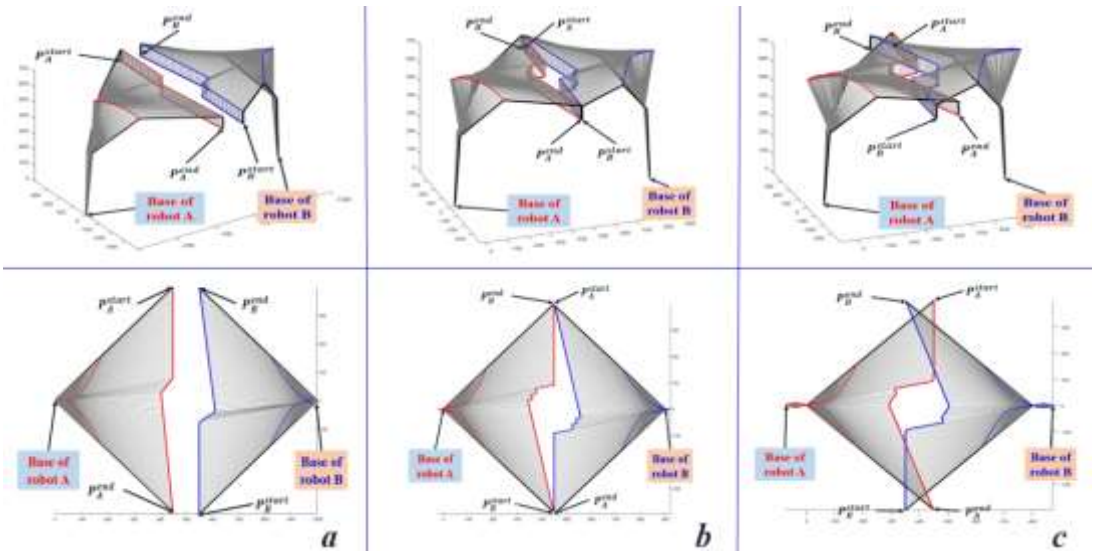


Fig. 24 Results of path planning simulation of dual-manipulator cooperation using velocity based method with decreasing distance between the bases of two manipulators where *a*, *b* and *c* represent the simulation results with distances of 1000 mm, 900 mm and 800 mm, respectively.

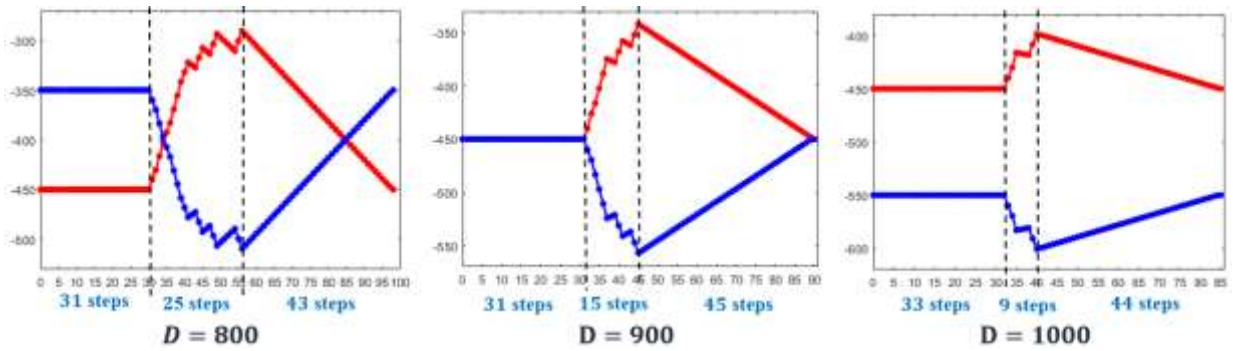


Fig. 25 Simulation results of movements along axis y in the scenarios of Fig.19. The red and blue curves represent the movements of manipulators A and B, respectively. For each image, the horizontal axes represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

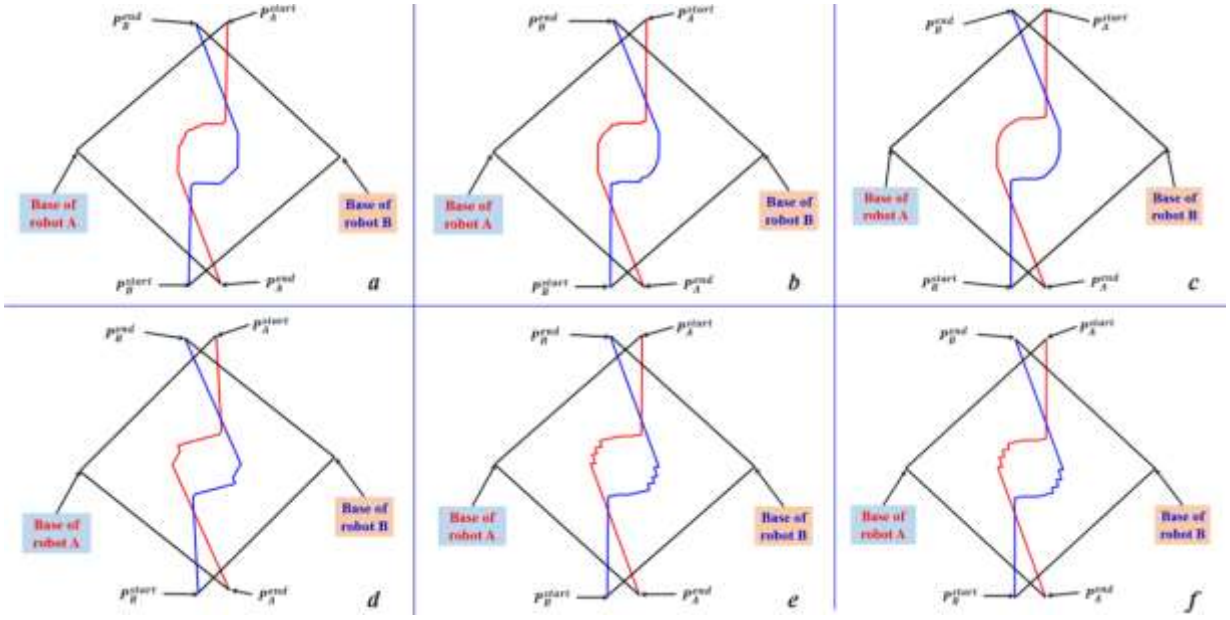


Fig. 26 Simulation results for dual-manipulator cooperation using PSMS method with decreasing step lengths, (a) 10 mm, (b) 5 mm, and (c) 1 mm for the coordinate system based method and (d) 10 mm, (e) 5 mm, and (f) 1 mm for the velocity based method.

(b) The geometries of the paths simulated using each of the two methods are not the same. Comparing Figs. 16 and 21, it

can be seen that the paths generated using the velocity based method had larger displacements along $\boldsymbol{x}$- and $\boldsymbol{y}$- axis than the coordinate system based method, however the planned stepped within the "avoidance period" were smaller by using velocity based method than that of the coordinate system based method.

(c) The simulated paths of the velocity based method (Figs. 21~25) had more apparent "twists and turns" than those of the coordinate system based method (Figs. 16~20). This is due to changes in the reference direction vectors throughout the entire planning period in the velocity-based method, whereas the reference direction vectors remain constant in the coordinate-based method.

In order to clearly identify the influence of step length on the flexibility of the simulated paths under different methods, simulations were performed using both methods according to scenario c (Fig.19 and Fig.24), with step lengths of 10 mm, 5 mm, and 1 mm. The results are shown in Fig. 26. While a decrease in step length resulted in a smoother path using the coordinate system based method, there was little effect shown on paths generated using the current velocity based method (Fig. 26). The reason is not totally clear, however, one possibility is that the reference direction vectors of the velocity based method changes continuously throughout the planning period whereas the reference direction vectors of the coordinate based method remain constant.

Furthermore, the operating time required for the velocity-based method is longer than the coordinate system based method. As such, the coordinate system based method is more effective and demonstrates greater stability in comparison to the velocity-based method. Therefore, the coordinate system based method will be used to demonstrate use of the PSMS method for triple manipulator cooperation.

6.2.2 Simulation of triple-manipulator manufacturing system

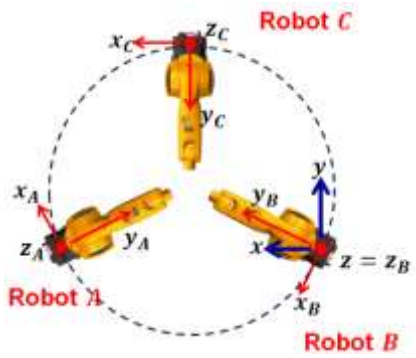


Fig. 27 Schematic of scenario for triple-manipulator cooperation

For the three cooperating manipulators A, B and C, we can define their D-H parameters as:

$a_2^i = 25\text{mm};\quad a_3^i = 315\text{mm};\quad a_4^i = 35\text{mm};\quad d_4^i = 365\text{mm}$ (i=1,2,3)

where A, B, and C were given the same parameter values, and the relationships between the positions of the manipulators are shown in Fig. 27. The bases of the triple manipulators consists of an equilateral triangle with length of sides $900\ mm$. The minimum tolerance for the distance between two manipulators was set as $D_{min} = 50\ mm$. The global coordinate system originated at the base of B, which means the coordinates of the base of A, B, and C were $(900,0,0)$, $(0,0,0)$, and $\left(450,450\sqrt{3},0\right)$.

During the simulation, manipulator A moves from $\boldsymbol{P_A^{start}} = (660.3,571.4,400)$ to $\boldsymbol{P_A^{end}} = (260.3,-121.4,600)$ while manipulator B simultaneously moves from $\boldsymbol{P_B^{start}} = (589.7,-121.4,600)$ to $\boldsymbol{P_B^{end}} = (189.7,571.4,400)$, and manipulator C simultaneously moves from $\boldsymbol{P_C^{start}} = \left(\boldsymbol{25,450(\sqrt{3}-1),500}\right)$ to $\boldsymbol{P_C^{end}} = (825,450(\sqrt{3}-1),500)$, all under the global coordinate system.

The simulation results based on the above scenario are presented in Fig. 28.

When a collision is about to occur, all manipulators move towards their respective bases, successfully bypassing the PCA and achieving the goal of collision avoidance, as shown in Fig. 23.

Again, to examine the per step execution time using the PSMS method for triple-manipulator cooperation, the same three simulation scenarios were used for which only the base distance D was varied from 950 mm to 900 mm to 850 mm, and each simulation was tested 5 times. Execution times for each scenario over the entire path-planning period are summarized in Table 7.

Table 7 Execution time for simulation based on triple manipulators (unit: \μs) for different distances

Num.	$\boldsymbol{D = 950\ mm}$ **/88 steps**	$\boldsymbol{D = 900\ mm}$ **/84 steps**		$\boldsymbol{D = 850\ mm}$ **/101 steps**
1	137257.11	137562.24		181038.46
2	135620.94	139697.94		178944.83
3	136982.56	138442.63		180225.33
4	135779.06	143752.65		177698.12
5	138257.86	138176.63		181908.09
Average	136779.5	139526.42		179962.97

Since the planning process of triple-manipulator cooperation is complex, the complete period of path planning can be divided into four parts: period of simultaneous triple-manipulator avoidance; period of simultaneous dual-manipulator avoidance; period of single manipulator avoidance; and non-collision period. For each of the defined periods, a distinct solution to the algorithm is required, which will lead to different execution times. As such, it is difficult to determine the exact execution time for each period. However, using the timer program embedded within the algorithm, it was possible to determine average values of execution time for triple manipulators, which were are all less than 10 ms.

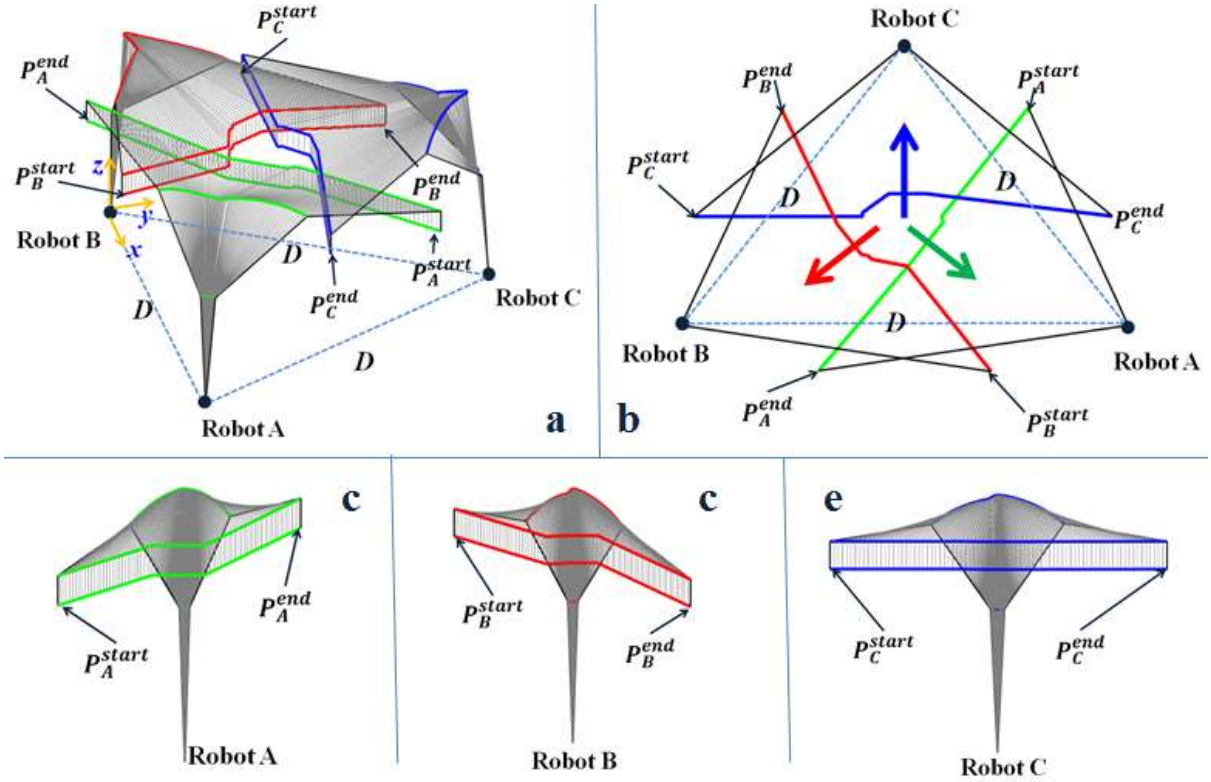


Fig. 28 Results of triple-manipulator cooperation simulation using PSMS method: (a) 3D view; (b) overview of paths; (c-d) side view of paths for each manipulator A, B, and C, respectively. The green, red, and blue curves represent the paths of all joints of manipulator A, B, and C, respectively, in Cartesian space.

