Article

A Sampling-Based Unfixed Orientation Search Method for Dual Manipulator Cooperative Manufacturing

Chang Su 1,* and Jianfeng Xu 2

1 School of Mechanical Science & Engineering, Huazhong University of Science & Technology, Wuhan 430074, China
2 State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science & Technology, Wuhan 430074, China; jfxu@hust.edu.cn
* Correspondence: suchang@hust.edu.cn

Abstract: The case of dual manipulators with shared workspace, asynchronous manufacturing tasks, and independent objects is named a dual manipulator cooperative manufacturing system, which requires collision-free path planning as a vital issue in terms of safety and efficiency. This paper combines the mathematical modeling method with the time sampling method in the classification of robot path-planning algorithms. Through this attempt we can achieve an optimal local search path during each sampling period interval. Our strategy is to build the corresponding non-linear optimization functions set based on the motion characteristics of the dual manipulator system. In this way, the path-planning problem can be turned into a purely mathematical problem of solving the non-linear optimization programming equations set. The spatial geometric analysis is used to linearize the predicted dual-manipulator minimum distance equation, thus linearizing the non-linear optimization equations set. Finally, this system of linear optimization equations will be mapped directly into a virtual Euclidean space and then solved intuitively using the spatial geometry theory. By simulation and comparing with the previous strategies, we find that the planning results of the newly proposed planning strategy are smoother and have shorter deviations as well as a higher algorithmic efficiency in terms of spatial geometric properties.

Keywords: dual manipulator system; cooperative manufacturing; collision-free path planning; non-linear optimization programming; minimum distance prediction

1. Introduction

1.1. Subsection

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 speeds, and a higher accuracy performance [1]. Thus, tasks requiring numerous workers can undoubtedly be accomplished cooperatively by multiple manipulators, meaning that multi-manipulators will not only work side-by-side but as dyads and teams. Analogous to the definition of Human–Robot Interaction (HRI) [2,3], the case of a dual manipulator manufacturing system can be divided into cooperation and collaboration according to the arrangement of the tasks. For the case of a collaborative 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 such circumstances can be transformed into the currently mature internal obstacle avoidance strategy [4]. However, for the case of the multiple manipulator cooperative manufacturing system, problems and limitations arise in the small overlapping workspaces that accommodate numerous cooperative manipulators, resulting in collisions if no related countermeasures are put in place. Under this circumstance, a path-planning algorithm that can circumvent the
constraints of the mechanical structure, simplify the planning space, and determine the paths of all manipulators involved in the cooperative system in real-time will be the kernel.

1.2. Relative Work

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. Since a single manipulator can be modeled as a particle in its own configuration space (C-space), the particle-based path-planning methods [5] can be used for single manipulator path-planning directly. Limited progress has been made for the non-collision path planning of multiple cooperating manipulators in C-space. Obstacles were modeled as simple geometries by Jia et al. [6], 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. [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, because of the massive amount of elements in the maps, the whole algorithm tends to be time-consuming. Li et al. [8] 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 it is 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 each arm’s trajectory arrangement and movement order 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.

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. [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 is extended due to difficulties in setting up the VRM, and raising the method to multiple robots is 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 choose a free path; however, dynamic environments cannot be accommodated since the obstacle must 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 the path planning for one robot is allowed at a time, and the planning process is performed offline. 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. It 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. [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 et al. [14] used a collision map to detect the collisions between two robots 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 collision. Rodriguez et al. [15] 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 accessible paths 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. [16] 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. Multi-manipulator path-planning algorithms can be instantly divided into a map search-based method, time-sampling-based method, mathematical model method, and others [17]. The map search-based method [18–20] is a global optimization algorithm that will cost a colossal amount of time in planning map construction. In comparison, the time-sampling-based method [21–23] divides the whole planning period into several orderly but isolated intervals and splices all the calculated outcomes of each interval into a complete result path or planning map in order.

2. Previous Work and Paper Organization

In our previous work, a novel method called the Sampling-based Operation Space Map Search (SbOSMS) method (picture c in Figure 1), which combines the map search method (picture a in Figure 1) with the time-sampling-based method (picture b in Figure 1), was proposed. Equivalently, a local planning map was established during each time interval, and an optimal local path was determined following the local planning map. All chronologically planned path segments are glued together to generate a collision-free path, which is highly effective at the cost of forgoing the global optimum.

![Figure 1](image.png)

*Figure 1.* Schematic diagram of four different path-planning strategies for industrial manipulators. Picture (a) shows the planning strategy of map-based planning algorithms such as A*; Picture (b) shows the planning strategy of sampling-based planning algorithms such as RRT, PRM, and APF, etc.; Picture (c) is SbOSMS strategy which combines a map-based planning strategy and a sampling-based planning strategy to reduce both the search space and search time, as well as overcome the defect of random planning results for RRT; Picture (d) shows the proposal of this paper, which changes the node map in picture c into local-global sphere search space around EEF position point during every sampling period.
When focusing on how to build a suitable search map during each sampling period, we find that the nodes constituting the search map are several specific points selected by a particular rule on the surface of the entire search space (usually a sphere space) in that sampling period (Figure 2). Therefore, calculating the optimal path points in the whole search space (sphere space) surface rather than merely the constructed search map will be the central focus of this paper’s study, as shown in picture d in Figure 1.

Figure 2. Schematic of a local search map within a sampling period in SbOSMS and our proposal. The set of all blue dots located on the surface of the green sphere in the right half of the diagram forms the search map for the SbOSMS method during a single sampling period. This paper attempts to extend the local search map within a single sampling period from these “special point sets” to the entire green sphere surface area.

