A novel non-collision path planning strategy for multi-manipulator cooperative manufacturing systems

Chang Su1 · Jianfeng Xu1

Received: 26 March 2021 / Accepted: 25 January 2022 / Published online: 3 March 2022
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2022

Abstract
Analogous to the definition of human–robot interaction, the case of multiple manipulators with shared workspace, asynchronous manufacturing tasks, and independent objects is named as a multi-manipulator cooperative manufacturing system. Multi-manipulator cooperation is increasingly used in modern industrial manufacturing systems and requires collision-free path planning as a vital issue in terms of safety and efficiency. This study proposes a novel method called Sampling-based Operation Space Map Search method, which combines a map search method with a time-sampling-based method. Two candidate position determination methods are proposed to establish local planning maps for all manipulators individually during each sampling time interval. After a specific search map simplification process, the optimal local path fragments within each sampling period can be determined by the map search method. Then, all chronologically planned path segments can be glued together to generate collision-free paths for all manipulators in a multi-manipulator cooperative manufacturing system. The simulation results demonstrated that the proposed strategy could successfully achieve collision avoidance of dual manipulator system whilst meeting the real-time requirements for cooperative assembling scenarios. Compared with the conventional map search method, this proposal is highly effective at the cost of forgoing the global optimum. Further satisfactory simulation results for triple manipulators indicate that our algorithm can be extended to multi-manipulator cooperative manufacturing applications.

Keywords Multi-manipulator cooperative manufacturing system · Non-collision path planning · Sampling-based operation space map search

1 Introduction

1.1 Background
At present, six-degree-of-freedom (6-DOF) industrial manipulators play an increasingly important role in automated manufacturing due to numerous advantages, including the provision of tireless repetitive labor, faster-moving speed, and higher-accuracy performance [1]. Thus, tasks requiring numerous workers can undoubtedly be accomplished cooperatively using multiple manipulators, indicating that multi-manipulators not only work side-by-side, but also as dyads and teams. Analogous to the definition of human–robot interaction [2, 3], the case of dual or multiple manipulators manufacturing system can be divided into collaboration and cooperation according to the arrangement of the tasks (Fig. 1). For the case of a collaborative manufacturing system, all manipulators together with the executed mechanism constitute a complete multi-DOF closed-loop or parallel manufacturing system. Thus, the non-collision manufacturing control strategy under such circumstances can be transformed into the currently mature internal obstacle avoidance strategy [4]. However, for the case of multiple manipulators cooperative manufacturing system (e.g., assembly as shown in the right side of Fig. 1), problems and limitations are encountered in the small overlapping workspaces that accommodate numerous cooperative manipulators, resulting in collisions if no related countermeasures are appropriately put in place. Under this circumstance, a path planning algorithm that can circumvent the constraints of the mechanical structure simplifies the planning space, and determines the paths of all manipulators involved in the cooperative system in real-time constitute the kernel.
A 6-DOF industrial manipulator implies a type of serial chain robot with six revolving joints, and its structure is much more complex than that of a mobile robot. A single manipulator can be modeled as a particle in its own configuration space (C-space); therefore, the particle-based path planning methods [5] can be directly used for single manipulator path-planning. Till date, limited research progress has been made for non-collision path planning of multiple cooperating manipulators in C-space. Obstacles were modeled as simple geometries by Jia et al. [6], which were then projected into the C-space of a manipulator. However, this method is not effective for multiple manipulators due to their complex structures. Further, Yu and Wang [7] divided the C-space into multiple layers based on specific dimensions and implemented a rapidly created C-space grid map on each layer. However, owing to the massive number of elements in the maps, the entire algorithm tends to be time-consuming. Li et al. [8] planned collision-free paths for two-dimensional (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. Nonetheless, the method was computationally too expensive for real-time applications, and it was difficult to project a 6-DOF manipulator body into the C-space of another 6-DOF manipulator. Harada et al. [9] 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 trajectory arrangement and movement order of each arm to allow a target object to pass from starting point to endpoint. However, the possibility of collision between the two working arms was not considered.

Therefore, how to explore a simple, effective, and reliable execution space path planning algorithm has become one of the more enthusiastic research focuses in recent decades. An online collision-free trajectory generation algorithm for dual-arm robots was established by Lee et al. [10] 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 was extended due to difficulties setting up the VRM, and raising the method to multiple robots was found to be challenging. Larsen et al. [11] divided the working area into different zones and set up obstacle models within each zone. A master–slave method was then used to select a free path; however, dynamic environments could not be accommodated since the obstacle models were required to be static. Cohen et al. [12] combined the C-space with execution space to build a “motion space.” They used the Lazy Weighted A* method to plan a non-collision path in an environment containing N-manipulators. Still, only path planning for one robot was allowed at a time, and the planning process was performed offline. Although 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. It did not consider the coordination of different tasks.

Additional methods have been proposed for the non-collision path planning of multiple manipulators. For example, Chiddarwar and Ramesh [13] used the A* method to plan a collision-free path in C-space without considering the motion of other robots, and a path modification sequence (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 and Aiyama [14] used a collision map to detect the collisions between two robots to avoid deadlocks, which could occur if one robot became an obstacle to another. Unfortunately, the method simply delays the movement of the robot to avoid collision. Rodríguez et al. [15] suggested an approach based on a variation of the probabilistic road map, called the probabilistic road map with obstacles. The method does not exclude collision samples with removable objects, instead classifies them as collided obstacle(s) and allows the search for accessible paths, highlighting which objects must be removed from the workspace to make a valid path. This approach removes any obstacles along the path of the working manipulator; however, it does not enable the manipulator to actually bypass obstacles. Habinejad and Nekoo [16] used the artificial potential field 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-stated review that when it comes to the path planning of dual or multiple industrial manipulators, the negative influence, due to the above-mentioned characteristics such as model
complexity, high dimensionality, and complex planning space, becomes 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 [17]. Map search-based method [18–20] is a global optimization algorithm that costs colossal time planning map construction. Comparatively, the time-sampling-based method [21–23] divides the entire 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.

2 Previous studies and paper organization

For the scenario of multi-manipulator cooperative manufacturing, the real-time online paths replanning due to the prospective collision must be carried out within the shared working space of all manipulators. Therefore, the simplest and the most intuitive solution is to discretize the shared working space and use the A* algorithm for path planning of all manipulators individually (Fig. 2a). Although it involves sensitive principle and globally optimal path selection, this method does not accommodate real-time path planning in irregular motion tempo due to the time-consuming process of constructing discrete global maps.

In this study, a novel method called Sampling-based Operation Space Map Search (SbOSMS) method, which combines the map search method with the time-sampling-based method, was proposed. Equivalently, a local planning map was established during each time interval, and an optimal local path was determined in accordance with the local planning map. All chronologically planned path segments were glued together to generate a collision-free path, which is highly effective at the cost of forgoing the global optimum.