7 Conclusion & Future research

In this paper, a novel path planning method called SbPSMS was proposed, which can successfully facilitate the planning of non-collision paths for all manipulators working under multiple-manipulator cooperate manufacturing scenarios according to the real-time circumstance at any time while all manipulators are working along their own tasks without pause.

Firstly, a method based on PSO-BP neural network to predict the potential collision of the dual manipulators system is presented. The hybrid algorithm utilizes PSO algorithm to optimize BP neural network initial weights to obtain a better result. Finally, the BP neural network outputs the predicted minimum distance and potential collision is detected if the minimum distance is less than the safety distance threshold. Through the given training data set, the MAE of the predicted minimum distance on the test set is about 1.71mm less than standard BP neural network.

Then, two different methods were presented to build up the three reference direction vectors in order to determine the candidate positions of all manipulators: coordinate system based method and velocity based method. Further, the process for decreasing the number of nodes of the PSSM, in order to meet real-time execution requirements, was described in detail. Finally, a series of simulation tests were performed, and demonstrated that the SbPSMS method can successfully achieve collision avoidance of manipulators whilst meeting the real-time requirements. However, different methods of setting up the PSSM may lead to different path geometries. Moreover, the method itself dictates the influence of step length on the smoothness of the path. Whilst decreasing the step length resulted in a smoother path for the coordinate system based method, minimal effects were seen when using the current velocity based method. Furthermore, the operation time of the current velocity based method is longer than that of the coordinate system based method. Therefore, it can be concluded that the coordinate system based method is more effective and demonstrated greater stability. The simulation of the coordinate system based method with triple manipulator cooperation produced satisfactory results, suggesting our algorithm can be extended to applications using multiple manipulators.

In future work, the maximum number of cooperating manipulators that the SbPSMS can deal with simultaneously should be clarified. In addition, planning of paths within entire SFR will be a major focus.

Declarations

Author contribution Chang Su conceived of the study, designed the study, and wrote the manuscript. All authors were involved in collecting and analyzing the data, also revisions

Funding This research is supported by the Bosch (China) Investment Ltd. "Smart Manufacturing" international cooperation projects with Huazhong University of Science & Technology (Grant No.: 0232100008).

Availability of data and materials The authors confirm that the data and materials supporting the findings of this study are available within the article.

Ethical approval The work contains no libelous or unlawful statements, does not infringe on the rights of others, or contains material or instructions that might cause harm or injury.

Consent to participate The authors consent to participate.

Consent to publish All authors agree to transfer copyright of this article to the publisher.

Competing Interests The authors declare that they have no conflict of interest.

Reference

1 Caccavale, Fabrizio, and Masaru U (2016) Cooperative manipulation. Springer Handbook of Robotics. Springer International Publishing, pp 989-1006
2 Jonas S, Verena K, Christin H and Klaus B (2015) Human Centered Assistance Applications for the working environment of the future. Occupational Ergonomics 12:83–95
3 Iina A, Timo S, Ilari M (2018) Refining levels of collaboration to support the design and evaluation of human-robot interaction in the manufacturing industry. 51st CIRP Conference on Manufacturing Systems 72:93–98
4 M Choi, S Lim, J Lee, et al (2010) Cooperative Path Planning for Redundant Dual-Arm Robot Using Low-Dimensional Sample-Based Algorithm. Mechatronic Systems, pp: 701-708
5 A Nash, S Koenig, and C Tovey (2010) Lazy theta∗: any-angle path planning and path length analysis in 3D. Proceedings of the Third Annual Symposium on Combinatorial Search 2:153–154
6 S Karaman and E Frazzoli (2011) Sampling-based algorithms for optimal motion planning. International Journal of Robotics Research 30:846–894
7 R Koker (2013) A genetic algorithm approach to a neural-network based inverse kinematics solution of robotic manipulators based on error minimization. Information Sciences 222:528–543
8 Sucan IA and Chitta S (2012) Motion planning with constraints using configuration space approximations. IEEE International Conference on Intelligent Robots and Systems, pp:1904-1910
9 Jia Q, Chen G, Sun H and Zheng H (2010) Path planning for space manipulator to avoid obstacle based on A* algorithm. Chinese Journal of Mechanical Engineering 46: 109-115
10 Yu N and Wang Z (2011) Collision avoidance planning of manipulator based on C-space layered search arithmetic. 2011 International Conference on Electronic and Mechanical Engineering and Information Technology (EMEIT), pp:3258-3262
11 Li L, Shi Z, Guan Y, Zhao C, Zhang J and Wei H (2014) Formal verification of a collision-free algorithm of dual-arm robot in HOL4. IEEE International Conference on Robotics & Automation (ICRA), pp: 1380-1385
12 Harada K, Foissotte T, Tsuji T, Nagata K, Yamanobe N, Nakamura A and Kawai Y (2012) Pick and place planning for dual-arm manipulators. IEEE International Conference on Robotics and Automation, pp:2281-2286
13 Lee S, Moradi H and Yi C (2012) A real-time dual-arm collision avoidance algorithm for assembly. IEEE International Symposium on Assembly and Task Planning, pp:7-12
14 Larsen L, Pham V-L, Kim J and Kupke M (2015) Collision-free path planning of industrial cooperating robots for aircraft fuselage production. IEEE International Conference on Robotics and Automation, pp: 2042-2047
15 Cohen B, Chitta S and Likhachev M (2014) Single- and dual-arm motion planning with heuristic search. The International Journal of Robotics Research 33: 305-320
16 Chiddarwar SS and Ramesh Babu N (2011) Conflict free coordinated path planning for multiple robots using a dynamic path modification sequence. Robotics and Autonomous Systems 59: 508-518
17 Afaghani AY and Aiyama Y (2014) Advanced-collision-map-based online collision and deadlock avoidance between two robot manipulators with PTP commands. 2014 IEEE International Conference on Automation Science and Engineering (CASE), pp: 1244-1251
18 Rodríguez C, Montano A and Suarez R (2014) Planning manipulation movements of a dual-arm system considering obstacle removing. Robotics and Autonomous Systems 62: 1816-1826
19 Korayem MH and Nekoo SR (2016) The SDRE control of mobile base cooperative manipulators: Collision free path planning and moving obstacle avoidance. Robotics and Autonomous Systems 86: 86-105
20 L. Yang, J Qi, D. Song, J Xiao, J Han, and Y Xia (2016) Survey of Robot 3D Path Planning Algorithms. Journal of Control Science and Engineering, pp:1-22
21 C Tovey, S Greenberg, and S Koenig (2003) Improved analysis of D∗. IEEE International Conference on Robotics and Automation, pp:3371–3378
22 A Nash, S Koenig, and C Tovey (2010) Lazy theta∗: any-angle path planning and path length analysis in 3D. The Third Annual Symposium on Combinatorial Search, pp:153–154
23 L Verscheure, L Peyrodie, N Makni, N Betrouni, S Maouche, and M Vermandel (2010) Dijkstra's algorithm applied to 3D skeletonization of the brain vascular tree: evaluation and application to symbolic. The Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp:3081–3084
24 V A Luchnikov, N N Medvedev, L Oger, and J-P Troadec (1999) Voronoi-Delaunay analysis of voids in systems of nonspherical particles. Physical Review E 59:7205–7212
25 S Choudhury, S Scherer, and S Singh (2013) Rrt∗-ar: sampling based alternate routes planning with applications to autonomous emergency landing of a helicopter. IEEE International Conference on Robotics and Automation, pp:3947–3952
26 S M LaValle (1998) Rapidly-exploring random trees a new tool for path planning. Tech. Rep
27 Kockara S, Halic T, Iqbal K, Bayrak C, Rowe R (2007) Collision detection: A survey. In Proceedings of the 2007 IEEE International Conference on Systems, Man and Cybernetics, pp:4046–4051
28 Jiménez, Pablo, Federico Thomas, and Carme Torras (2001) 3D collision detection: a survey. Computers & Graphics 25.2: 269-285
29 Okada K, M Inaba and H Inoue (2005) Real-time and Precise Self Collision Detection System for Humanoid Robots. IEEE International Conference on Robotics and Automation, pp: 1060-1065
30 S Gottschalk, MC, Lin D (1996) OBBTree: A hierarchical structure for rapid interference detection. 23rd annual conference on Computer graphics and interactive techniques, pp:171–180
31 Gottschalk, S (2000) Collision queries using oriented bounding boxes. PhD thesis, The University of North Carolina at Chapel Hill
32 Corrales, J A, FA Candelas, and F Torres (2011) Safe human–robot interaction based on dynamic sphere-swept line bounding volumes. Robotics and Computer-Integrated Manufacturing 27.1: 177-185
33 Hildebrandt AC, Wittmann R, Wahrmann D, Ewald A, Buschmann T (2014) Real-Time 3D Collision Avoidance for Biped Robots. International Conference on Intelligent Robots and Systems, pp:4184–4190

34 Tsai, Chi-Shen (2014) Online Trajectory Generation for Robot Manipulators in Dynamic Environment—An Optimization-based Approach. PhD thesis, University of California, Berkeley
35 Steinbach K, Kuffner J, Asfour T, Dillmann R (2006) Efficient Collision and Self-collision Detection for Humanoids Based on Sphere Tree Hierarchies. 6th IEEE-RAS International Conference on Humanoid Robots, pp:560–566
36 X Jing, W Guangxin, L Chongyang and L Hong (2016) Real-time collision detection for manipulators based on fuzzy synthetic evaluation. IEEE International Conference on Mechatronics and Automation, pp:777-782
37 Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989) Multilayer feedforward networks are universal approximators. Neural networks 2.5, pp: 359-366
38 Venter, Gerhard, and Jaroslaw Sobieszczanski-Sobieski (2003) Particle swarm optimization. AIAA journal 41.8:1583-1589
39 Dehuri, Satchidananda, and Sung-Bae Cho (2010) A comprehensive survey on functional link neural networks and an adaptive PSO–BP learning for CFLNN. Neural Computing and Applications 19.2:187-205
40 Ren, Chao, et al (2014) Optimal parameters selection for BP neural network based on particle swarm optimization: A case study of wind speed forecasting. Knowledge-Based Systems 56: 226-239
41 Chang, Cheol, Myung Jin Chung, and Zeungnam Bien. Collision-free motion planning for two articulated robot arms using minimum distance functions. Robotica, 1990,8.2: 137-144
42 Spong, M W, Hutchinson, S, Vidyasagar, M. Robot Modeling and Control. Wiley, New York, 2005
43 Zhu C, Zhang J, Liu Y, et al (2020) Comparison of GA-BP and PSO-BP neural network models with initial BP model for rainfall-induced landslides risk assessment in regional scale: a case study in Sichuan, China. Nat Hazards 100:173–204
44 Peram, Thanmaya, Kalyan Veeramachaneni, and Chilukuri K Mohan (2003) Fitness-distance-ratio based particle swarm optimization. IEEE Swarm Intelligence Symposium, pp: 174-181
45 J Kennedy and R Eberhart (1995) Particle swarm optimization. International Conference on Neural Networks, pp:1942-1948
46 Y Shi and R Eberhart (1998) A modified particle swarm optimizer. IEEE International Conference on Evolutionary Computation Proceedings, IEEE World Congress on Computational Intelligence, pp:69-73
47 Ying-tang, WANG Wei ZHANG (2007) Analysis and improving way of BP ANN in predicting time series data. Computer Engineering and Design,

Figures

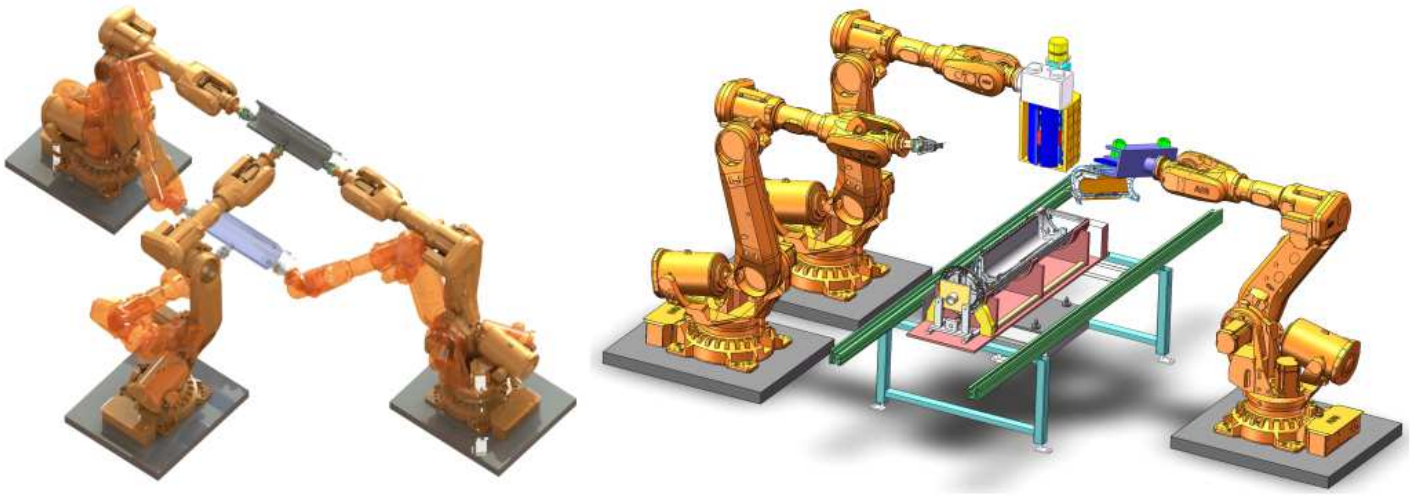


Figure 1

The schematic diagram of multi-manipulator cooperate and collabo-rate manufacturing system. Multi-manipulator collaborate manufac-turing system (Left): the case of multiple manipulators with shared workspace, simultaneous manufacturing tasks and common object; Multi-manipulator cooperate manufacturing system (Right): the case of multiple manipulators with shared workspace, non-simultaneous tasks and separate object;