In this paper, a brief path-planning method review of the dual manipulator cooperative system is shown in Section 1. Section 2 provides a concise snapshot of our initial efforts on this issue and the limitations we encountered. Section 3 proposes a mathematical model for the non-linear optimization of dual manipulator path planning in detail. To convert all equation terms in the set of non-linear optimization equations into equations with the same independent variables that have been transformed in Section 4, we use spatial geometry to linearize the minimum distance prediction equation within a specific sampling period. Subsequently, in Section 5, spatial vector geometry is used to quickly solve the set of equations with the same independent variables that have been transformed in Section 4. Section 6 presents the simulation results and relevant analyses. Finally, in Section 7, some conclusions are offered.

3. Mathematic Model

Here, the following motion for an arbitrary dual manipulator cooperative system is preset: manipulator A starts from \( P^\text{start}_A \) heading towards \( P^\text{target}_A \), in parallel, manipulator arm B is also moving on its initial path from the starting point \( P^\text{start}_B \) to the target point \( P^\text{target}_B \). As we consider generating reasonable geometrical paths for all manipulators, the processing of finding the optimal position points around the current positions in their Cartesian space with a distance of \( L_s \) will be focused on during each planning period. Figure 3 visualizes the core for building the mathematical model of our proposal with the circumvention strategy of the dual manipulator system in a certain sampling period \( t \sim (t + t_s) \). Assume the search step length during a sampling time interval \( t_s \) in the planning space for both manipulators is \( L_s^t \) and we make it constant during every sampling period before reaching the target \( L_s^t = L_s \ (t \in [1, 2, \ldots, N]) \).
A path-planning strategy. The upper part of Figure 4 (picture a and c) corresponds to the interval scenario, while the bottom half (b, d) falls under the interact scenario specified in Figure 4. Different initial paths and coexistence scenarios give rise to various choices of obstacle avoidance points \( P^t_{ie} \) \( (i \in \{A, B\}) \). For the interval scenario, the relative position vector \( (P^t_{Be} - P^t_{Ae}) \) between manipulator B and A is outside the working space of manipulator A (consisting of vectors \( (P^t_{Be} - P^t_{Ae}) \), \( (P^t_{Be} - P^t_{Ae}) \) and \( (P^t_{Be} - P^t_{Ae}) \) together). In this case, it is only necessary to consider how to reach its terminal \( P^t_{Ae} \) as quickly as possible while avoiding the obstacle to ensure that manipulator A has the safest tendency to move towards its base. However, when it comes to interact scenarios, the relative position vector is in the working space of manipulator A, then the first thing that should be considered for manipulator A is to evade immediately along the direction of its base and proceed to its target after it has wholly bypassed the danger area.
Figure 4. Three dual-manipulator coexistence scenarios. The blue dotted line represents the initial path of manipulator A, and the red dotted line represents the initial path of manipulator B. The blue area is the operation space of manipulator A, and the red area is the operation space of manipulator B. In the scenario of “Overlap”, the initial path of manipulator A is in the operation space of manipulator A and the initial path of manipulator B is in the operation space of manipulator B. In scenario “Interval”, the two initial paths overlap and the EEF points of the two manipulators should be driven back towards their points of departure. In scenario “Interact paths”, the avoidance strategy pushes the EEFs of the two manipulators into the working space of another manipulator, undoubtedly increasing the possibility of collision.

When the minimum distance between the two manipulators is smaller than a predefined value, the dual manipulators are considered in the danger area in which there is a high probability of collision. The overlap between the two working areas is defined as the danger area. Therefore, when the two manipulators move into the danger area, the EEFs of the dual manipulators should move towards their respective bases to avoid the possible collision. Under this circumstance, the two manipulators are regarded as obstacles to each other. According to Figure 4, the non-linear optimization equations set of dual manipulator path planning will be established:

$$\min \sum_{i \in \{A,B\}} \mu_i \left[ (\alpha_i V^i_{ie} + \beta_i (P^i_{ie} - P^i_{ie}')) \cdot (P^i_{ie} - P^{i+1}_{ie})^T + \| P^{i}_{target} - P^{i+1}_{ie} \|^2 \right]$$

s.t.

$$P^i_{Ae} - P^{i+1}_{Ae} = \text{steplength}$$

$$P^i_{Be} - P^{i+1}_{Be} = \text{steplength}$$

$$d_{i+1,t} \geq L_{\text{min}}$$

where $V^i_{Ae}, V^i_{Be}$ are the linear velocities of EEFs of manipulator A and B at time $t$;

$P^i_{Ae}, P^i_{Be}$ are the Cartesian coordinates of EEFs of manipulator A and B at time $t$;

$P^{i+1}_{Ae}, P^{i+1}_{Be}$ are the Cartesian coordinates of EEFs of manipulator A and B at time $(t + t_s)$;

$P^i_{target}, P^i_{target}$ are the target Cartesian coordinates of manipulator A and B;

steplength indicates the length of the search step of the manipulators’ EEF within one sampling cycle $t_s$;

$d_{i+1,t} \geq L_{\text{min}}$ is the minimum distance between the two manipulators during sampling period $t \sim (t + t_s)$;

$L_{\text{min}}$ is the threshold of the minimum distance value;

$\mu_A, \mu_B$ are the positive circumventing weighting factors that meet $\mu_A + \mu_B = 1$.

$\mu_A = 0$ means manipulator A will always follow its initial path without modification and vice versa.

$\alpha_A, \alpha_B, \beta_A$ and $\beta_B$ are positive deviation direction weighting coefficients for manipulators A and B, which take values in the set of $[0,1]$.