The first step in path planning is manipulator modeling and collision detection. In this study, the spheres-swept volume method [24] was used as the robot modeling method for its simplicity, ease of calculation, and real-time performance capabilities. Moreover, the space algebraic geometry method [25] was selected for collision detection based on its quick and easy implementation. Subsequently, two different candidate position determination methods as well as a specific search map simplification (SMS) process are elaborated to establish the individual local planning maps for all manipulators. After the map search algorithm–based path planning procedure, the remaining study involved the repetition of the same operation during each sampling time interval until all manipulators reached their targets.

In this paper, a brief review of path planning method of multiple manipulator cooperation is presented in Sect. 1. Section 2 provides a concise snapshot of our initial efforts on this issue and the limitations we encountered. Moreover, the overall architecture and algorithmic flow of this study are also summarized. Sections 3, 4 and 5 present the explicit content of the SbOSMS method. In Sect. 3, determination of two candidate position point selection methods is presented for a single manipulator, providing the nodes of the local search map for each cooperative manipulator. Section 4 describes the SMS method for dual and multiple manipulators. Section 5 introduces the formulae for the cost function as an evaluation index of the optimal path. Section 6 presents the simulation results and relevant analyses. Finally, in Sect. 7, conclusions are presented.

---

Fig. 2 Schematic illustration of path planning of the dual-manipulator system: a discretization of the shared working space and use of the A* algorithm for path planning of all manipulators individually and b Sampling-based Operation Space Map Search (SbOSMS) method
3 Candidate position determination for single manipulator

The innermost part of the SbOSMS algorithm involves the quick establishment of the local search map of all manipulators located in their respective operation spaces during each sampling cycle. The candidate position points for a single manipulator consist of all possible position points of the EEF that the manipulator can reach within the operation space during the next sampling period. Determination of these candidate positions is the cornerstone of the strategy proposed herein and the key to building a simple and effective operation space search map (OSSM).

It is assumed that the moving step length of a manipulator EEF during every sampling interval is \(L_{\text{step}}\). Therefore, the set of all candidate positions of a single manipulator in the next sampling period constitutes a spherical surface, in which the current Cartesian position coordinates of the EEF, \(P_{\text{EEF}}\), is the sphere center and \(L_{\text{step}}\) is the radius, and this spherical surface set is named as search feasible region (SFR). Theoretically, there can be an infinite number of candidate position points within SFR.

However, a spherical surface is an infinite continuous point set that cannot be used as a discrete search map; thus, selection of a specific number of points is necessary. Herein, an approach to find three non-coplanar vectors as reference direction vectors and determine all candidate positions for a single manipulator, referring to building “cells” in chemistry, is presented. In this section, two different methods are proposed to find the three reference direction vectors: the first one is anchored on the base coordinate system of each manipulator, and the second one considers the current velocity of each manipulator EEF.

Before discussing how to determine the three reference direction vectors, the concepts of main movement axis, secondary movement axis, displacement axis, and main movement surface are defined.

**Definition**

Assuming that a point moves from the start position \(P_{\text{start}}\) to the endpoint \(P_{\text{end}}\) under an arbitrary coordinate system \(Oxyz\); the vector from \(P_{\text{start}}\) to \(P_{\text{end}}\) is \(P_{\text{disp}} = (x_{\text{disp}}, y_{\text{disp}}, z_{\text{disp}})\), which indicates the size and direction of the movement. Displacements in the positive \(x\), \(y\), and \(z\) directions are \(x_{\text{disp}}\), \(y_{\text{disp}}\), and \(z_{\text{disp}}\), respectively. The values \(\|x_{\text{disp}}\|\), \(\|y_{\text{disp}}\|\), and \(\|z_{\text{disp}}\|\) can be compared and sorted in descending order. 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. 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 shown in Fig. 3, if \(\|x_{\text{disp}}\| \geq \|y_{\text{disp}}\| \geq \|z_{\text{disp}}\|\), then the \(x\)-axis is the main movement axis, the \(y\)-axis is the secondary movement axis, and the \(z\)-axis is the displacement axis, and the \(xoy\)-plane is the main movement plane.

![Diagram of displacement axis, secondary movement axis, and main movement axis](image)

**Fig. 3** Diagram of displacement axis, secondary movement axis, and main movement axis of a moving path from \(P_{\text{start}}\) to \(P_{\text{end}}\) under \(oxyz\) coordinate system

The \(y\)-axis is the secondary movement axis, the \(z\)-axis is the displacement axis, and the \(xoy\)-plane is the main movement plane.

### 3.1 Coordinate system-based method

The principle of the coordinate system-based method involves the projection of the base coordinate system of the single manipulator onto the global coordinate system of the multi-manipulator cooperative manufacturing system. Figure 4 exhibits the standard base coordinate system for a single manipulator. Clearly, when the manipulator is in its initial position (i.e., the values of all six joints are zero), the \(x\)-\(y\) plane of the base coordinate system is the bottom surface of the base of the manipulator, and the origin point of base coordinates system is the center point of the bottom surface of the base. The \(z\) axis of the base coordinates system is the axis perpendicular to the bottom surface of the base.
and points to the direction of the mechanical manipulator arm. The y axis is the axis in the \( x - y \) plane and points to the position of EEF, and then, the x axis can be determined by the right-hand rule. Then, the base coordinate system of the manipulator \( O_{xyz} \) is built up. \( O_{xyz} \) is the global coordinate system.

The execution steps of the coordinate system-based method are presented 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. The base coordinate system can be arbitrary; therefore, the base coordinate system and the global coordinate system may not coincide;

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

\[
x_r = y_r \times (x \times y_r)
\]  

(1)

which indicates that the reference direction vector \( x_r \) is close to the \( x \) axis and perpendicular to both the \( z_r \) and \( y_r \).

However, in some cases, such as the \( z \) axis of the base coordinate system remains in the main movement plane, which indicates that the \( z \) axis is perpendicular to the displacement axis, or the \( y \) axis coincides with the displacement axis, which indicates that the projection of \( y \) axis on the main movement plane is the origin point itself, then the aforementioned process will lose efficacy. In these cases, it is only required to interchange the \( y \) axis and the \( z \) axis of the base coordinate system, and then the three reference direction vectors can be obtained according to the processes mentioned above.

Obviously, these three reference direction vectors \((x_r, y_r, z_r)\) are perpendicular to each other as the base coordinate system and global coordinate system do.

### 3.2 Velocity-based method

The coordinate system-based approach determines the three reference direction vectors merely based on the manipulator’s position in its operation space with no reference to its kinematic parameters. The law of inertia indicates that an object continues to move along its current state of motion when no external force is applied. When an industrial manipulator encounters a potential collision threat and must re-plan its subsequent path, the smaller the deviation of the manipulator’s path point in the next sampling period from the direction of the manipulator’s moving velocity at the current instant, the smaller the energy loss of the manipulator. This conforms to the principle of energy minimization constraints in robot control algorithms. Therefore, herein, a new method was proposed to determine the three reference direction vectors based on the current velocity.