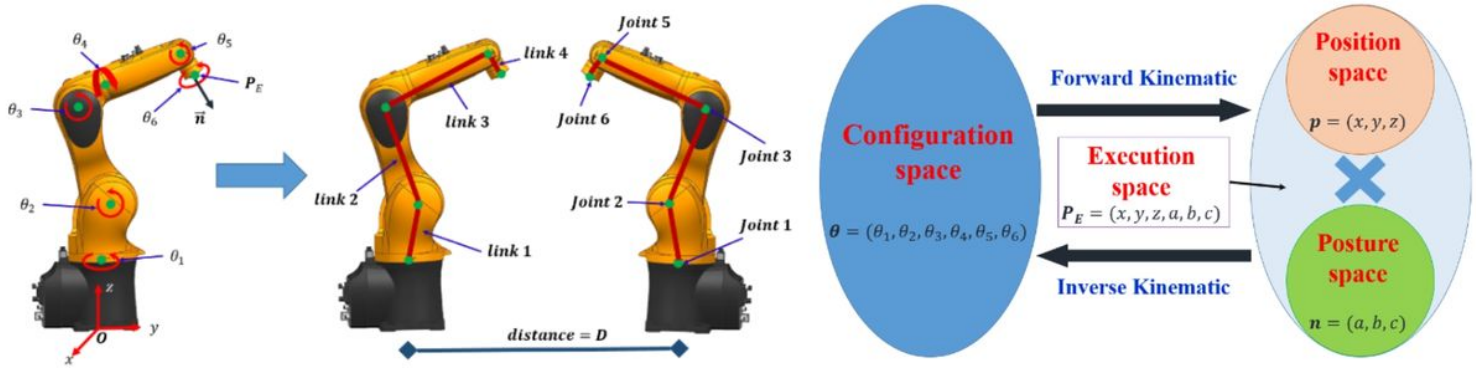


Figure 2

Modelling of a 6-DOF industrial manipulator in (a): Oxyz is the base coordinate system of the manipulator; n is the direction vector of the central axis of the sixth joint; P_E is the position coordinate vector in Oxyz. All position coordinates that the manipulator EEF can reach within the space set are gathered and designated as the position space. n=(a,b,c) is the direction vector of the central axis of the sixth joint, which can also be used to represent the pose of manipulator EEF. All possible direction vectors that manipulator EEF can reach are collected in a set, referred to as the posture space. The combination of the position space and posture space constitutes the execution space. The value of each joint can be denoted as θ_i (i=1~6). Then θ=(θ_1,θ_2,θ_3,θ_4,θ_5,θ_6) can be recognized as a point in the 6-DOF configuration space (C-space). The relations between the execution space, C-space, position space, and

posture space are shown in (b). Every point in the C-space can determine a only point in execution space via forward kinematic calculation, but every point in execution space can correspond to more than one point in the C-space for the inverse kinematic calculation of 6-DOF manipulator has multiple solutions.

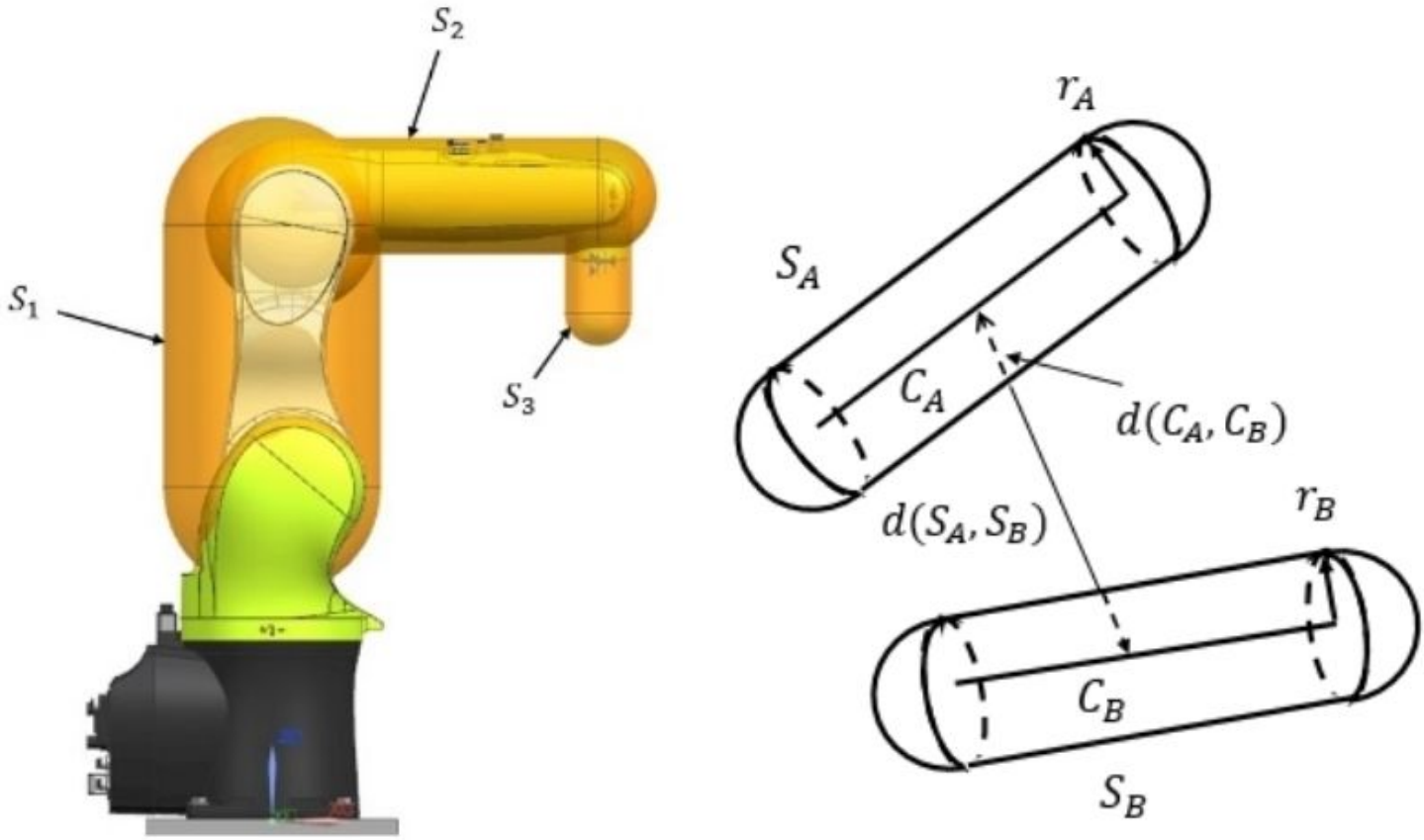


Figure 3

Left: Manipulator model approximated by sphere-swept volume. Right: Minimum distance of two capsule models