We simplify the Formula (1):
\[
\min \sum_{i \in (A,B)} \mu_i \left[ \begin{pmatrix} a_i V_{ie} + \beta_i \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T + \| P_{ie}^{target} - P_{ie}^{+t_i} \| ^2 \right] \\
\sim \min \sum_{i \in (A,B)} \mu_i \left[ \begin{pmatrix} a_i V_{ie} + \beta_i \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T + \left( P_{ie}^{target} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T \right] \right. \\
\sim \min \sum_{i \in (A,B)} \mu_i \left[ \begin{pmatrix} a_i V_{ie} + \beta_i \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T + \left( P_{ie}^{target} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T \right] \\
\sim \min \sum_{i \in (A,B)} \mu_i \left[ \begin{pmatrix} a_i V_{ie} + \beta_i \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T + \left( P_{ie}^{target} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T \right] \\
\sim \min \sum_{i \in (A,B)} \mu_i \left[ \begin{pmatrix} a_i V_{ie} + \beta_i \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T + \left( P_{ie}^{target} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T \right] \\
\left. \right] \\
\right. \\
\end{align*}

Here, we can set

\[
\begin{align*}
\alpha_1 &= \mu_A \left[ \begin{pmatrix} a_A V_{ie} + \beta_A \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T + \left( P_{ie}^{target} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T \right] ; \\
\alpha_2 &= \mu_B \left[ \begin{pmatrix} a_B V_{ie} + \beta_B \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T + \left( P_{ie}^{target} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T \right] ; \\
x_1 &= P_{ie}^{target} - P_{ie}^{+t_i} \left( \text{steplength} \right) ; \quad x_2 = P_{ie}^{target} - P_{ie}^{+t_i} \left( \text{steplength} \right) ; \\
\end{align*}
\]

The objective function in the initial mathematical model (1) can be simplified as:

\[
\min \quad a_1 \cdot x_1^T + a_2 \cdot x_2^T 
\tag{2}
\]

The restricted condition functions in the initial mathematical model (1) can be translated into:

\[
\begin{align*}
\| x_1 \| &= 1 \\
\| x_2 \| &= 1 \\
d_{\text{min}} \geq L_{\text{min}}
\end{align*}
\tag{3}
\]

For Formulas (2) and (3), if we can find a method to convert the inequality expressions of \( d_{\text{min}} \geq L_{\text{min}} \) into the inequality equation with \( (x_1, x_2) \) as a variable, the entire initial mathematical model can be converted into a vector optimization model with corresponding variables \( (x_1, x_2) \). Under this case, we can solve the vector optimization model using existing mature mathematical tools efficiently.

Next, we will describe in detail the specific methods for determining the values of \( \alpha_A, \alpha_B, \beta_A \) and \( \beta_B \). Back to Figure 4, there are three direct external factors that affect the obstacle avoidance strategy of an industrial manipulator: the current velocity of motion \( V_{ie} \), the target point \( P_{ie}^{\text{target}} \) and the base of the manipulator \( P_{ie}^{\text{base}} \). In Formula (1),

\[
\min \alpha_i V_{ie} \left( P_{ie}^t - P_{ie}^{+t_i} \right)^T \]

guarantees to maintain the motion inertia of the manipulator as much as possible during the current planning sampling period, \( \min \| P_{ie}^{target} - P_{ie}^{+t_i} \|^2 \)

drives the manipulator towards its target point along the feasible shortest path, and

\[
\min \beta_i \left( P_{ie}^{\text{base}} - P_{ie}^t \right) \cdot (P_{ie}^t - P_{ie}^{+t_i})^T
\]

gives the manipulator a tendency to go to its safest place—base. Take manipulator A for example, the geometric relationship between the three spatial vectors \( (P_{ie}^{target} - P_{ie}^{+t_i}), (P_{ie}^{target} - P_{ie}^t), \) and \( (P_{ie}^{base} - P_{ie}^t) \) can be used to determine the values of two variables \( \alpha_A \) and \( \beta_A \).

\[
P_{ie}^{target} - P_{ie}^{base} = \omega \ast (P_{ie}^{base} - P_{ie}^t) + \varphi \ast (P_{ie}^{target} - P_{ie}^t) + \tau \ast ((P_{ie}^{base} - P_{ie}^t) \times (P_{ie}^{target} - P_{ie}^t))
\tag{4}
\]

If \( \omega > 0 \) and \( \varphi > 0 \), \( (P_{ie}^{target} - P_{ie}^t) \) is in the region of space defined by vectors \( (P_{ie}^{base} - P_{ie}^t) \) and \( (P_{ie}^{base} - P_{ie}^t) \). In this case, we can define \( \alpha_A = 1, \beta_A = 0 \), which means the manipulator A can circumvent the vector \( (P_{ie}^{base} - P_{ie}^t) \), a vector representing the dangerous spatial positions of the current dual manipulator system, by simply following the current motion tendency.
If \( \omega < 0 \) \( \land \) \( \varphi < 0 \), \( (P^\text{target}_A - P^\text{base}_A) \) is out of the space region defined by vectors \( (P^\text{base}_A - P^\text{end}_A) \) and \( (P^\text{end}_B - P^\text{end}_A) \). In this case, the mere maintenance of the motion trend towards the termination points do not guarantee manipulator A circumventing vector \((P^\text{end}_B - P^\text{end}_A)\). Therefore, forgoing the motion option of heading towards the terminals and temporarily seeking avoidance towards the base of the manipulator A is a better motion strategy, making \( \alpha_A = 0, \beta_A = 1 \).

Such derivations and conclusions also apply to manipulator B.

4. Minimum Distance Prediction

Collision detection is a fundamental theoretical research area of robotics, focusing on establishing a reasonable manipulator model and calculating the minimum distance between the manipulator and the obstacle. Collision detection can generally be divided into static collision detection, dynamic collision detection, and continuous collision detection [24]. Static collision detection is for the case where the manipulator and the obstacle are in a static state. In this case, collision detection can be modeled as the process of computing the minimum distance between two static geometric models in Cartesian space. Dynamic collision detection is where at least one of the manipulators and the obstacle is moving. The dynamic collision detection problem can be transformed into static collision detection because continuous time can be discretized. Both the manipulator and the obstacle can be considered static at each time interval. Continuous collision detection uses time as a variable to study the trend of the minimum distance between the manipulator and the obstacle in a given time interval.