First, a new coordinate system called velocity coordinate system \( O_{vxyz} \) is set up to calibrate the motion of every single manipulator, where \( O_x \) is the current position point of manipulator EEF. The right half side of Fig. 6 shows that the velocity of the manipulator EEF at point \( P \) is \( v_A \) at the current time instant. Thus, the direction of vector \( v_A \) can be considered as the direction of the reference \( x_r \)-axis. Then, the EEF position \( P \) and the base of the manipulator \( O \) can be connected to construct the space vector \( PO \), which is projected onto the bottom surface of the manipulator base in the same coordinate system and serves as our reference \( y_r \)-axis. Finally, the \( z_r \)-axis of the candidate position points can be calculated from the \( x_r \) and \( y_r \)-axis according to the following formula:

\[
z_r = x_r \times y_r
\]

(2)

Then, how to determine the three reference direction vectors is the same as presented in Sect. 3.1.

### 3.3 Candidate position point determination

After setting up the three reference direction vectors, determination of candidate positions for a single manipulator becomes an easy task. Figure 5 demonstrates that it is assumed that the coordinate of the center point (black point) of the EEF at time \( t \) is \( P = (x_t, y_t, z_t) \), and \( \Delta P = (\Delta x, \Delta y, \Delta z) \) is the step gain required for the EEF to reach the target position along the three reference direction vectors, where \( \Delta x = \Delta y = \Delta z \). In other words, if the \( x \) axis coordinate of point \( P \) 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 by using the following formula:

\[
P_{cad} = (x_{t+1}, y_{t+1}, z_{t+1})
\]

\[
= \Theta \left\{ \left( x_t + \Delta x, y_t, z_t \right), \left( x_t, y_t - \Delta y, z_t \right), \left( x_t, y_t, z_t + \Delta z \right) \right\}
\]

(3)

where \( \Theta \) is the set flag and the operator “\( \times \)” denotes “combination.”
These coordinate combinations can set up as 27 points (marked $P_1 - P_{27}$; yellow points in Fig. 5) in Cartesian space, and form a cube with side length $2\Delta x$ about point $P$. These 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, respectively, which results in difficulties in setting up the cost function.

In order to fix every displacement, an inscribed sphere of radius $\Delta x$ is placed into each cube, as shown in Fig. 5. From point $P$ to $P_1 - P_{27}$, there are 27 radials intersecting the inscribed sphere resulting in 27 intersection points marked as $P_{1}^\prime - P_{27}^\prime$ (green points in Fig. 5). The intersection points are the candidate positions with constant displacement in Cartesian space for a single manipulator during the next scanning period.

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

4 Search map simplification process

For a multi-manipulator system, after setting up the “unit cell” structure of a single manipulator, the next step involves the setting up of a reasonable OSSM for the entire system. The number of nodes in the “unit cell” can be reduced according to the physical and kinematic characteristics of the multi-manipulator system; therefore, a system of dual manipulators and one with no less than three manipulators exhibit different physical space position features. Thus, the SMS of these two systems must be described separately.

4.1 Search map simplification for dual-manipulator system

The two manipulators are marked as A and B in Fig. 6. The global coordinate system of the dual-manipulator is $T = (x, y, z)$, while $T_A = (x_A, y_A, z_A)$ and $T_B = (x_B, y_B, z_B)$ are the base coordinate systems of manipulators A and B, respectively. The two manipulators are mounted on the same plane; therefore, the $z$-axis of each base coordinate system in the same direction can be defined as the $z$-axis of the global coordinate system.

As mentioned in Sect. 3.3, the candidate positions of manipulators A and B can be expressed as $P_{\text{cad}}^t_A$ and $P_{\text{cad}}^t_B$, respectively, with each containing 27 elements in their own three reference direction vectors. Every element represents a position point under a global coordinate system, $P_{\text{cad}}^t_A$ can be set as the horizontal coordinate and $P_{\text{cad}}^t_B$ as the vertical coordinate, in order to compose an OSSM with $27 \times 27$ nodes. Finding an optimal node in this map can provide an optimal solution for the dual-manipulator cooperation in the next step. Furthermore, if the time variable is considered, the time dimension can be added to the OSSM to produce

Fig. 5 Candidate position points of a single manipulator in its operation space. The left-hand side is a diagram of candidate positions determined by the coordinate system-based method; the right-hand side is the candidate positions determined via the velocity-based method.

Fig. 6 Diagram of universal dual-manipulator cooperation situation.
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. 7).

However, an OSSM with $27 \times 27$ nodes is not efficient enough to meet real-time requirements. Moreover, when using A* to execute path planning operations in the above-stated OSSM, it becomes extremely easy to fall into the local minima trap. Based on the fact that the SbOSMS method produces locally optimal paths in each sampling period, an oversized OSSM can significantly increase the probability of bumping into a local minimum trap. Therefore, additional measures must be taken to reduce the number of nodes.

Every element of the candidate positions for a single manipulator consists of 3-axis coordinates according to their own reference direction vectors, in which $(x_r, y_r, z_r)$ represents three non-coplanar reference direction vectors. The nodes of OSSM can be reduced along the $x_r$, $y_r$, and $z_r$-axis, respectively, based on the following movement analyses:

(a) Along $x_r$-axis

Based on the previous assumption, the coordinate value of point $P$ along the $x_r$-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 the following three choices: “step backward toward the starting point,” “remain stationary,” and “step forward toward the target.” The “step backward to the starting point” option does not promote the efficient movement of a manipulator; therefore, this choice can be abandoned. Thus, movement along the main movement axis can be reduced from 3 to 2 choices.

(b) Along $y_r$-axis

The movement in the main movement plane of manipulator A (Fig. 6) is shown in Fig. 8.

“Movement toward the manipulator base” (green arrow) and “remain stationary” options are the only two reasonable avoidance movements along $y_r$-axis, because movement toward the base of the other manipulator leads to the increase in the possibility of collision. In the example scenario, manipulator A abandons moving along positive $y_r$-axis, while manipulator B abandons along the negative $y_r$-axis. In a similar way, the choice of movement along the secondary movement axis can also be reduced from 3 to 2 choices.

(iii) Along $z_r$-axis

In the example scenario, the $z_r$-axes of dual manipulators are the same and both are same with the $z_g$-axis of the global
coordinate system. The coordinate values at time $t$ for the two manipulators under the global coordinate system are $z'_A$ and $z'_B$, respectively. Two boundary values $\varepsilon$ and $-\varepsilon$ are set, where $\varepsilon > 0$. If $z'_A - z'_B > \varepsilon$, the reasonable movements of manipulators A and B along their respective $z'_g$-axes are in the positive (toward the target point) and negative (away from the target point) directions, respectively. In contrast, if $z'_A - z'_B < -\varepsilon$, the reasonable movements of manipulators A and B along their respective $z'_g$-axes are in the negative and positive directions, respectively. However, if $\|z'_A - z'_B\| \leq \varepsilon$, 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.