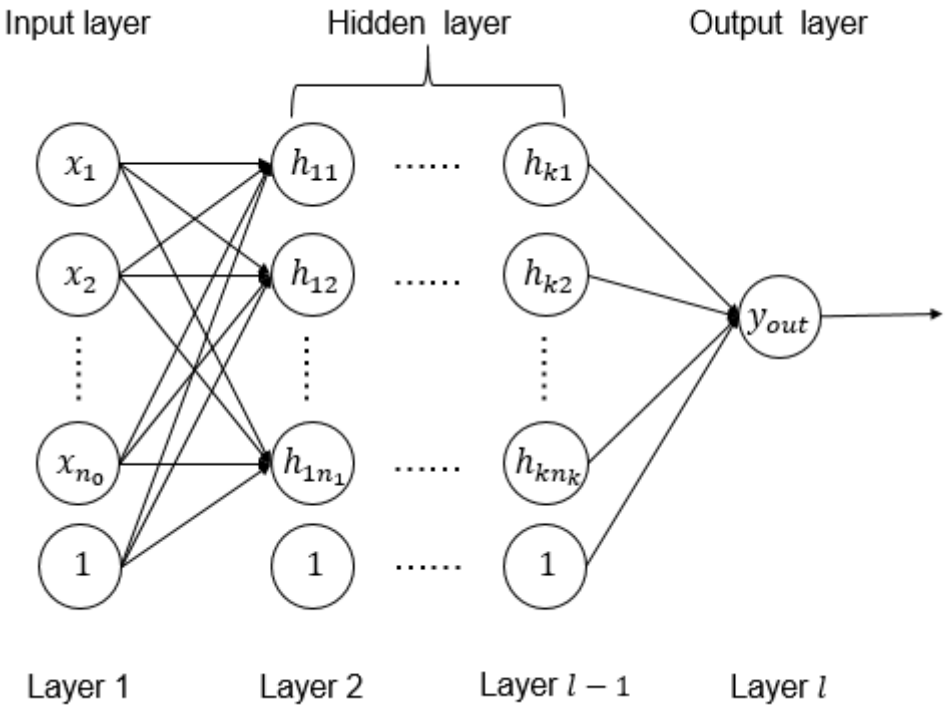


Figure 4

Model of BP neural network

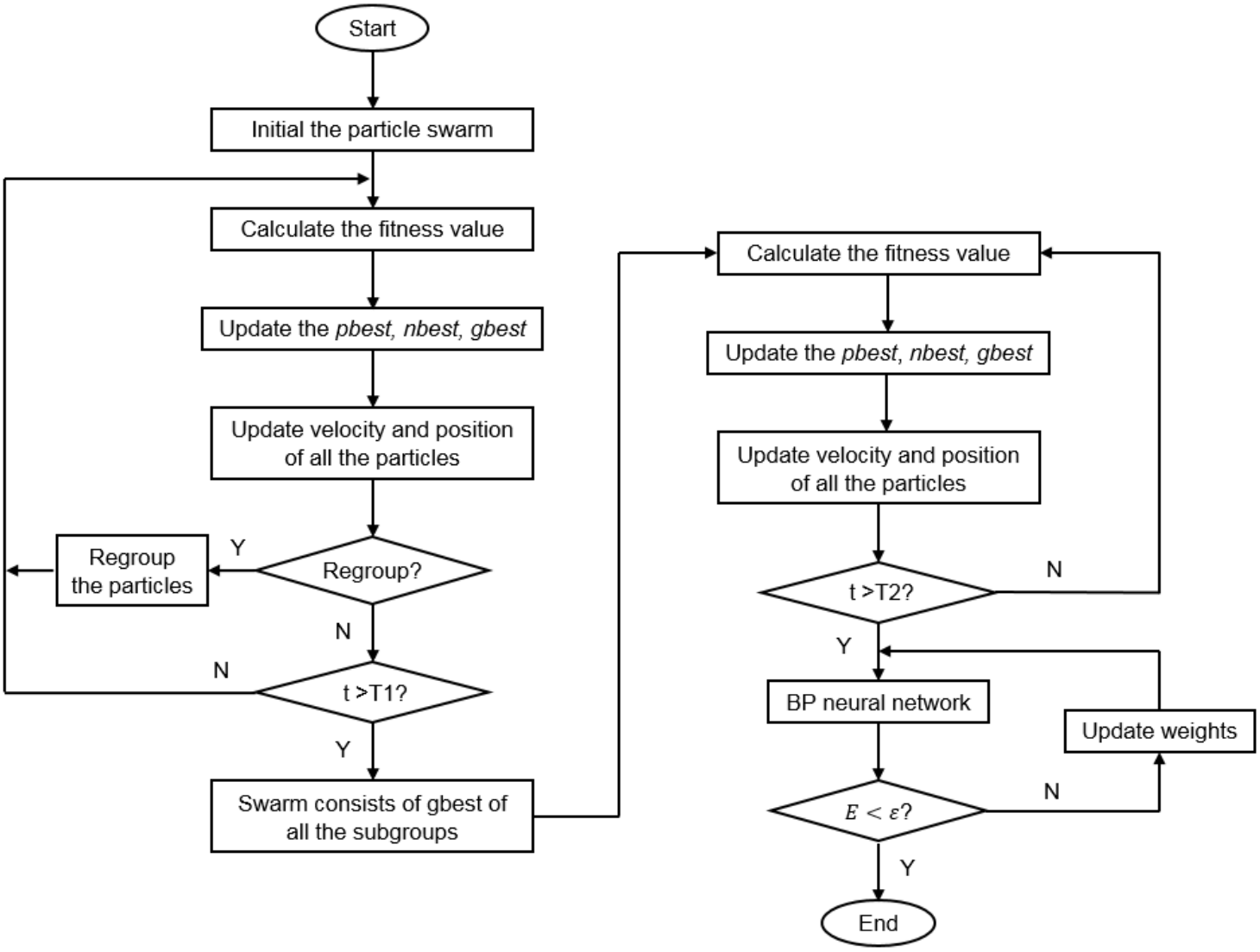


Figure 5

Flowchart of PSO-BP neural network for collision detection

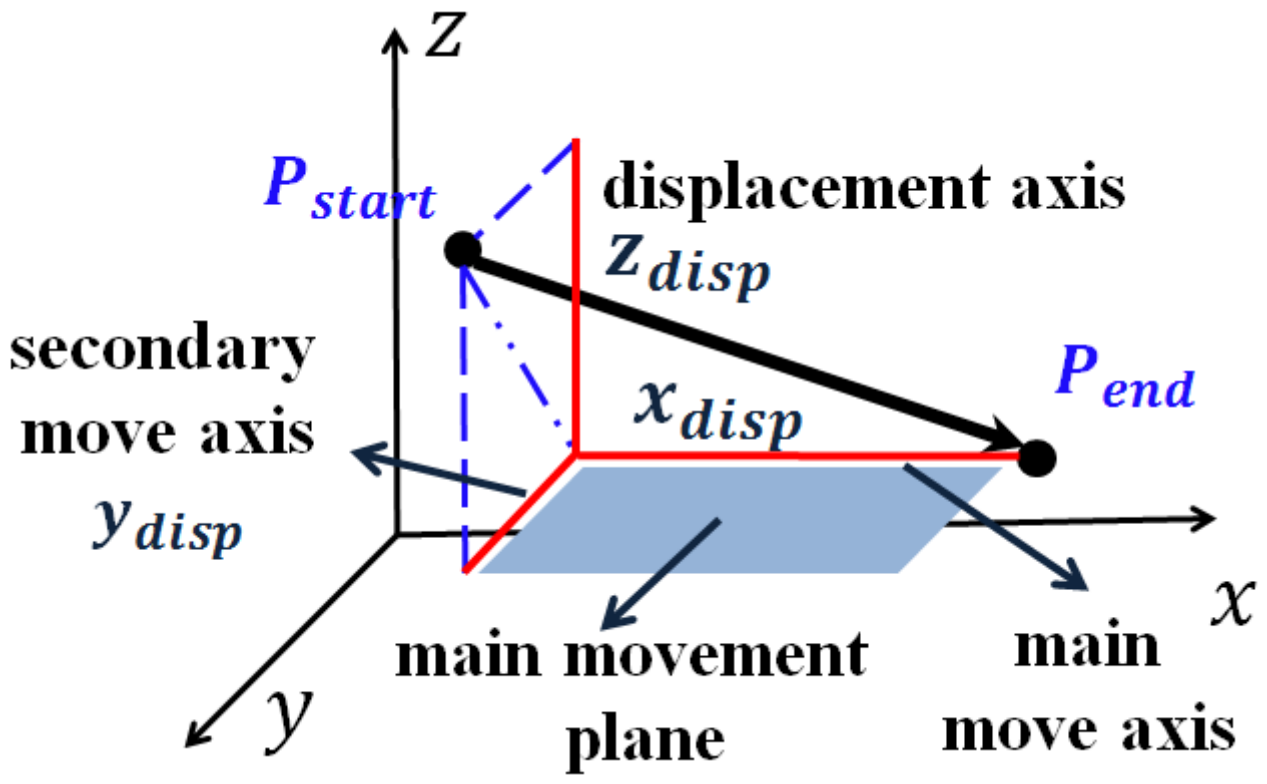


Figure 6

Diagram of displacement axis, secondary movement axis, main movement axis, and main movement plane of a moving path from P_start to P_end under oxyz coorsinate system.

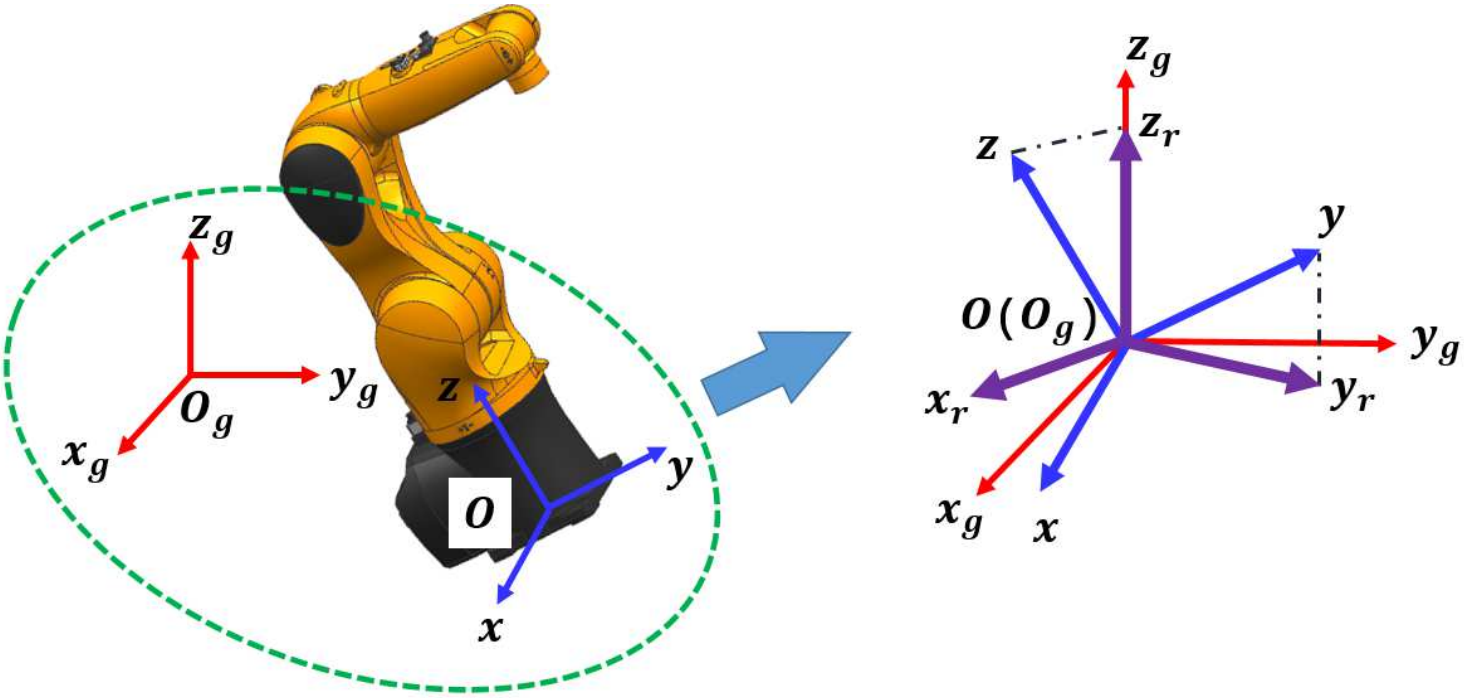


Figure 7

Diagram of building up the three reference direction vectors via coordinate system base method. O_g x_g y_g z_g is the global coordinate system as well as Oxyz is the base coordinate system. The posture of base coordinate system is arbitrary. Take the scenario of Fig.4 for example, the reference direction vector z_r is the projection vector of z axis on the displacement axis z_g, y_r is the projection vector of y axis on the main movement plane O_g x_g y_g, and x_r is close to x axis and perpendicular to both the reference direction vector z_r and y_r

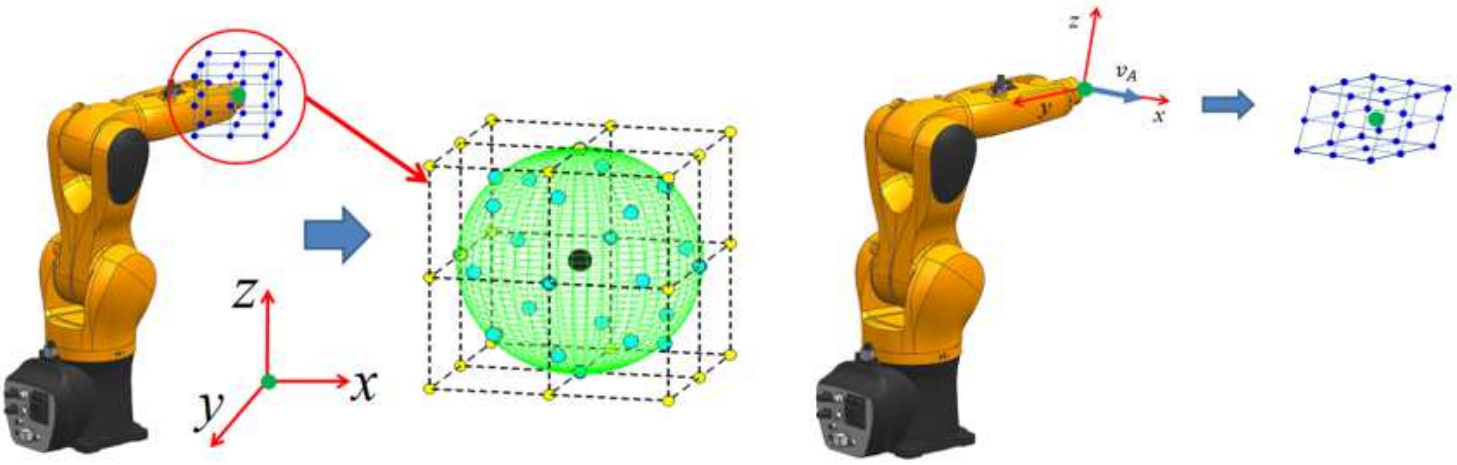


Figure 8

Candidate positions of single manipulator EEF in Cartesian space. The left hand side is a diagram of candidate positions as determined by coordinate system based method; and the right hand side are the candidate positions determined using the velocity based method.

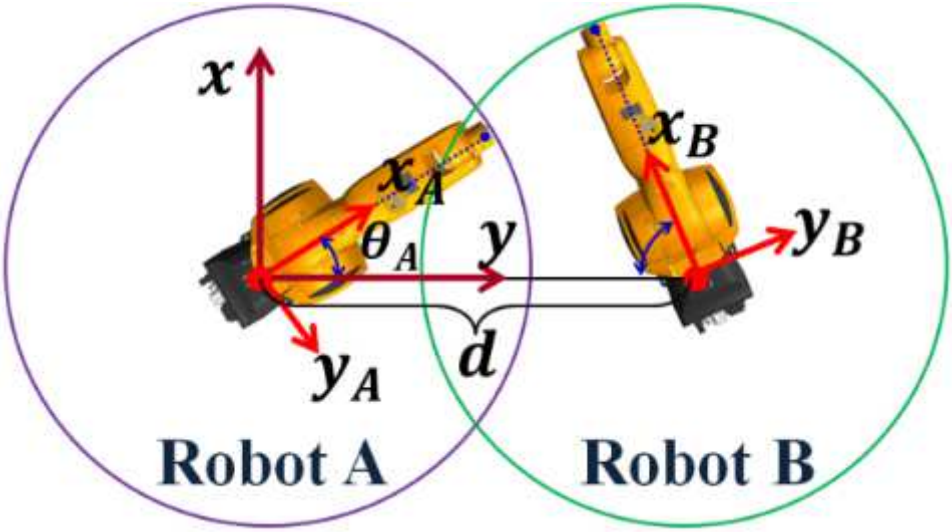


Figure 9

Diagram of universal dual-manipulator cooperation situation

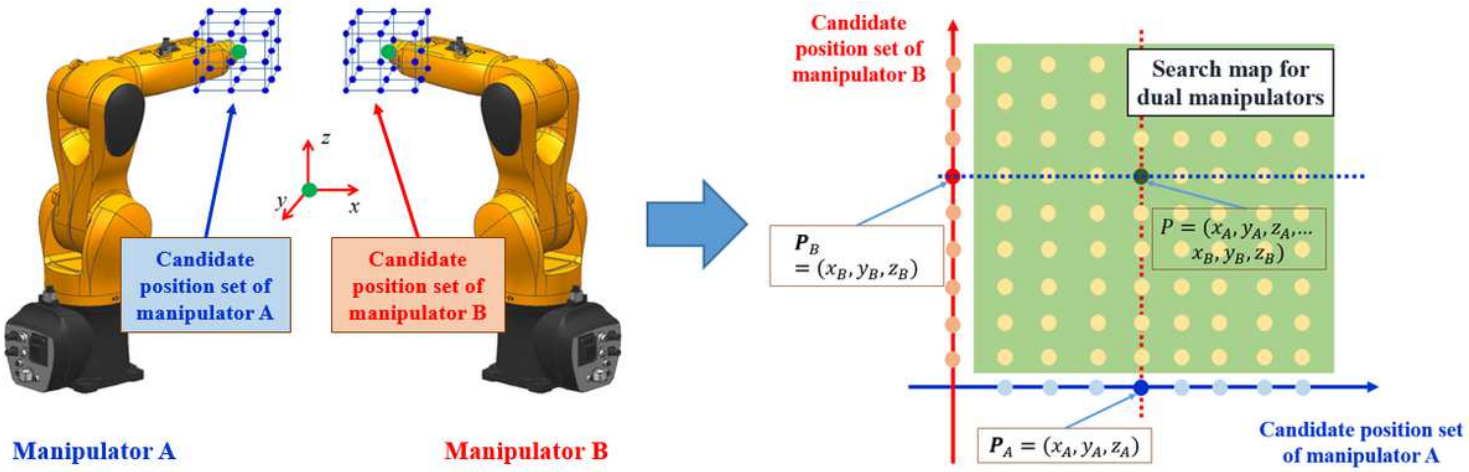


Figure 10

Relation diagram between set of candidate positions and PSSM of dual-manipulator system. The left hand side is the diagram of candidate positions of the two manipulators, and the right hand side is the diagram of PSSM for dual manipulators. In PSSM, the candidate positions of manipulators A constitute the discrete points on horizontal coordinate and those of manipulators B constitute the discrete points on the vertical coordinate.