Obviously, by observing Formulas (2) and (3), the optimal function and condition functions in the optimization equations are all contain the same variable term \((x_1, x_2)\). However, the inequality function \(d^\text{min}_{t_1 + t_2} \geq L^\text{min}\) cannot connect with the optimization function via the common variables through the existing discrete collision detection algorithm. The static collision detection and dynamic collision detection methods are not satisfied to determine the relationship between the minimum distance and the movement trend of path-planned manipulators. Therefore, we need to find a continuous collision detection algorithm that transforms the constraint functions of the minimum distance into inequality functions that take \((x_1, x_2)\) as variables, so that the whole set of optimization equations can be transformed into a function set with the same vector variables, which will be easy for us to solve.

The first step for collision detection is robot modeling. For multi-axis chain manipulators, there are many mature modeling methods [25]. Here, we use the Sweep Sphere Volume (SSV) [26] method, which envelops each link of a manipulator’s arm with a capsule body.

For \(d^\text{min}_{t_1 + t_2} \geq L^\text{min}\), it is equivalent to:

\[
d^\text{min}_{t_1 + t_2} \geq L^\text{min} \iff d^\text{min}_t - \Delta d_{t_1 + t_2} \geq L^\text{min} \iff \Delta d_{t_1 + t_2} \leq d^\text{min}_t - L^\text{min}
\]  

(5)

where \(d^\text{min}_t\) is the shortest distance between manipulator and obstacle(s) at current time instant \(t\), so its value must be known, and \(L^\text{min}\) is a predefined value. Thus, we need to figure out how to express \(\Delta d_{t_1 + t_2}\) in some kind terms of \((x_1, x_2)\) in the dual manipulator situation. In the following content of this section, we will propose a novel minimum distance prediction (MDP) method to meet our requirements.

MDP method is derived from the algebraic geometry (AG) method of collision detection. Via the AG method, the problem of calculating the minimum distance between objects is transformed into the calculation of distance between abstract space geometries. From Figure 5, we can observe that the movements of the minimum-distance points of the two manipulators along their respective mechanical arms are continuous, and there exists only one minimum-distance point for each manipulator at a time. At the same time, the calculation of minimum distance between two manipulators can be simplified into the calculation of minimum distance between two space segments according to the principle of collision detection. Therefore, based on the knowledge of the positions and the
velocities of the two minimum-distance points under the current instant, we can predict the possible locations of the minimum-distance points and the distance between them in the next scanning period. Furthermore, the velocities of two manipulator EEFs and the positions of mechanical arms on which the two minimum-distance points will stand under the present moment can determine the variation of the minimum distance during the next scanning period.

\[ \Delta d_{t+t_s} = \frac{\mathbf{V}_{BA}^{t+t_s} \cdot (\mathbf{P}_{AC}^{t} - \mathbf{P}_{BC}^{t})}{\| \mathbf{P}_{BC}^{t} - \mathbf{P}_{AC}^{t} \|} \cdot t_s \]

Figure 5. Schematic of minimum distance prediction for dual manipulators system within the sampling period \( (t \sim t + t_s) \). \( O_1O_2 \) and \( O_3O_4 \) denote the minimum distance links for manipulator A and B, and \( \mathbf{P}_{AC}^{t} \) and \( \mathbf{P}_{BC}^{t} \) are the closest points on the two minimum distance links, respectively. \( \mathbf{V}_{A}^{t+t_s} \) and \( \mathbf{V}_{B}^{t+t_s} \) are the displacement velocity vectors of \( \mathbf{P}_{AC}^{t} \) and \( \mathbf{P}_{BC}^{t} \) at the current moment \( t \). Here, we simplify the non-linear movement of the minimum distance point along the link to a linear advection, so that \( \Delta d_{t+t_s} \) can be easily approximated using dynamic spatial geometry.

As shown in Figure 5, according to the current moving state of the two minimum-distance points of the two manipulators, we can get:

\[ \Delta d_{t+t_s} = \mathbf{V}_{BA}^{t+t_s} \cdot \mathbf{P}_{BA}^{t} \cdot t_s = \left( \mathbf{V}_{A}^{t+t_s} - \mathbf{V}_{B}^{t+t_s} \right) \cdot \mathbf{P}_{BA}^{t} \cdot t_s \]

where \( \mathbf{P}_{BA}^{t} = \mathbf{P}_{BC}^{t} - \mathbf{P}_{AC}^{t} \) and \( \mathbf{P}_{BA}^{t} \) represent the minimum distance point position vectors of manipulator A and B under the global coordinate system. According to the different types of manipulators, the relationships between \( \left( \mathbf{V}_{A}^{t+t_s} \& \mathbf{V}_{B}^{t+t_s} \right) \) and \( (x_1, x_2) \) are different, while the points \( \mathbf{P}_{AC}^{t} \) and \( \mathbf{P}_{BC}^{t} \) will change along with the manipulator body links with time.