However, if the $z'_g$-axes of dual manipulators are both along the same axis, but the axis is not the $z'_g$-axis of the global coordinate system, then the along-axis of the global coordinate system is considered as a new “$z'_g$-axis” and step c is repeated. If the $z'_g$-axes of dual manipulators are not along the same axis, then manipulator A and B only need to move in the main movement plane of each manipulator.

The above-mentioned analysis shows that the number of nodes in the OSSM can be reduced from $27 \times 27$ to $7 \times 7$, or in some cases, $3 \times 3$.

### 4.2 Search map simplification for multiple manipulators

According to the analysis presented in Sect. 4.1, additional processing procedures should be performed to simplify the OSSM in order to meet the requirements for real-time calculation. In this section, the Operation Space Map Search method is applied to the cooperation of multiple manipulators in the same way it can be applied to the 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. 9.

In the schematic, $T_A = (x_A, y_A, z_A)$, ..., $T_N = (x_n, y_n, z_n)$ are the base coordinate systems of manipulators A to N, individually, and $T_g = (x_g, y_g, z_g)$ is the global coordinate system. Moreover, $T_{Ag}$ is the transformation matrix of coordinate system $T_A$ in relation to the global coordinate system $T_g$, and similarly, $T_{Bg}$ to $T_{Ng}$. According to Sect. 3, one manipulator has 27 candidate position points. Therefore, for $N$ manipulators, the entire system produces an

![Fig. 9 Schematic showing position and coordination transformation relationships for the cooperation of multiple manipulators. Notably, the base coordinate of each manipulator can be arbitrary, and the overlapping of all the workspaces of the manipulators is not required.](image-url)
OSSM with $27^N$ nodes, which undoubtedly hinders its practical application.

Noteworthy, if the coordinates for the starting points $p_{\text{start}}$, $p_{\text{start}}^B$, …, $p_{\text{start}}^N$ and target points $p_{\text{arg}}$, $p_{\text{arg}}^B$, …, $p_{\text{arg}}^N$ 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 easily identified. Thus, the number of elements in the OSSM can be reduced following the same logic used in Sect. 4.1.

When the coordinates for the starting points and target points of all manipulators are obtained, 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 $z_g$ displacement axis are collected as the set of “$z_g$ displacement axis manipulators” (whose total number is $i$). Similarly, the set of “$x_g$ displacement axis manipulators” (whose total number is $j$) and “$y_g$ 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 $x_r$-axis, there are two candidate positions (“remain stationary” or “move one step toward its target”).
(b) Along the $y_r$-axis, there are two candidate positions (“remain stationary” or “move one step toward its base”).
(c) Along the $z_r$-axis, situation is slightly more complex and must be classified based on the real-time situation. All the $z_r$-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, $\varepsilon$ and $-\varepsilon$ where $\varepsilon > 0$, such that values greater than $\varepsilon$ 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, i.e., remain stationary or move one step toward 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, i.e., 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-mentioned 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, reduction in the number of candidate positions is independent. However, the above-mentioned 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 obtain an optimal result for every manipulator during its next movement.

Next, the reduction results of the number of OSSM nodes for the cooperation of three manipulators were displayed herein. Assuming all $x_r$-axes are $x_g$-axes, all $y_r$-axes are $y_g$-axes, and all $z_r$-axes are $z_g$-axes under the global coordinate system, the results of the element reduction were obtained, as presented in Table 2.

| Robot A | Robot B | Robot C | Nodes of OSSM |
|---------|---------|---------|---------------|
| Before reduction |
| $x_r$-axis |
| $y_r$-axis |
| $z_r$-axis |
| After reduction |
| $x_r$-axis |
| $y_r$-axis |
| $z_r$-axis |

| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
|------------------|------------------|------------------|------------------|------------------|------------------|
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |
| $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ | $z_g > z_g > z_g$ |

“−” in the table indicates moving away from the target point; “+” indicates moving toward the target point. Herein, all cases are not enumerated, but still represent all possible scenarios for number of elements of the OSSM of three cooperating manipulators.
Table 2 presents that the current positions of the manipulators are $P'_A = (x'_A, y'_A, z'_A)$, $P'_B = (x'_B, y'_B, z'_B)$, and $P'_C = (x'_C, y'_C, z'_C)$, under each reference direction vectors.

### 5 Cost function and optimal solution selection

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

$$
\theta^t_i = (\theta^t_{1i}, \theta^t_{2i}, \theta^t_{3i}, \theta^t_{4i}, \theta^t_{5i}, \theta^t_{6i})
$$

(4)

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

$$
\theta^{t+1}_i = (\theta^{t+1}_{1i}, \theta^{t+1}_{2i}, \theta^{t+1}_{3i}, \theta^{t+1}_{4i}, \theta^{t+1}_{5i}, \theta^{t+1}_{6i})
$$

(5)

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

$$
P^{t+1}_i = (x^{t+1}_i, y^{t+1}_i, z^{t+1}_i)
$$

(6)

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

$$
P^s_i = (x^s_i, y^s_i, z^s_i)
$$

(7)

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

$$
F(t) = H(t) + G(t)
$$

(8)

where $F(t)$ is the total cost value of the current node in the search map at the current time instant, $H(t)$ denotes the cost value from the current node to candidate node, and $G(t)$ represents the cost value from the current node to the 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 operation space can be calculated individually. The cost value for the $i$th node in the OSSM of the $j$th manipulator can be calculated as follows:

$$
F^j_i(t) = \omega^H_i \times H_i(t) + \varphi^G_i \times G_i(t)
$$

(9)

$$
= \omega^H_i \times ||\theta^{t+1}_i - \theta^t_i|| + \varphi^G_i \times ||P^s_i - P^{t+1}_i||
$$

where $\|A\|$ is the secondary norm of vector $A$, and $\omega^H_i$ and $\varphi^G_i$ 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

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

(10)

where $c_j = (c_1, c_2, \ldots, c_N)$ 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 OSSM for all manipulators, and summarized as the following mathematical model:

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

(11)

### 6 Simulation and analysis

In this study, two different methods were proposed to set up the direction vector of OSSM for the multiple manipulators cooperative system. Herein, the two methods were first tested for the dual manipulator scenario, and the simulation results were comparatively analyzed. Subsequently, the scenario of a triple manipulator system was used as an example to show how the method works in the situation of multiple manipulators. Then, a simulation comparison between our proposal and A* was introduced and analyzed in detail. Finally, the simulation results were evaluated, and some conclusions were drawn from these results. Moreover, the execution time of each method was assessed to determine whether it was possible to meet the real-time requirements.