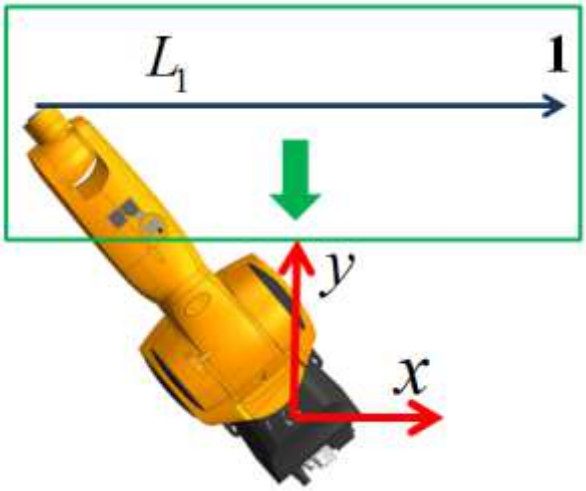


Figure 11

Movement in the main movement plane of manipulator A

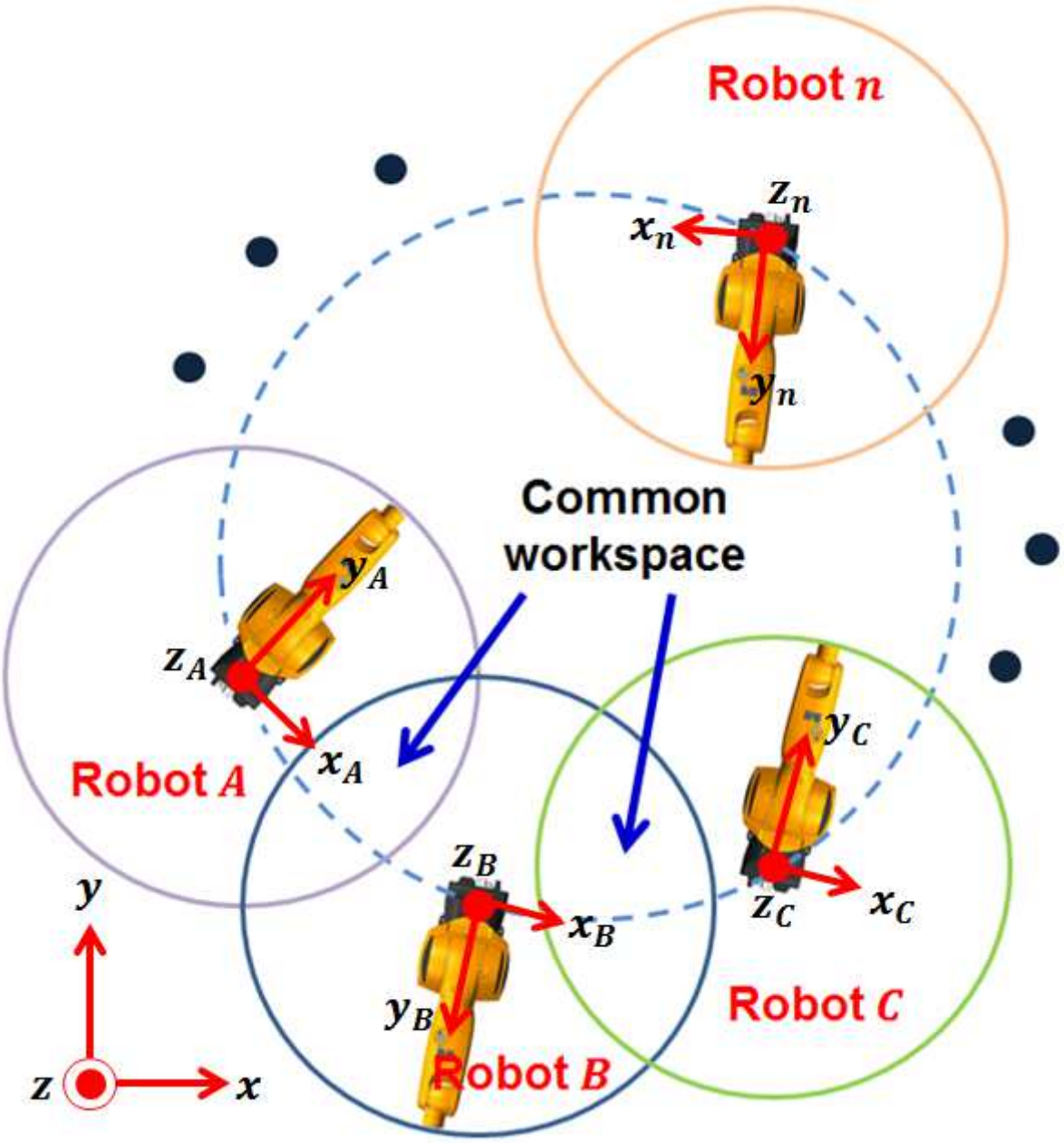


Figure 12

Schematic of position and coordination transformation relationships for cooperation of multiple manipulators. Note that the base coordinate of each manipulator can be arbitrary, and we do not require that all the workspaces of the manipulators overlapped

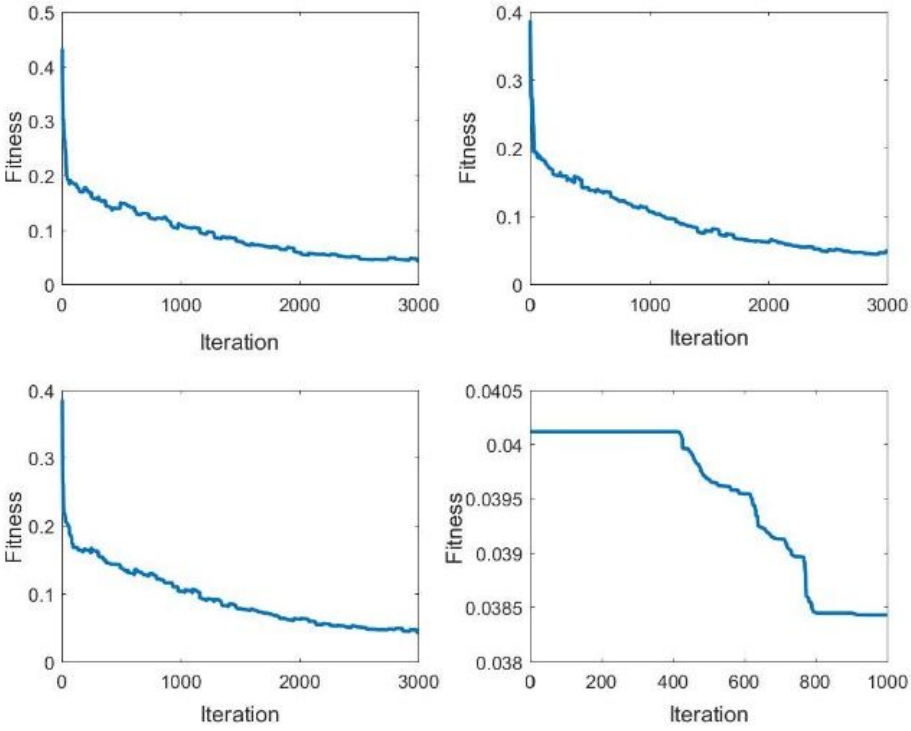


Figure 13

Fitness-iteration of the PSO. (a)(b)(c) correspond to three subgroups in the first optimization stage and (d) is the second optimization stage of the hybrid method. All of the fitness values are average of the ten trials.

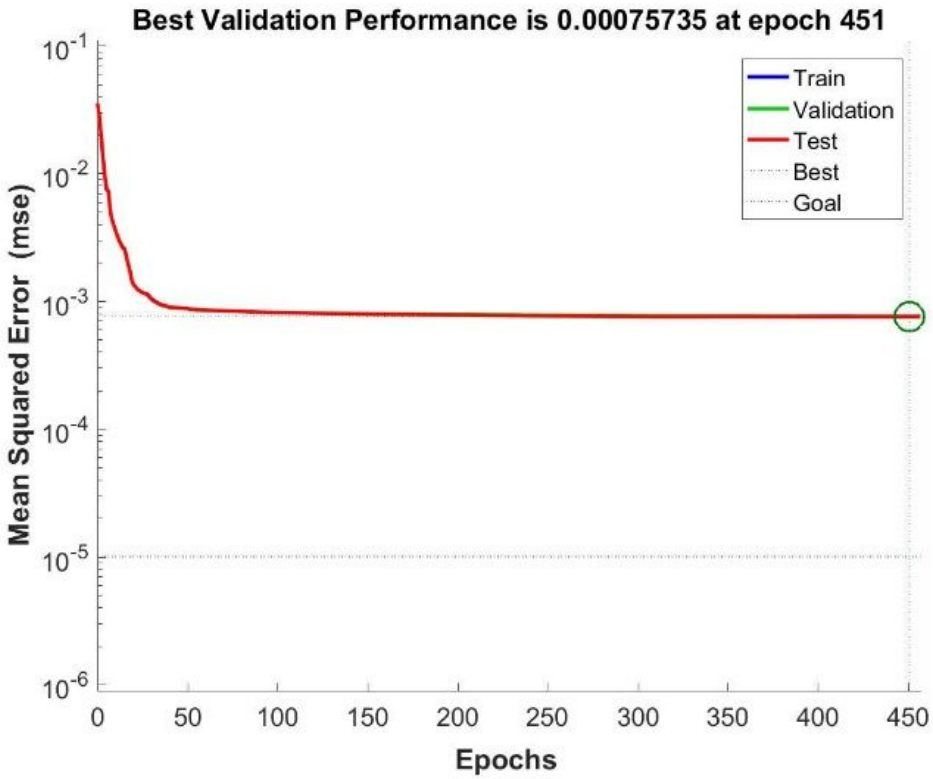
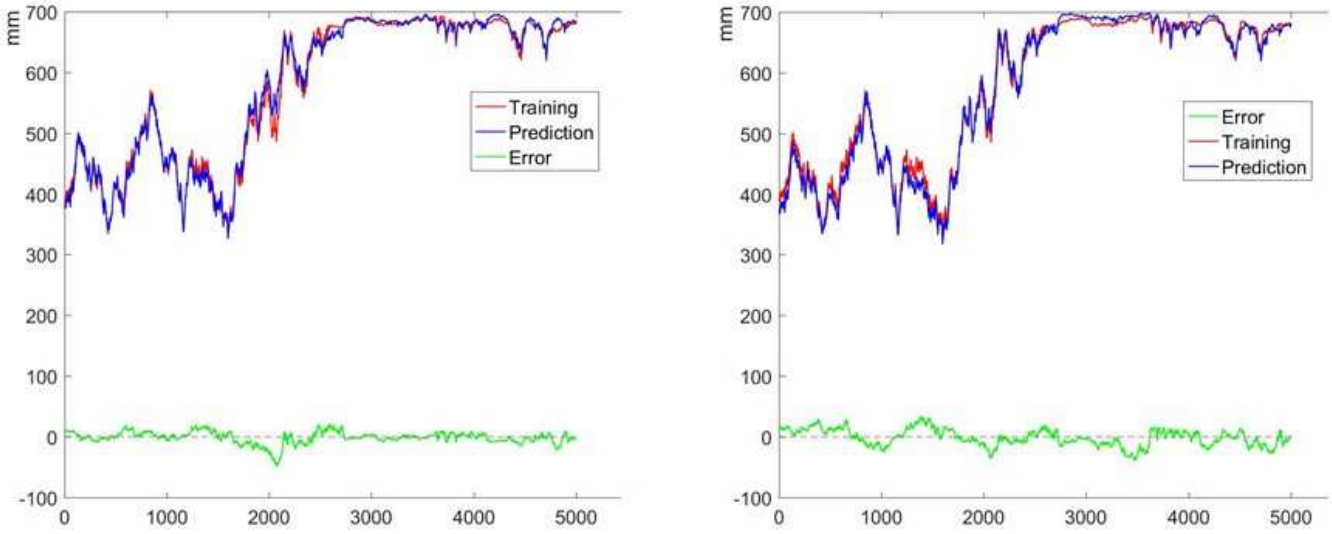


Figure 14

Training of BP neural network



a. Prediction error on test set of PSO-BP hybrid method

b. Prediction error on test set of BP neural network

Figure 15

Prediction error on test set of PSO-BP method and BP neural network

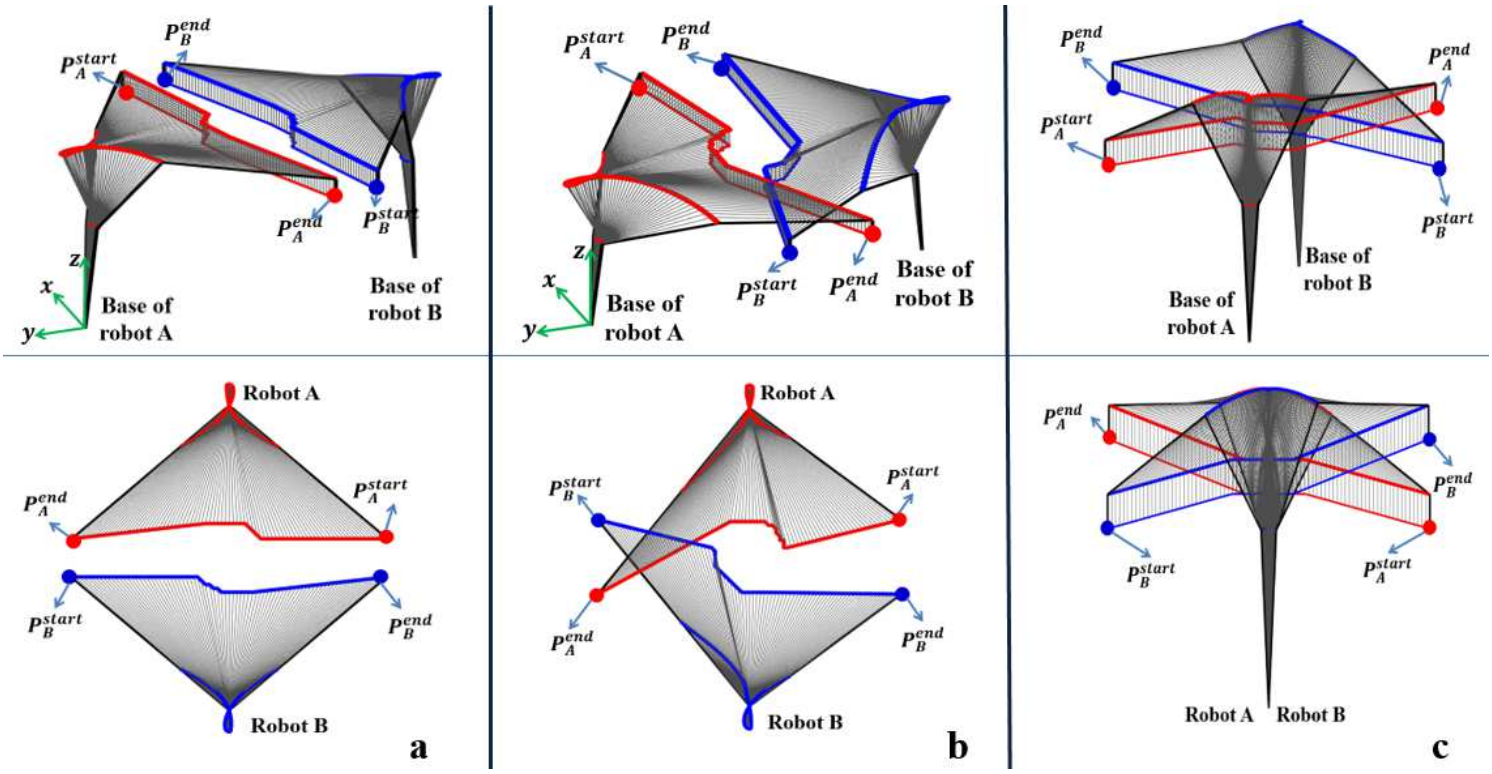


Figure 16

Simulation results of path planning for dual-manipulator cooperation using coordinate system based method where a, b and c correspond to the simulation results under scenarios a, b, and c, respectively. The paths of each joint of A and B in Cartesian space are represented by the red and blue curves, respectively. The upper images show the 3D views, and the lower images are the top view.