Meanwhile, the relationship between the velocities of minimum-distance points and those of manipulator EEFs can be obtained referring to:

\[ \begin{align*}
\left( \mathbf{V}_{A}^{t+t_s} \right)^T &= \left( J_A J_{d_{\mathbf{A}}}^{-1} \right)^T \cdot \left( \mathbf{v}_{\mathbf{A}}^{t+t_s} \right)^T \\
\left( \mathbf{V}_{B}^{t+t_s} \right)^T &= \left( J_B J_{d_{\mathbf{B}}}^{-1} \right)^T \cdot \left( \mathbf{v}_{\mathbf{B}}^{t+t_s} \right)^T 
\end{align*} \]
where $V_{Ac}^{i+1}$ and $V_{Be}^{j+1}$ are the velocities of $P_{Ac}$ and $P_{Be}$; $V_{Ac}^{j+1}$ and $V_{Be}^{i+1}$ are the velocities of the EEFs of manipulator A and manipulator B, individually; $P_{Ac}$ is the coordinate of $P_{Ac}$ and $P_{Be}$ is the coordinate of $P_{Be}$.

$$
\Delta d_{i+1} = \frac{P_{BA}}{P_{BA}^T} \left( \left( V_{Ac}^{i+1} - V_{Be}^{j+1} \right)^T \cdot \Delta t \right)
= \frac{P_{BA}}{P_{BA}^T} \left[ \left( J_{Ac}^{-1} \right) \cdot \left( \left( V_{Ac}^{i+1} \right)^T - \left( J_{Be}^{-1} \right) \cdot \left( V_{Be}^{j+1} \right)^T \right) \cdot \Delta t \right]
= \frac{P_{BA}}{P_{BA}^T} \left[ \left( J_{Ac}^{-1} \right) \cdot \left( V_{Ac}^{i+1} \right)^T \cdot \Delta t \right] - \left[ \frac{P_{BA}}{P_{BA}^T} \left( J_{Be}^{-1} \right) \right]
\cdot \left[ \left( V_{Be}^{j+1} \right)^T \cdot \Delta t \right]
= \frac{P_{BA}}{P_{BA}^T} \left[ \left( J_{Ac}^{-1} \right) \cdot \left( \left( V_{Ac}^{i+1} \right)^T - \left( V_{Ac}^{i+1} \right)^T \right) \right] - \left[ \frac{P_{BA}}{P_{BA}^T} \left( J_{Be}^{-1} \right) \right]
\cdot \left[ \left( \left( V_{Be}^{j+1} \right)^T \right) \right]
\cdot \left( x_2^T \cdot \text{steplength} \right) + \left[ \frac{P_{BA}}{P_{BA}^T} \left( J_{Ac}^{-1} \right) \right]
\cdot \left( x_2^T \cdot \text{steplength} \right) \leq d_{i}^{min} - L_{min}
$$

Thus:

$$
\left[ \frac{P_{BA}}{P_{BA}^T} \left( J_{Be}^{-1} \right) \right] \cdot x_2^T + \left[ \frac{P_{AB}}{P_{AB}^T} \left( J_{Ac}^{-1} \right) \right] \cdot x_1^T \leq d_{i}^{min} - L_{min}^{\text{steplength}}
$$

(8)

Set:

$$
b_1 = \frac{P_{AB}}{P_{AB}^T} \left( J_{Ac}^{-1} \right); b_2 = \frac{P_{BA}}{P_{BA}^T} \left( J_{Be}^{-1} \right); D = \frac{d_{i}^{min} - L_{min}^{\text{steplength}}}{\text{steplength}}
$$

The initial mathematical model of Formula (1) can be simplified as:

$$
\begin{align}
\min \quad & a_1 \cdot x_1^T + a_2 \cdot x_2^T \\
\text{s.t.} \quad & b_1 \cdot x_1^T + b_2 \cdot x_2^T \leq D \\
& \| x_1^T \| = 1; \| x_2^T \| = 1;
\end{align}
$$

(9)

Obviously, it is a vector linear programming problem.

5. Sampling-Based Unfixed Orientation Search Method

This chapter will describe converting vector optimization mathematic models into a scalar optimization problem. For Formula (9), assume that $b_1 \cdot x_1^T = x$, $b_2 \cdot x_2^T = y$. Then, we try to transfer the target function into the function of the two independent variables, $x$ and $y$. Now we take $b_1 \cdot x_1^T = x$ as an example, the following operational formulas can be obtained based on the vector diagram shown in Figure 6:

$$
c_1 = \frac{x}{\| b_1^T \|}; \quad \omega_1 = (b_1 \times a_1) \times b_1;
$$

$$
\left( d_1 \right)_{\text{max}} = \sqrt{1 - \left( \frac{x}{\| b_1^T \|} \right)^2 \cdot \left( \frac{\omega_1}{\| \omega_1 \|} \right)}
$$

$$
\left( d_1 \right)_{\text{min}} = -\sqrt{1 - \left( \frac{x}{\| b_1^T \|} \right)^2 \cdot \left( \frac{\omega_1}{\| \omega_1 \|} \right)}
$$

$$
\min(a_1 \cdot x_1^T) = a_1 \cdot (c_1 + (d_1)_{\text{min}}) = a_1 \cdot \left( \frac{x}{\| b_1^T \|^2} \cdot b_1 - \sqrt{1 - \left( \frac{x}{\| b_1^T \|} \right)^2 \cdot \left( \frac{\omega_1}{\| \omega_1 \|} \right)} \right)
= \frac{a_1 b_1^T}{\| b_1^T \|^2} x - \frac{a_1 \omega_1^T}{\| b_1^T \| \| \omega_1 \|} \sqrt{\| b_1^T \|^2 - x^2} = a_1 x - \beta_1 \sqrt{\gamma_1^2 - x^2}
$$
where:
\[\alpha_1 = \frac{a_1 \cdot b_1^2}{||b_1||^2}; \beta_1 = \frac{a_1 \cdot \omega_1^2}{||b_1|| ||\omega_1||} > 0; \gamma_1 = ||b_1|| > 0\]
\[\alpha_2 = \frac{a_2 \cdot b_2^2}{||b_2||^2}; \beta_2 = \frac{a_2 \cdot \omega_2^2}{||b_2|| ||\omega_2||} > 0; \gamma_2 = ||b_2|| > 0\]

\[\vec{d}_1 \]min
\[\vec{d}_1 \]max
\[\vec{x}_1 \]min
\[\vec{x}_1 \]max
\[\vec{a}_1\]
\[\vec{b}_1\]
\[\vec{e}\]
\[\vec{\omega}\]

Figure 6. The diagram of vector conversion relation of the optimal Equation (9).