Assuming $N$ manipulators in a system of multiple manipulators cooperating, all 6-DOF industrial manipulators have standard D-H parameters, which are presented in Table 3.

The simulation results for the dual-manipulator and triple-manipulator systems were analyzed individually. For more than three manipulators, the results were relatively similar to those obtained for the triple-manipulator system, except for the increase in the number of the OSSM nodes and execution time. All the algorithms were programmed in C++ and run on a 64-bit Windows operating system with an i7-4790 CPU GHz with 8 GB RAM.

#### Table 3 D-H parameters for a 6-DOF manipulator

| Link $j$ | Link offset $d_j$ | Link length $a_{j-1}$ | Twist angle of link $\alpha_{j-1}$ | Rotation angle of link $\theta_j$ |
|----------|-------------------|----------------------|-----------------------------------|-----------------------------|
| 1        | 0                 | 0                    | 0                                 | $\theta_1$                  |
| 2        | $d_1$ mm          | 0                    | $-90^\circ$                       | $\theta_2$                  |
| 3        | $d_2$ mm          | 0                    | $0^\circ$                         | $\theta_3$                  |
| 4        | $d_3$ mm          | $d_2$ mm             | $-90^\circ$                       | $\theta_4$                  |
| 5        | 0                 | 0                    | $90^\circ$                        | $\theta_5$                  |
| 6        | 0                 | $-90^\circ$          | $\theta_6$                       |                             |
6.1 Simulation of dual-manipulator cooperative manufacturing system

For manipulators A and B, we can define $a_i^1 = 25 \text{ mm}$, $a_i^2 = 315 \text{ mm}$, $a_i^3 = 35 \text{ mm}$, and $d_i^4 = 365 \text{ mm}$ ($i = 1, 2$), where A and B are given the same parameters and positioned face-to-face at a distance of $D = 900 \text{ mm}$. The minimum tolerance for the distance between the two manipulators was set as $D_{\text{min}} = 20 \text{ mm}$. The global coordinate system originated at the base of B, which indicates that the coordinates of the base of B were $(0, 0, 0)$ and for the base of A were $(0, -900, 0)$. The step length of each manipulator and boundary values were predefined as $l_i = 10 \text{ mm}$ and $\varepsilon = 40 \text{ mm}$, respectively.

Three simple simulation scenarios were established as follows:

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

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

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

6.1.1 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-mentioned three scenarios are the $x$-axis, $y$-axis, $z$-axis, and $xoy$-plane.

The simulation results using the coordinate system-based method for each of the above-mentioned scenarios are presented in Fig. 10.

Movements along the $x$-, $y$-, and $z$-axis for both manipulators are shown in Fig. 11.

Moreover, the paths of both manipulators are clearly divided into three distinct parts, marked as 1, 2, and 3 (Fig. 11) and named as the “original path period,” “avoidance period,” and “forward period,” respectively. The

![Fig. 10 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 operating space are represented by the red and blue curves, respectively. The upper images show the 3D views, and the lower images exhibit the top view.](image-url)
“original path period” indicates that the manipulator moves along its original path without deviation, the “avoidance period” is the case where a manipulator path is re-planned in order to avoid collision, and the “forward period” is the last situation for which the manipulator moves toward its target along a straight line.

For any scenario, it can be observed that when a collision is about to occur, it can be avoided by manipulator A in the $-y$ direction and avoided by B in the $+y$ 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 $z$-axis stay constant during the “avoidance period” for all three scenarios. This is attributed to the fact that for each of the three scenarios, when the two manipulators moved into the PCA, the

---

**Fig. 11** 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 entire path plan period, and the vertical axes represent the coordinate values under corresponding reference direction vectors.
coordinate values of the z-axis of the two manipulators $Z'_A$ and $Z'_B$ met the condition of $\|z'_A - z'_B\| \leq \varepsilon$, as described in Sect. 4.1. Herein, simulation d was set up to observe how the coordinate values of the z-axis would change when the condition $\|z'_A - z'_B\| \leq \varepsilon$ is untenable.

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

Figure 12 demonstrates that, when the two manipulators move into the PCA, $\|z'_A - z'_B\| > 100 > \varepsilon$; therefore, B has the choice of “+z” and “remain stationary,” and A has the choice of “−z” and “remain stationary,” according to the principles of the PSMS method. The results show that $z'_A$ remained stationary, while $z'_B$ changed along +z direction in the “avoid period.”

Real-time execution is also a key assessment index of our algorithms. In order to obtain the per step execution time of the SbOSMS method, manipulator A was driven to move from $P_{\text{start}}^A = (-400, -450, 600)$ to $P_{\text{end}}^A = (400, -450, 600)$ and manipulator B was moved from $P_{\text{start}}^B = (-400, 450, 600)$ to $P_{\text{end}}^B = (400, 450, 600)$ in the base coordinate system of each manipulator, simultaneously. Then, the base distance $D$ was changed from 1000 to 900 to 800mm, and each change was tested 5 times. The simulation results are presented in Fig. 13.

The test results of execution time of entire path planning period are summarized in Table 4.

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 using the variables $T_{\text{op}}, T_{\text{ap}},$ and $T_{\text{fp}},$ and by default, the three variables remain constant for each period. Figure 14 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.

Table 4: Execution time for simulation based on dual manipulators (unit: μs) for different distances

| Simulation num. | $D = 1000$mm | $D = 900$mm | $D = 800$mm |
|----------------|--------------|--------------|--------------|
|                | 81 steps     | 86 steps     | 91 steps     |
| 1              | 44,881.44    | 67,200.45    | 75,222.38    |
| 2              | 44,856.35    | 67,678.86    | 74,210.83    |
| 3              | 45,633.26    | 69,841       | 74,384.46    |
| 4              | 44,125.07    | 68,584       | 76,169.49    |
| 5              | 44,267.05    | 65,112.65    | 74,865.43    |
| Average        | 44,752.63    | 67,683.4     | 74,970.52    |
Using a timer program embedded in the algorithm, average values of $T_{op}$, $T_{ap}$, and $T_{fp}$ could be obtained as follows:

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

Then

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

Comparative analysis of the above-mentioned results with those presented in Table 5 indicates that the errors are within the allowable range. The results suggest that path planning through the SbOSMS method for dual manipulators can be executed in less than 3 ms and fully meet the real-time requirements.

6.1.2 Results of simulation by velocity-based method

The simulation results for path planning, using the velocity-based method under scenarios a, b, and c, are presented in Fig. 15. The three images shown in the lower half of Fig. 15 indicate that regardless of the scenario, when a collision is about to occur, both A and B can plan a new path with different geometries from those presented in Fig. 10. However, A usually avoids a collision by moving in the $-y$ direction, while B moves in the $+y$ direction, indicating both manipulators move toward their 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. 17) show movements along $x$-, $y$-, and $z$-axis under scenario d through the velocity-based method, clearly shown to be different along the $z$-axis from those using the coordinate system-based method (Fig. 12), mainly due to differences in the principles employed in each method.