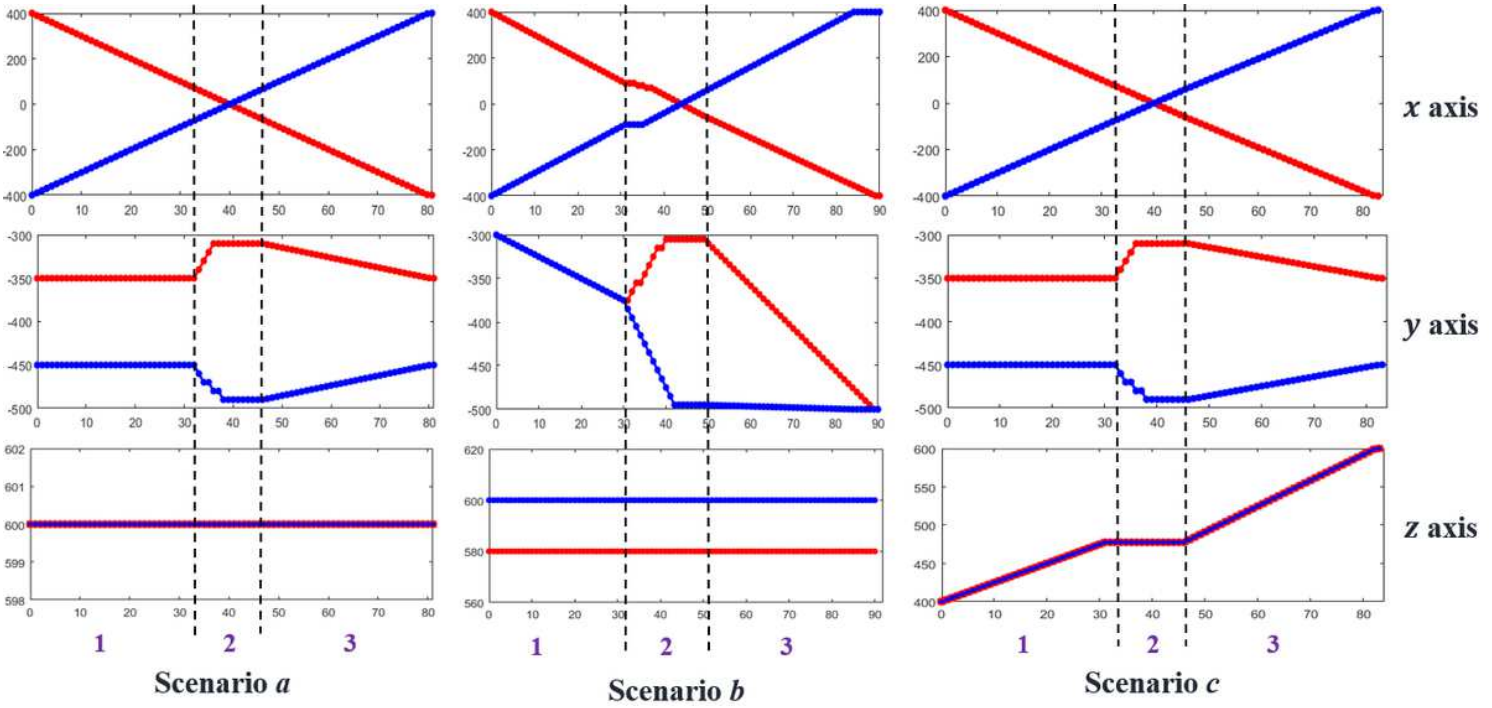


Figure 17

Movements along x-, y-, and z-axis of manipulators A and B for scenarios a, b, and c using coordinate system based method. The red and blue curves represent the movements of manipulators A and B, respectively. For each image, the horizontal axes represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

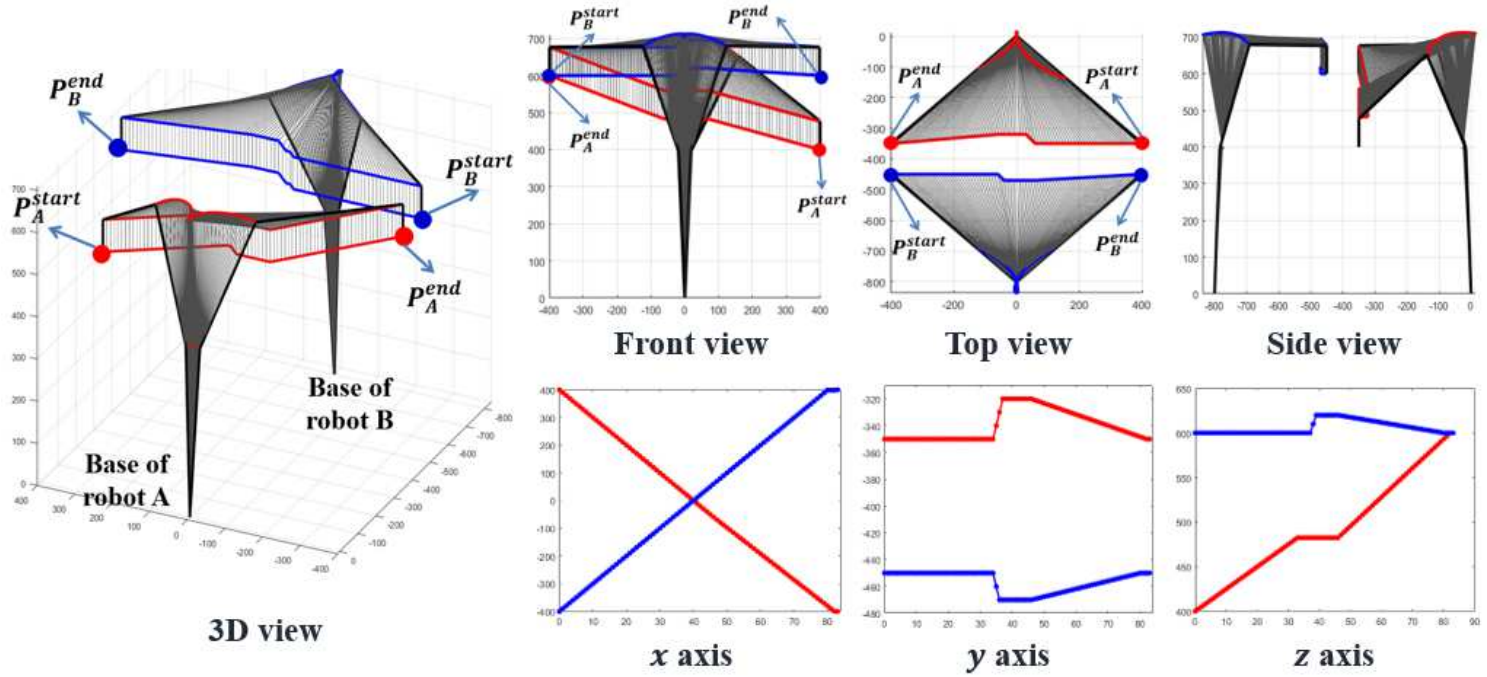


Figure 18

Results of path planning simulation and movements along x-, y-, and z-axis for dual-manipulator cooperation under scenario d via coordinate system based method

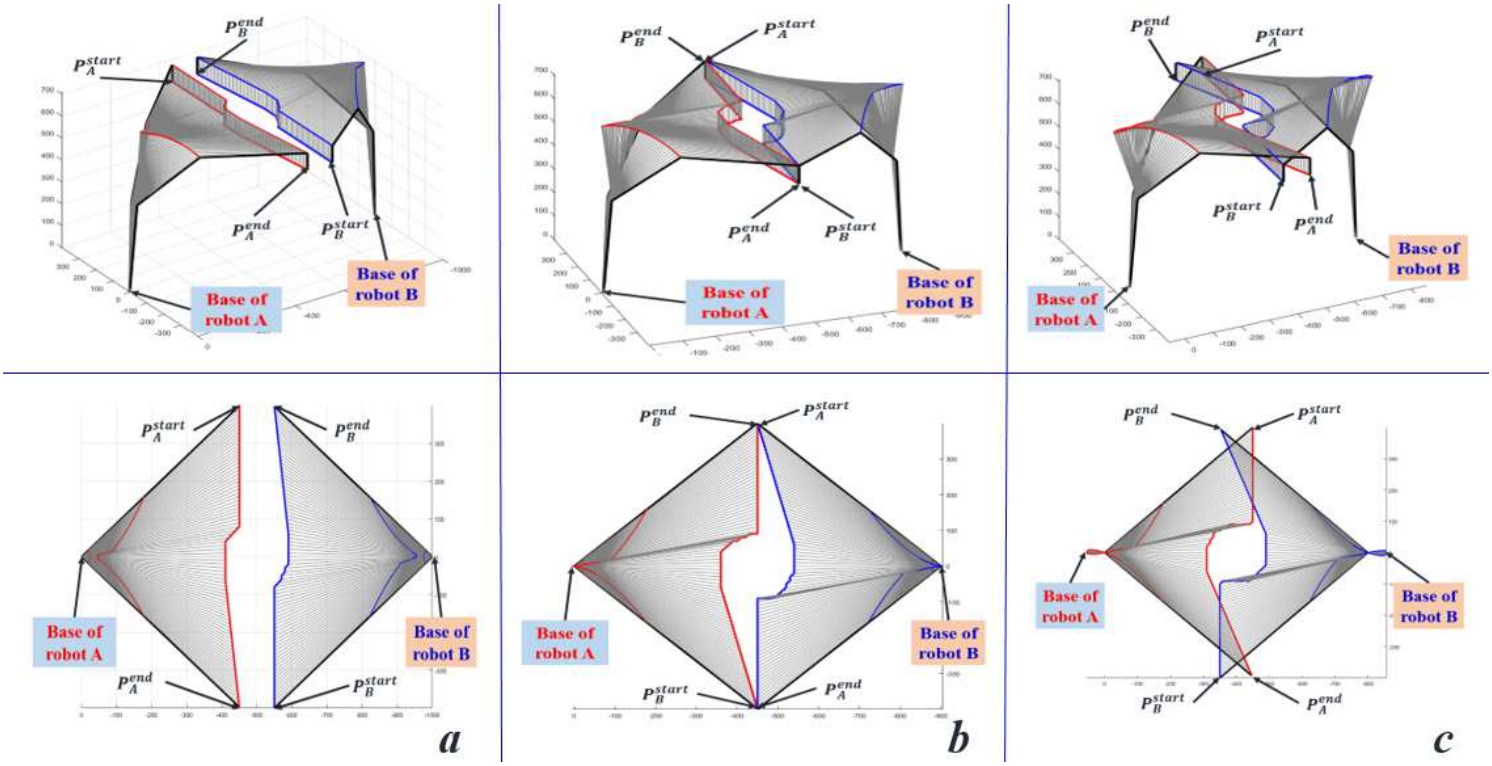


Figure 19

Results of path planning simulation for dual-manipulator cooperation based on coordinate system based method with decreasing distance between the bases of two manipulators, where a, b and c are the simulations with distances of 1000 mm, 900 mm and 800 mm, respectively.