Therefore, the initial mathematical model will eventually be transferred into the following form:

\[
\begin{align*}
\min & \quad \alpha_1 x - \beta_1 \sqrt{\gamma_1^2 - x^2} + \alpha_2 y - \beta_2 \sqrt{\gamma_2^2 - y^2} \\
\text{sub to} & \quad x + y \leq D
\end{align*}
\] (10)

where:
\[a_1 = \mu_A \left[ \frac{a_A \cdot v^e_A + \beta_A (I_{A}^{\text{base}} - P_{Ae}^e)}{||v_A^{\text{base}} + \beta_A (I_{A}^{\text{base}} - P_{Ae}^e)||} + \left( P_{Ae}^e - P_{Ae}^i \right) \right] ;
\]
\[a_2 = \mu_B \left[ \frac{a_B \cdot v^e_B + \beta_B (I_{B}^{\text{base}} - P_{Be}^e)}{||v_B^{\text{base}} + \beta_B (I_{B}^{\text{base}} - P_{Be}^e)||} + \left( P_{Be}^e - P_{Be}^i \right) \right] ;
\]
\[D = \text{apopt-length}_{\text{min}} ;
\]
\[x_1 = \frac{p_{Ae}^e - p_{Ae}^{i-1}}{\text{step-length}}, x_2 = \frac{p_{Be}^e - p_{Be}^{i-1}}{\text{step-length}}, b_1 = \frac{p_{Ae}^i}{||p_{Ae}||} (I_A^{\text{base}}) ; b_2 = \frac{p_{Be}^i}{||p_{Be}||} (I_B^{\text{base}}) ;
\]
\[\alpha_1 = \alpha_1 \cdot b_1^2 ; \beta_1 = \alpha_1 \cdot \omega_1^2 ; \gamma_1 = ||b_1|| ; \alpha_2 = \alpha_2 \cdot b_2^2 ; \beta_2 = \alpha_2 \cdot \omega_2^2 ; \gamma_2 = ||b_2|| ;
\]
\[x = b_1 \cdot x_1^i ; y = b_2 \cdot x_2^i
\]

Apparently, Formula (10) is a binary non-linear optimal solution equation that can be easily solved using "nlopt" open-source software library in the vs. programming platform.

6. Simulation and Analysis

In this paper, assuming 6-DOF industrial manipulators have standard D-H parameters, the simulation results for the dual manipulator systems will be analyzed individually. All the algorithms were programmed in C++ and run on a 64-bit Windows operation system with an i7-4790 CPU with 8 GB RAM. All coordinate values in this article are in millimeters.

6.1. Simulation of Single Manipulator Avoidance

When \(\mu_A = 0\) or \(\mu_B = 0\), the dual manipulators avoidance strategy will be converted to a single manipulator avoidance strategy. Here, we define \(\mu_A = 0\) to show the simulation.
results of single manipulator avoidance. The space coordinates of the base of manipulator A are (0, 0, 0) while manipulator B’s base is situated at (0, -900, 0). Six different path planning cases and the corresponding simulation results are presented and compared in Figures 7 and 8 intensively.

**Figure 7.** Comparison of single manipulator avoidance simulation results for four initial path cases. As the simulation results are 3D path curves, the results are presented in a comprehensive and extensive visualization with four views: axonometric, top, front, and side views.

**Figure 8.** Comparison of single manipulator avoidance simulation results for three initial path cases. These three cases yield three completely different simulation results by changing the coordinates of manipulator A on the Y-axis of the WCS while maintaining the same spatial position of manipulator B.
Case 1: the EEF of manipulator B stays at $P_B = (0, -450, 500)$ while manipulator A moves from $P_A^{start} = (400, -450, 500)$ towards $P_A^{end} = (-400, -450, 500)$.

Case 2: the EEF of manipulator B stays at $P_B = (100, -450, 500)$ while manipulator A moves from $P_A^{start} = (400, -450, 600)$ towards $P_A^{end} = (-400, -450, 400)$.

Case 3: the EEF of manipulator B stays at $P_B = (0, -450, 500)$ while manipulator A moves from $P_A^{start} = (400, -400, 500)$ towards $P_A^{end} = (-400, -500, 500)$.

Case 4: the EEF of manipulator B stays at $P_B = (0, -450, 500)$ while manipulator A moves from $P_A^{start} = (400, -450, 400)$ towards $P_A^{end} = (-400, -450, 600)$.

Case 5: the EEF of manipulator B stays at $P_B = (0, -500, 500)$ while manipulator A moves from $P_A^{start} = (400, -450, 500)$ towards $P_A^{end} = (-400, -450, 500)$.

Case 6: the EEF of manipulator B stays at $P_B = (0, -400, 500)$ while manipulator A moves from towards $P_A^{end} = (400, -500, 500)$ towards $P_A^{end} = (-400, -500, 500)$.

We can simulate the effect of the algorithm proposed in this paper on the path-planning strategy of a dual manipulator system with different initial paths by using the control variables method. Take Case 1 as a benchmark case, both Cases 2 and 4 add displacement in the Z-axis direction of the world coordinate system (WCS), distinguishing the two by the opposite direction of displacement. The displacement of the manipulator A in the Z-axis is a descent from the high to the low coordinate in Case 2, while in Case 4 the situation is reversed. Similarly, and in turn, Case 3 incorporates a Y-axis movement.