To obtain the per step execution time through the velocity-based method, the simulation scenario similar to the coordinate system-based method was established (Fig. 13). The simulation results are summarized in Fig. 18. The results for execution time over the entire path-planning period are summarized in Table 5.

By 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 presented in Table 6, it was found that the errors were within the allowable range. The results indicate that path planning using the SbOSMS method for dual manipulators can be executed in less than 4 ms, fully meeting the real-time requirements.

(1) Comparison of simulation results

Based on the analysis and results presented in the previous sections, both methods can be used to plan non-collision 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 exhibit the top view

![Fig. 15 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 exhibit the top view.]

![Fig. 16 Movements of manipulators A and B along x-, y-, and z-axis for scenarios a, b, and c by 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 entire path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.]

Scenario a

Scenario b

Scenario c
paths for dual-manipulator cooperation systems in real-time. However, a number of similarities and differences can be observed via careful comparative analysis of the two methods:

(a) The simulated paths of both methods are not smooth throughout, and there are “twists and turns” in the geometries. One reason may be that all nodes in the search map are special positions within the SFR.

Fig. 17 Results of path planning simulation and movements along x-, y-, and z-axis dual-manipulator cooperation under scenario d via velocity-based method

Fig. 18 Results of path planning simulation of dual-manipulator cooperation by velocity-based method with decreasing distance between the bases of two manipulators where a, b, and c represent the simulation results with distances of 1000mm, 900mm, and 800mm, respectively
Another may be the influence of step length, for which the simulation and analysis are presented in Fig. 20.

(b) The geometries of the paths simulated using each of the two methods are not the same. Comparative analysis of Figs. 10 and 15 indicates that the paths generated by the velocity-based method exhibit larger displacements along $x$- and $y$-axes than the coordinate system-based method. However, the planned stepped within the “avoidance period” were smaller by using the velocity-based method than that of the coordinate system-based method.

(c) The simulated paths of the velocity-based method (Figs. 15, 16, 17, 18 and 19) exhibit more apparent “twists and turns” than those of the coordinate system-based method (Figs. 10, 11, 12, 13 and 14). This is due to the 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 system-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 by employing both methods according to scenario c (Figs. 13 and 18), with step lengths of 10mm, 5mm, and 1mm. The results are shown in Fig. 20. Although 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. 20). The reason is not totally clear; however, one possibility is that the reference direction vectors of the velocity-based method change 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 that required for the coordinate system-based method. As such, the coordinate system-based method is more effective and demonstrates greater stability compared to the velocity-based method. Therefore, the coordinate system-based method is used to demonstrate the SbOSMS method for triple manipulator cooperation.

### Table 6

| Num. | $D = 950\text{mm}/88$ steps | $D = 900\text{mm}/84$ steps | $D = 850\text{mm}/101$ steps |
|------|-----------------------------|-----------------------------|-----------------------------|
| 1    | 137,257.11                  | 137,562.24                  | 181,038.46                  |
| 2    | 135,620.94                  | 139,697.94                  | 178,944.83                  |
| 3    | 136,982.56                  | 138,442.63                  | 180,225.33                  |
| 4    | 135,779.06                  | 143,752.65                  | 177,698.12                  |
| 5    | 138,257.86                  | 138,176.63                  | 181,908.09                  |
| Average | 136,779.5                  | 139,526.42                  | 179,962.97                  |

**Fig. 19** 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 entire path plan period and the vertical axes represent the coordinate values under corresponding reference direction vectors.
Simulation of the triple-manipulator manufacturing system

For the three cooperating manipulators A, B, and C, their D-H parameters can be defined as follows:

\[ a_i = \begin{cases} 25 \text{ mm} & \text{for } i = 2, 3, 4; \\ 315 \text{ mm} & \text{for } i = 1; \end{cases} \]

\[ d_i = \begin{cases} 35 \text{ mm} & \text{for } i = 2, 3, 4; \\ 365 \text{ mm} & \text{for } i = 1. \end{cases} \]

where \( A, B, \) and \( C \) are given the same parameter values, and the relationships among the positions of the manipulators are shown in Fig. 21. The bases of the triple manipulators consist of an equilateral triangle with a length of sides 900 mm. The minimum tolerance for the distance between two manipulators was set as \( D_{\text{min}} = 50 \) mm. The global coordinate system originated at the base of \( B \), which indicates that the coordinates of the base of \( A, B, \) and \( C \) were \((900, 0, 0), (0, 0, 0), \) and \((450, 450\sqrt{3}, 0)\).

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

The simulation results based on the above-mentioned scenario are presented in Fig. 22.

When a collision is about to occur, all manipulators move toward their respective bases, successfully bypassing the PCA and achieving the goal of collision avoidance.

Again, to examine the per step execution time through the PSMS method for triple-manipulator cooperation, the same three simulation scenarios were used. Only the base distance D was varied from 950 to 900 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 6.

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

### 6.3 Simulation comparison of A* and SbOSMS in dual-manipulator cooperative manufacturing system

Figure 22 indicates that the SbOSMS algorithm is based on an adaptation of the A* algorithm. Considering the generality of the algorithms, five different cases of dual-manipulator cooperative manufacturing system, each with different initial motion paths, were established, and the two SbOSMS algorithms and A* algorithm were simulated and compared for all five cases. Herein, the settings for the D-H parameters and the position of the manipulator base are maintained as presented in Sect. 6.1. For manipulators A and B, we can define $a_i^1 = 25$ mm, $a_i^2 = 315$ mm, $a_i^3 = 35$ mm, and $d_i^4 = 365$ mm ($i = 1, 2$), where $A$ and $B$ are given the same parameters and positioned face-to-face at a distance of $D = 900$ mm. The minimum tolerance for the distance between the two manipulators was set as $D_{\text{min}} = 20$ mm. The global coordinate system originated at the base of B, which indicates that the coordinates of the base of B were (0, 0, 0) and for the base of A were (0, −900, 0). The step length of each manipulator and boundary values were predefined as $l_s = 10$ mm and $\varepsilon = 40$ mm, respectively.

Case 1: the EEF of manipulator A moves from $P_{\text{start}}^A = (400, -450, 500)$ toward $P_{\text{end}}^A = (-400, -450, 500)$
Fig. 23 Comparison of dual manipulators avoidance simulation results for five initial cases: As the simulation results are 3D path curves, the results are presented in a comprehensive and extensive visualization manner with four views: axonometric, top, front, and side views.
Fig. 24 Comparison of simulation results of three path planning algorithms for Case 1

Fig. 25 Comparison of simulation results of three path planning algorithms for Case 2
while manipulator A moves from $P_{A}^{\text{start}} = (-400, -500, 500)$ toward $P_{A}^{\text{end}} = (400, -500, 500)$.