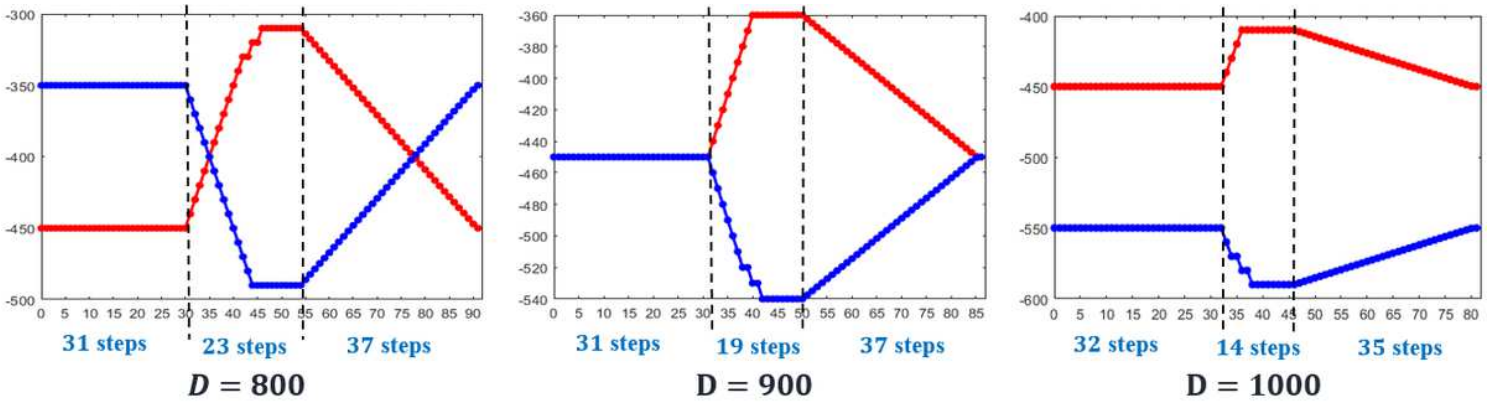


Figure 20

Simulation results of movements along axis y in the scenarios of Fig.19. The red and blue curves represent the movements of manipulators A and B, respectively. For each image, the horizontal axes represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

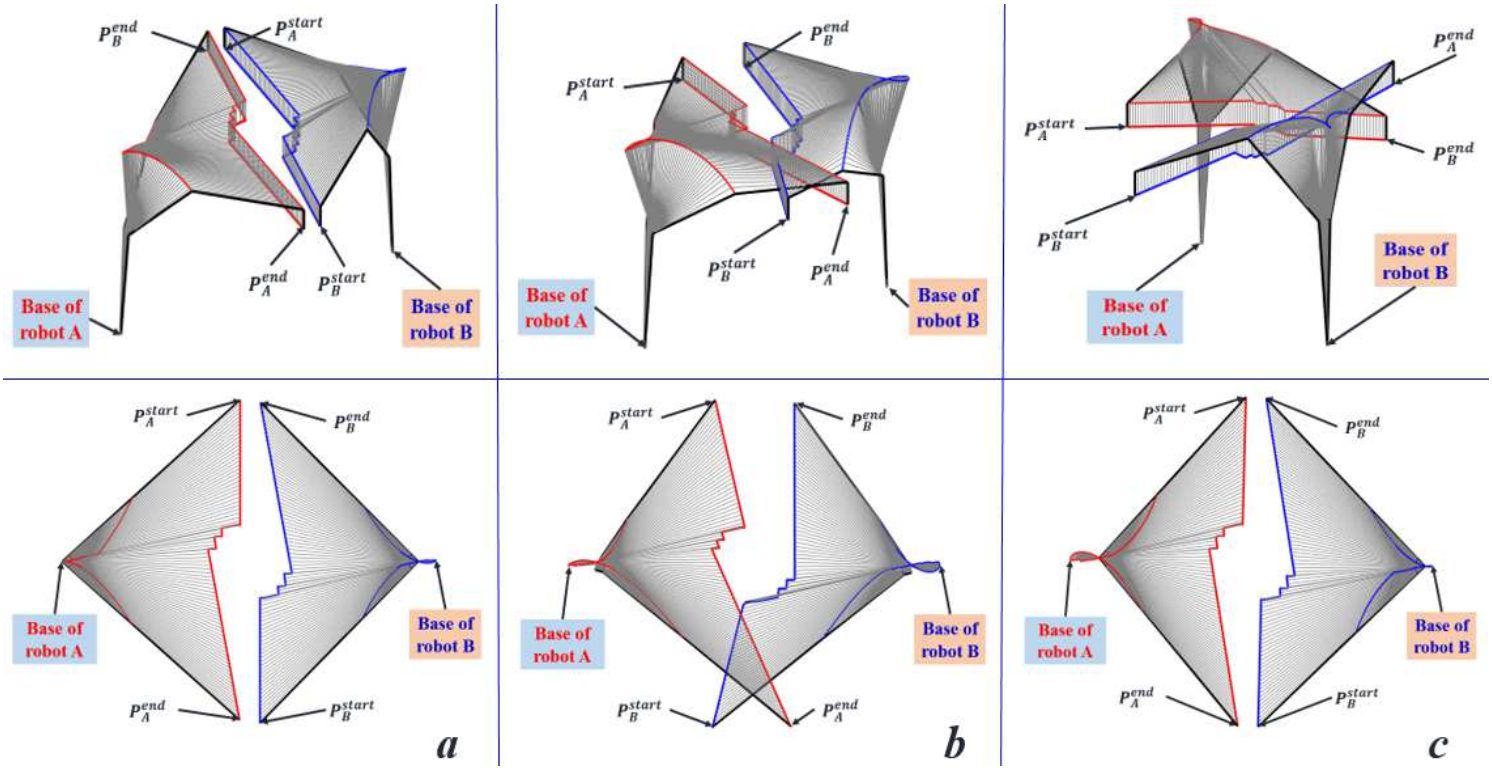


Figure 21

Results of path planning simulation for dual-manipulator cooperation using velocity based method where a, b and c correspond to results for scenarios a, b and c, respectively. The paths of all joints in A and B in Cartesian space are represented by the red and blue curves, respectively. The upper images show 3D views of each scenario, while the lower images are the top view.

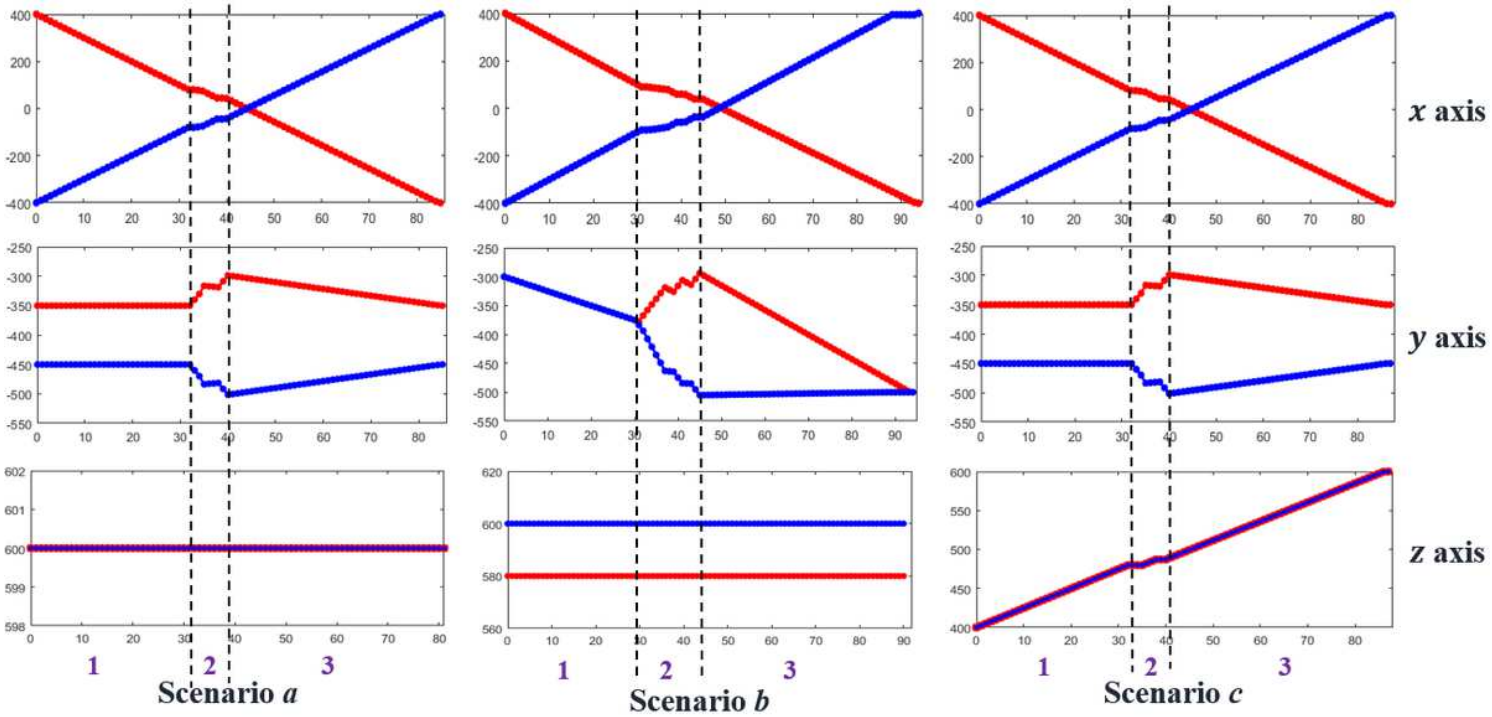


Figure 22

Movements of manipulators A and B along x-, y-, and z-axis for scenarios a, b and c using velocity based method. The red and blue curves represent the movements of manipulators A and B, respectively. For each image, the horizontal axes represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

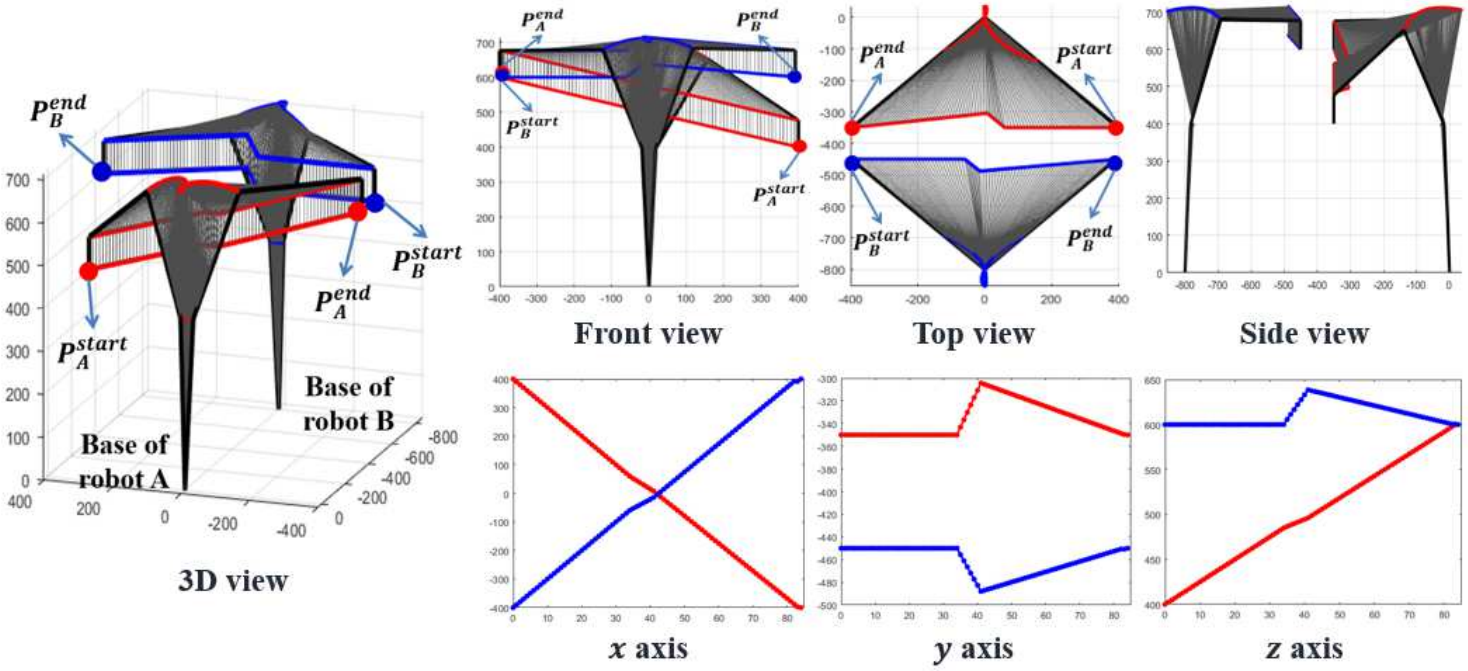


Figure 23

Results of path planning simulation and movements along x-, y-, and z-axis dual-manipulator cooperation under scenario d via velocity based method

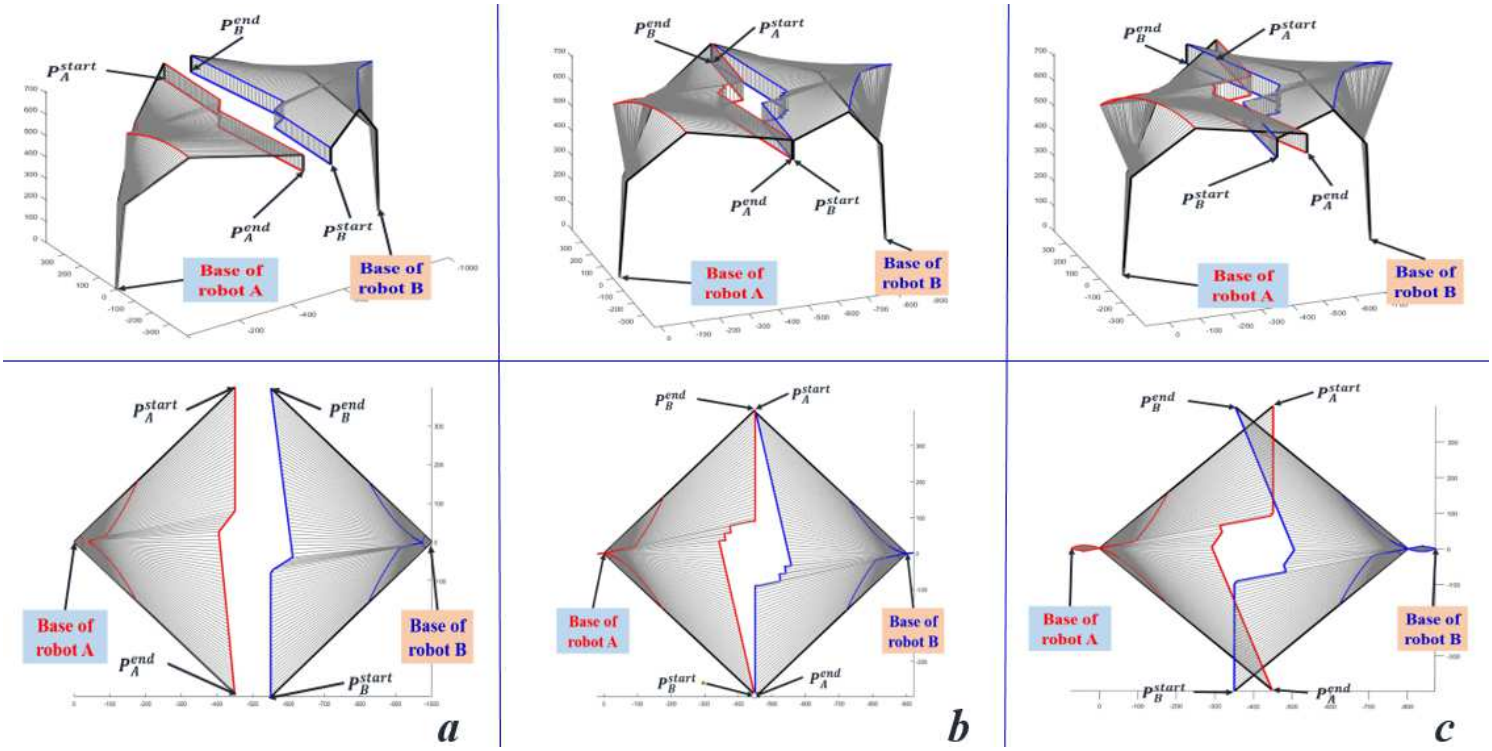


Figure 24

Results of path planning simulation of dual-manipulator cooperation using velocity based method with decreasing distance between the bases of two manipulators where a, b and c represent the simulation results with distances of 1000 mm, 900 mm and 800 mm, respectively.