As the working space of an industrial manipulator is a finite three-dimensional space centered on the base of the manipulator, an avoidance strategy along the direction to the base is always the safest option for the obstacle avoidance of the manipulator. Figure 8 illustrates the path patterns of three different dual manipulator systems and the respective avoidance strategies. It is clear that Cases 1, 5, and 6 in Figure 7 are simulation avoidance results for each of these three path patterns.

The space coordinate vector of an industrial manipulator arm is usually comprised of two components $P = ([x, y, z], [α, β, γ])$, where $[x, y, z]$ represents the 3D space coordinates of the centre point of the manipulator EEF and $[α, β, γ]$ represents the pose direction vector of the manipulator EEF. The EEF pose vectors of the manipulator A and B are assigned the values $[α_A, β_A, γ_A] = [π/2, -π/2, 0]$ and $[α_B, β_B, γ_B] = [-π/2, π/2, 0]$ for all cases in Figures 7 and 8.

In order to verify the generality of our algorithm, we re-simulated all cases 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 $[α_A, β_A, γ_A] = [π/2, π, 0]$ and $[α_B, β_B, γ_B] = [-π/2, π/2, 0]$, the corresponding simulation results are shown in Figure 9.

By comparing the simulation results for all cases in Figures 7–9, some apparent and obvious consensus can be quickly reached. Despite different EEF pose vectors and different initial motion states, the algorithm proposed in this paper always can adaptively generate suitable and reasonable obstacle avoidance paths. However, due to changes in the EEF pose vectors, our algorithm also produces significantly different simulated planning paths for the same case. For example, in Cases 1 and 3, the simulation results in Figures 7 and 8 both show apparently large ‘bends’ in the Z-axis of the WCS and relatively less pronounced ‘bends’ in the X-Y plane, whereas in Figure 9 the simulated paths have only small ‘fluctuations’ along the Z-axis and large bends in the X-Y plane. Furthermore, the simulated path results for Cases 2 and 4 in Figures 7 and 9 are exactly opposite in the “bending” direction along the Z-axis. This discrepancy is directly due to the fact that various pose vectors of the EEF directly affect the space occupation of the manipulator bodies or in particular the end links, thus leading our algorithm to adaptively plan different obstacle avoidance paths under the planning principle of shortest paths when facing different space occupancies.
Figure 9. Comparison of single manipulator avoidance simulation results. Compared to Figures 7 and 8, the six cases in this diagram merely modify the end-effector pose vectors of the manipulators, whilst the simulation results obtained are significantly distinctive.

Retrospectively, Figure 1 shows that the insights advanced in this paper are improved by the progressive sequence of the map search method → SbOSMS → SbUOSM. Thus, it is essential to undertake comparative simulations of these three algorithms. Figure 10 compares the simulation results of Case 1 in Figure 8 utilizing the three methods described.
above. Equivalently, Figures 11 and 12 correspond to the comparative simulation results for Cases 5 and 6 in Figure 8, respectively.
By detailed comparison of Figures 10–12, some obvious observational conclusions can be summarized briefly. In terms of the softness of the planned path curve, the proposal in this paper takes speed into account as a factor in the optimization and thus achieves a smoother solution than the map search method and SbOSMS, which only consider planning space path points. When it comes to the spatial complexity of planning path curves, the paths planned by the SbOSMS algorithm are only offset in the X-Y plane in the WCS in all three cases, the map search method (A*) is shifted in all axis directions in space, while the SbUOSM algorithm is only biased in the X-Y plane in case 1, and the remaining cases are similar to A*. This is a side effect of the greater flexibility with which our proposals operate in the face of different scenarios.

The total deviation of the planned path from the initial path 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:

\[
\Delta g \rightarrow g_0 = \int_0^L \| f_0(t) - f(t) \| dt \tag{11}
\]

All the planning results in Figures 10–12 were used to calculate the total deviation utilizing the above equation and the results were assembled into Table 1.

### Table 1. Summary table of planning path deviations for the three algorithms under all six cases (Unit: mm).

|                | Case 1 | Case 2 | Case 3 | Case 4 | Case 5 | Case 6 |
|----------------|--------|--------|--------|--------|--------|--------|
| SbUOSM         | 4899.4 | 7183.7 | 7619.6 | 8185.9 | 7586.8 | 10,844 |
| A* in C-space  | 13,161 | 20,914 | 19,047 | 21,187 | 16,150 | 19,153 |
| SbOSMS         | 5814.5 | 8461.6 | 9015.4 | 8794.1 | 8629.2 | 11,878 |
Meanwhile, we simulate each of these three algorithms in the six cases mentioned above and assemble the respective algorithm execution time and the number of planning path steps into Table 2.

**Table 2.** Summary table of algorithm execution times/total planning steps for the three algorithms under the six cases (Unit: \(\mu s\)/step).

| Case | SbUOSM | A* in C-space | SbOSMS |
|------|--------|---------------|--------|
| 1    | 6876.7/95 | 45,992.6/145 | 13,477/95 |
| 2    | 6825.9/93 | 38,291.4/131 | 13,313.9/93 |
| 3    | 7703.3/95 | 48,498.9/151 | 13,148.3/97 |
| 4    | 7045.5/96 | 38,153.8/131 | 13,340.8/97 |
| 5    | 6405.9/88 | 42,050.7/138 | 12,391/89 |
| 6    | 8892.3/121 | 46,764.1/142 | 17,043.8/112 |

Comprehensive analysis of Tables 1 and 2 clearly shows that the proposal in this paper is significantly better than the map search method and the SbOSMS algorithm in terms of the deviation of the planning path, the execution time of the algorithm and even the number of planning steps.