Case 2: the EEF of manipulator A moves from $P_{A}^{\text{start}} = (400, -450, 500)$ toward $P_{A}^{\text{end}} = (-400, -450, 500)$ while manipulator A moves from $P_{B}^{\text{start}} = (-400, -450, 500)$ toward $P_{B}^{\text{end}} = (400, -450, 500)$.

Case 3: the EEF of manipulator A moves from $P_{A}^{\text{start}} = (400, -450, 500)$ toward $P_{A}^{\text{end}} = (-400, -450, 500)$ while manipulator A moves from $P_{B}^{\text{start}} = (-400, -400, 500)$ toward $P_{B}^{\text{end}} = (400, -400, 500)$.

Case 4: the EEF of manipulator A moves from $P_{A}^{\text{start}} = (400, -450, 500)$ toward $P_{A}^{\text{end}} = (-400, -500, 500)$ while manipulator A moves from $P_{B}^{\text{start}} = (-400, -400, 500)$ toward $P_{B}^{\text{end}} = (400, -400, 500)$.

Case 5: the EEF of manipulator A moves from $P_{A}^{\text{start}} = (400, -450, 400)$ toward $P_{A}^{\text{end}} = (-400, -450, 600)$ while manipulator A moves from $P_{B}^{\text{start}} = (-400, -400, 500)$ toward $P_{B}^{\text{end}} = (400, -450, 600)$.

Considering Case 1 as a benchmark, cases 1, 4, and 5, respectively, correspond to the three simulation scenarios presented in Sect. 6.1 (scenarios a, b, and c shown in Figs. 10 and 15), while cases 1, 2, and 3, respectively correspond to the three simulation scenarios a, b, and c shown in Figs. 13 and 18. Therefore, these five cases can completely cover all the simulation scenarios presented in Sect. 6.1. In this subsection, the three algorithms (A*, coordinate system-based SbOSMS, and velocity-based SbOSMS) were separately operated for each simulation case, and the simulation results were collated into five simulation comparison diagrams shown in Figs. 24, 25, 26, 27 and 28 according to the cases, so as to show the pros and cons of the three algorithms more comprehensively.

Moreover, the space coordinate vector of an industrial manipulator arm is usually comprised of two components $P_b = ([x, y, z], [\alpha, \beta, \gamma])$, where $[x, y, z]$ represents the 3D space coordinates of the centre point of the manipulator EEF and $[\alpha, \beta, \gamma]$ represents the pose direction vector of the manipulator EEF. The EEF pose vectors of the manipulator A and B are assigned the values $[\alpha_A, \beta_A, \gamma_A] = [\pi/2, -\pi/2, 0]$ and $[\alpha_B, \beta_B, \gamma_B] = [-\pi/2, -\pi/2, 0]$ for all cases in Sects. 6.1 and 6.2. In order to verify the generality of our algorithm, all cases were resimulated by simply changing the corresponding EEF pose vectors without changing their individual space coordinates. When the pose vector of the EEF of manipulator A and B are set to $[\alpha_A, \beta_A, \gamma_A] = [\pi/2, 0]$
and $[\alpha_B, \beta_B, \gamma_B] = [-\pi/2, \pi, 0]$, the corresponding simulation results of coordinate system-based method are shown in Fig. 23. For details on the simulation results of the velocity-based SbOSMS algorithm with the new manipulator End-effector spatial pose vector, comparative analysis of simulation results was carried out, as shown in Figs. 24, 25, 26, 27 and 28.

Comparison of the simulation results presented in Figs. 10, 13 and 23 indicate that some apparent and obvious consensus can be quickly reached. Despite different EEF pose vectors and different initial motion states, the algorithm proposed in this study can always adaptively generate suitable and reasonable obstacle avoidance paths. Furthermore, despite the changes in the EEF pose vectors, the algorithm also produces essentially the same simulated planning paths for the same case. In other words, the EEF pose vectors have a negligible impression on the simulation results of the algorithm proposed in this study. Based on this conclusion, in the following comparative simulations of the three algorithms, it is sufficiently easy to select only one pose vector.

Figures 24, 25, 26, 27 and 28 maintain the same EEF pose vectors as Fig. 23. The simulation results evidently indicate that the path curves planned by the $A^*$ algorithm exhibit the highest spatial complexity in any case; i.e., the path planned by the $A^*$ algorithm must be deviated in all axial directions of the world coordinate system, regardless of the initial conditions. In contrast, the spatial complexity of the path curves planned by either the coordinate system-based SbOSMS or the velocity-based SbOSMS always remains the same as the initial paths. In detail, for Cases 1 to 4 (Figs. 24, 25, 26 and 27), all the initial paths are in the $X$–$Y$ plane in the world coordinate system; therefore, the path curves planned by both SbOSMS algorithms are also in the $X$–$Y$ plane, with no deviation in the $Z$-axis direction. For Case 5 (Fig. 28), the planned paths are biased in all the $XYZ$ directions due to the fact that the initial paths are already shifted in the $Z$-axis direction.

The total deviation of the planned paths from the initial paths is also a key indicator of the merit of the obstacle avoidance path planning algorithm. Assuming that the equation of the planned path curve is $g_0 = f_0(L)$ and the equation of the initial path curve is $g = f(L)$, then the total offset of this planned path is as follows:

$$\Delta g - g_0 = \int_0^L ||f_0(t) - f(t)|| dt$$
All the planning results shown in Figs. 24, 25, 26, 27 and 28 were used to calculate the total deviation utilizing the above-mentioned equation, and the results are summarized in Table 7.

Moreover, these three algorithms in the 5 cases mentioned above were simulated in this study and the respective algorithm execution time and the number of planning path steps are summarized in Table 8.

Comprehensive analysis of data presented in Table 7 clearly shows that the total deviation of the path curves planned by both SBOSMS algorithms with respect to the initial path is much smaller than the planning result of A*. However, the deviation comparisons between the two SBOSMS algorithms yielded conflicting results for different cases. Moreover, Table 8 illustrates that both SBOSMS algorithms can accomplish the task of obstacle avoidance path planning with faster execution speeds and fewer search steps than the A* algorithm. In case of both SBOSMS algorithms proposed in this study, the speed of their algorithm execution and the number of search steps are in the same order of magnitude. For different cases, each of the two SBOSMS-based algorithms has pros and cons in terms of total planning time; however, the total number of search steps is always less for the coordinate system-based SBOSMS than for the velocity-based SBOSMS.

Collectively, both SBOSMS algorithms result in superior obstacle avoidance path curves relative to the A* algorithm. In the actual planning process, one can select to operate

### Table 7: Summary of planning path deviations for the three algorithms under all six cases (Unit: mm)

| Algorithm                      | Case 1  | Case 2  | Case 3  | Case 4  | Case 5  |
|-------------------------------|---------|---------|---------|---------|---------|
| A* search                     | 8341.2  | 9276.3  | 9373.5  | 8887.8  | 9006.1  |
| SbOSMS-velocity based         | 4793.1  | 6563.8  | 7663.9  | 6829.5  | 7084.1  |
| SbOSMS-coordinate system based| 4280.3  | 6162.2  | 8126.8  | 5964.7  | 6765.3  |
either of these two algorithms depending on the constraints of the initial conditions.