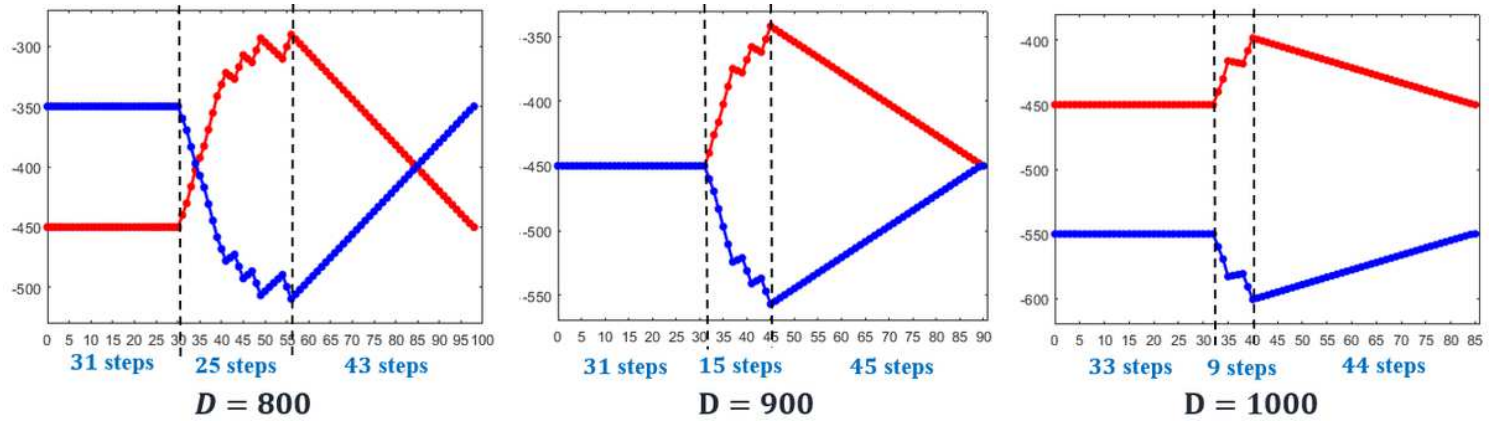


Figure 25

Simulation results of movements along axis y in the scenarios of Fig.19. The red and blue curves represent the movements of manipulators A and B, respectively. For each image, the horizontal axes

represent the step number of the whole path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.

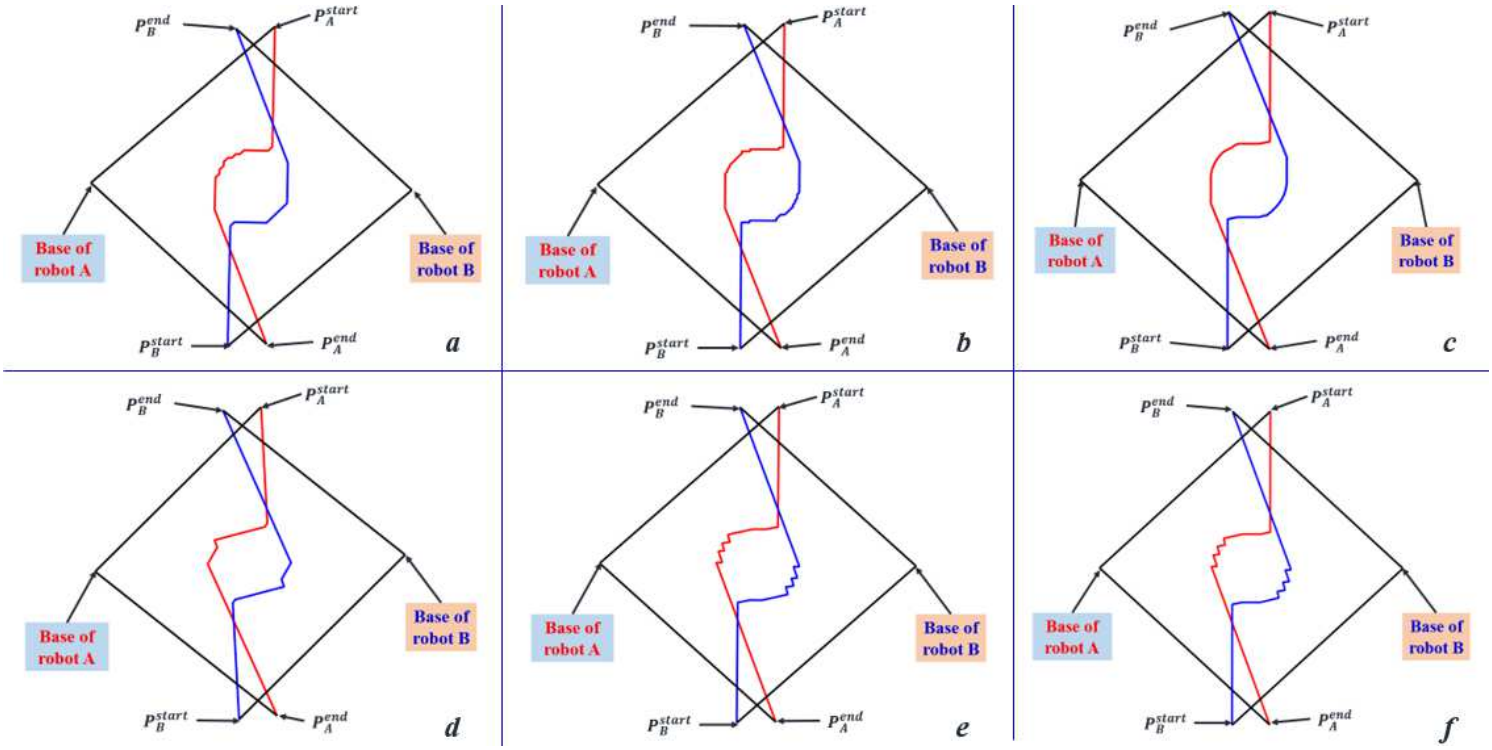


Figure 26

Simulation results for dual-manipulator cooperation using PSMS method with decreasing step lengths, (a) 10 mm, (b) 5 mm, and (c) 1 mm for the coordinate system based method and (d) 10 mm, (e) 5 mm, and (f) 1 mm for the velocity based method.

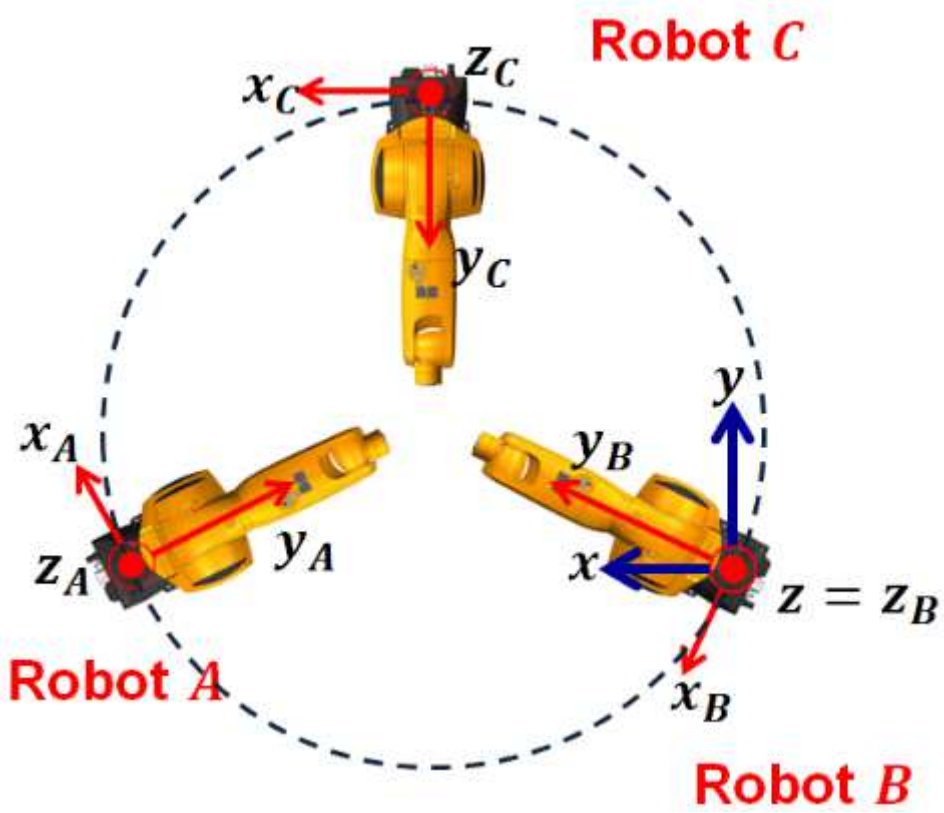


Figure 27

Schematic of scenario for triple-manipulator cooperation

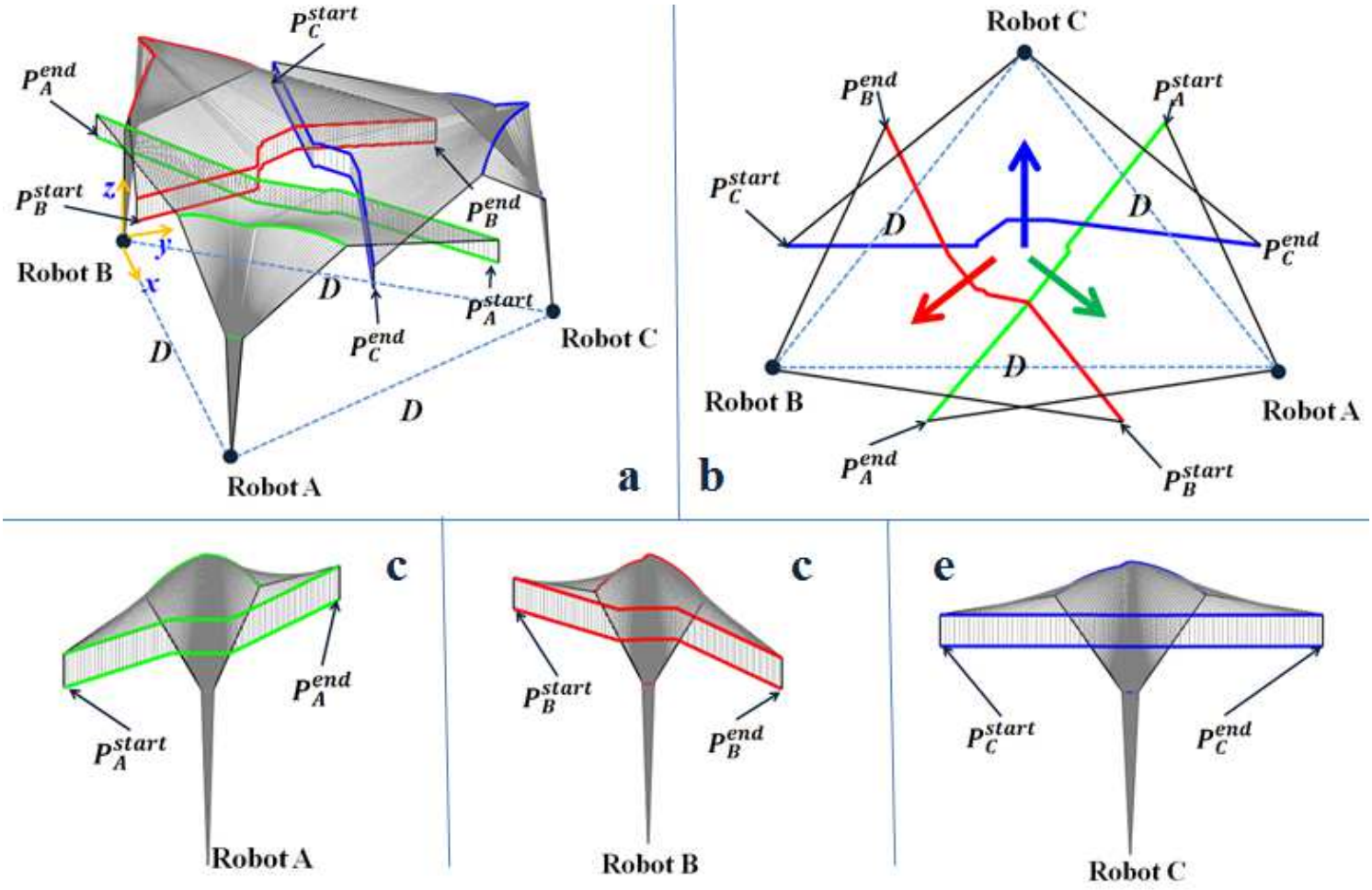


Figure 28

Results of triple-manipulator cooperation simulation using PSMS method: (a) 3D view; (b) overview of paths; (c-d) side view of paths for each manipulator A, B, and C, respectively. The green, red, and blue curves represent the paths of all joints of manipulator A, B, and C, respectively, in Cartesian space.