### Conclusions and future research

In this study, a novel path planning method called SbOSMS was proposed. This method can successfully facilitate the planning of non-collision paths for all manipulators working under multiple-manipulator cooperative manufacturing scenarios according to the real-time circumstance at any time, when all manipulators are working along with their tasks without pause.

This study presented two different methods, namely, coordinate system-based method and velocity-based method, to build up the three reference direction vectors to determine the candidate positions of all manipulators. Further, the process for decreasing the number of nodes of the OSSM to meet real-time execution requirements was described in detail. Finally, a series of simulation tests was performed and the results demonstrated that the SbOSMS method could successfully achieve collision avoidance of manipulators while meeting the real-time requirements. However, different methods of setting up the OSSM may lead to different path geometries. Moreover, the method itself dictates the influence of step length on the smoothness of the path. Although decrease in the step length resulted in a smoother path for the coordinate system-based method, minimal effects were observed when using the current velocity-based method. Furthermore, both the operation time and the total deviation of the path curves of the two SbOSMS algorithms exhibit pros and cons due to the different planning cases. Therefore, in the actual planning process, one can select to operate either of these two algorithms depending on the constraints of the initial conditions. The coordinate system-based simulation with triple manipulator cooperation produced satisfactory results, thus clearly indicating that our algorithm can be extended to applications using multiple manipulators.

The maximum number of cooperating manipulators that the SbOSMS can deal with simultaneously should be clarified in future research. Moreover, planning of paths within the entire SFR will be of significant focus.

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

#### Funding
This research is supported by the Bosch (China) Investment Ltd. “Smart Manufacturing” international cooperation projects with Huazhong University of Science and 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.

#### Declarations

**Ethical approval** The work does not contain libelous or unlawful statements and 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 for publication** All authors agree to transfer copyright of this article to the publisher.

**Competing interests** The authors declare no competing interests.

#### References

1. Caccavale F, Uchiyama M (2016) Cooperative manipulation. Springer Handbooks. Springer, Cham, Springer Handbook of Robotics. https://doi.org/10.1007/978-3-319-32552-1_39
2. Jonas S, Verena K, Christin H, Klaus B (2015) Human centered assistance applications for the working environment of the future. Occup Ergon 12:83–95
3. Iima A, Tino S, Ilari M (2018) Refining levels of collaboration to support the design and evaluation of human-robot interaction in the manufacturing industry. 51st CIRP Conf Manuf Syst 72:93–98
4. Choi M, Lim S, Lee J (2010) Cooperative path planning for redundant dual-arm robot using low-dimensional sample-based algorithm. Mechat Syst 701–708
5. Nash A, Koenig S, Tovey C (2010) Lazy theta*: any-angle path planning and path length analysis in 3D. Proc 3rd Ann Symp Comb Search 2:153–154

A* search algorithm is the name of a path planning algorithm, see the following link: https://en.wikipedia.org/wiki/A*_search_algorithm
6. Jia Q, Chen G, Sun H, Zheng H (2010) Path planning for space manipulator to avoid obstacle based on A* algorithm. Chin J Mech Eng 46:109–115
7. Yu N, Wang Z (2011) Collision avoidance planning of manipulator based on C-space layered search arithmetic. 2011 Int Conf Electr Mech Eng Int Technol (EMEIT) 3258–3262
8. Li L, Shi Z, Guan Y, Zhao C, Zhang J, Wei H (2014) Formal verification of a collision-free algorithm of dual-arm robot in HOL4. IEEE Int Conf Robot Autom (ICRA) 1380–1385
9. Harada K, Foissotte T, Tsui T, Nagata K, Yamanobe N, Nakamura A, Kawai Y (2012) Pick and place planning for dual-arm manipulators. IEEE Int Conf Robot Autom 2281–2286
10. Lee S, Moradi H, Yi C (2012) A real-time dual-arm collision avoidance algorithm for assembly. IEEE Int Symp Assembly Task Plan 7–12
11. Larsen L, Pham VL, Kim J, Kupke M (2015) Collision-free path planning of industrial cooperating robots for aircraft fuselage production. IEEE Int Conf Robot Autom 2042–2047
12. Cohen B, Chitta S, Likhachev M (2014) Single- and dual-arm motion planning with heuristic search. Int J Robot Res 33:305–320
13. Chiddarwar SS, Ramesh BN (2011) Conflict free coordinated path planning for multiple robots using a dynamic path modification sequence. Robot Auton Syst 59:508–518
14. Afaghani AY, Aiyama Y (2014) Advanced-collision-map-based online collision and deadlock avoidance between two robot manipulators with PTP commands. 2014 IEEE Int Conf Autom Sci Eng (CASE) 1244–1251
15. Rodriguez C, Montano A, Suarez R (2014) Planning manipulation movements of a dual-arm system considering obstacle removing. Robot Auton Syst 62:1816–1826
16. Korayem MH, Nekoo SR (2016) The SDRE control of mobile base cooperative manipulators: collision free path planning and moving obstacle avoidance. Robot Auton Syst 86:86–105
17. Yang L, Qi J, Song D, Xiao J, Han J, Xia Y (2016) Survey of robot 3D path planning algorithms. J Control Sci Eng 1–22
18. Tovey C, Greenberg S, Koenig S (2003) Improved analysis of D*. IEEE Int Conf Robot Autom 3371–3378
19. Nash A, Koenig S, Tovey C (2010) Lazy theta*: any-angle path planning and path length analysis in 3D. 3rd Ann Symp Comb Search 153–154
20. Verscheure L, Peyrodie L, N Makni, N Betrouni, S Maouche, M Vermandel (2010) Dijkstra’s algorithm applied to 3D skeletonization of the brain vascular tree: evaluation and application to symbolic. Ann Int Conf IEEE Eng Med Biol Soc 3081–3084
21. Luchnikov VA, Medvedev NN, Oger L, Troahec JP (1999) Voronoi-Delaunay analysis of voids in systems of nonspherical particles. Phys Rev E 59:7205–7212
22. Choudhury S, Scherer S, Singh S (2013) Rrt*-ar: sampling based alternate routes planning with applications to autonomous emergency landing of a helicopter. IEEE Int Conf Robot Autom 3947–3952
23. LaValle SM (1998) Rapidly-exploring random trees a new tool for path planning
24. Karim AM, Jingzhou Y (2006) Swept volumes: foundation, perspective and application. Int J Shape Model 87–127
25. Lumelsky VJ (1985) On fast computation of distance between line segments. Inf Process Lett 21(2):55–56

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.