**6.2. Simulation of Dual Manipulator Avoidance**

Similar to single manipulator avoidance, 5 cases will be used to simulate the operation of the SbUOSM algorithm in a dual manipulator avoidance scenario.

Case 1: the EEF of manipulator A moves from \(P^\text{start}_A = (400, -450, 500)\) towards \(P^\text{end}_A = (-400, -450, 500)\) while manipulator A moves from \(P^\text{start}_B = (-400, -500, 500)\) towards \(P^\text{end}_B = (400, -500, 500)\).

Case 2: the EEF of manipulator A moves from \(P^\text{start}_A = (400, -450, 500)\) towards \(P^\text{end}_A = (-400, -450, 500)\) while manipulator A moves from \(P^\text{start}_B = (-400, -450, 500)\) towards \(P^\text{end}_B = (400, -450, 500)\).

Case 3: the EEF of manipulator A moves from \(P^\text{start}_A = (400, -450, 500)\) towards \(P^\text{end}_A = (-400, -450, 500)\) while manipulator A moves from \(P^\text{start}_B = (-400, -450, 500)\) towards \(P^\text{end}_B = (400, -450, 500)\).

Case 4: the EEF of manipulator A moves from \(P^\text{start}_A = (400, -400, 500)\) towards \(P^\text{end}_A = (-400, -400, 500)\) while manipulator A moves from \(P^\text{start}_B = (-400, -400, 500)\) towards \(P^\text{end}_B = (400, -400, 500)\).

Case 5: the EEF of manipulator A moves from \(P^\text{start}_A = (400, -450, 400)\) towards \(P^\text{end}_A = (-400, -450, 400)\) while manipulator A moves from \(P^\text{start}_B = (-400, -450, 400)\) towards \(P^\text{end}_B = (400, -450, 400)\).

Similarly, when we refer to Case 2 as the benchmark case, based on the fact that both manipulators have relative motions in the dual manipulator avoidance, Cases 2 and 4 in Figure 8 are fused into Case 5 in Figure 13, which is used to show how this paper’s proposal operates when adding the displacement of the initial path in the Z-axis direction of the WCS; In Figure 13, Cases 1–3 investigate the effect of the relative distance between two manipulators on the planning results of the algorithm, similar to cases 1, 5 and 6 in single manipulator avoidance; Case 4 intersects the paths of the two manipulators in the X-Y plane of the world coordinate system, similar to Case 3 in Figure 8, aiming to reveal the specific simulation results of the algorithm for displacement increments in the Y-axis direction in the WCS.

As the effect of changing the end-effector pose vector on the planning results has already been shown in the section on single-arm avoidance (Figure 9), this variable will not be modeled in this chapter.

Figures 14–16 focus on the difference between the results of the three different algorithms. The analogous results for dual manipulator avoidance are similar to those for single manipulator avoidance. SbUOSM can always produce smoother paths. The results of the map search algorithm are always the most space-complex. Except for Figure 16, both SbUOSM and SbOSMS perform avoidance only in the X-Y plane of the world coordinate system. In Figure 16, the path points of SbUOMS are all distributed in the X-Z plane,
mainly because the dimensions of the end links are more distributed in the X-Y plane than in the X-Z plane.

Figure 13. Comparison of dual manipulators avoidance simulation results for five initial path cases. As the simulation results are 3D path curves, the results are presented in a comprehensive and extensive visualization with four views: axonometric, top, front and side views.
mainly be cause the dimensions of the end links are more distributed in the X-Y plane than in the X-Z plane.

Figure 14. Comparison of simulation results of three path-planning algorithms for Case 1.

Figure 15. Comparison of simulation results of three path-planning algorithms for Case 2.
Figure 15. Comparison of simulation results of three path-planning algorithms for Case 2.

Axonometric view
Top view
Side view
Front view

Figure 16. Comparison of simulation results of three path-planning algorithms for Case 3.

All the planning results in Figures 14–16 were used to calculate the total deviation utilizing the above equation, and the results were assembled in Table 3.

Table 3. Summary table of planning path deviations for the three algorithms under all six cases (Unit: mm).

| Case   | SbUOSM  | A* in C-space | SbOSMS |
|--------|---------|---------------|--------|
| Case 1 | 3944.9  | 8341.2        | 4280.3 |
| Case 2 | 5457.9  | 9276.3        | 6162.2 |
| Case 3 | 5647.5  | 9373.5        | 8126.8 |
| Case 4 | 4857.7  | 8887.8        | 5964.7 |
| Case 5 | 5846.8  | 9006.1        | 6765.3 |

Meanwhile, we simulate each of these three algorithms in the six cases mentioned above and assemble the respective algorithm execution time and the number of planning path steps into Table 4.

Table 4. Summary table of algorithm execution times/total planning steps for the three algorithms under the six cases (Unit: µs/step).

| Case   | SbUOSM  | A* in C-space | SbOSMS |
|--------|---------|---------------|--------|
| Case 1 | 16,650.6/90 | 16,029.7/88  | 50,392.1/85  |
| Case 2 | 16,344.1/88 | 67,822.4/108 | 58,154.9/91  |
| Case 3 | 18,334.9/100| 65,255.6/106 | 66,888.6/92  |
| Case 4 | 18,641.7/103| 69,112.5/107 | 54,660.6/91  |
| Case 5 | 67,594.8/104| 67,954.8/107 | 57,083.3/92  |

Comprehensive analysis of Tables 3 and 4 clearly shows that the proposal in this paper is significantly better than the map search method and the SbOSMS algorithm in terms of the deviation of the planning path, the algorithm’s execution time, and even the number of planning steps.

From Equation (1) we can also visualize that the planning results are also directly governed by the weighting factors $\mu_A$ and $\mu_B$. Figures 17 and 18 illustrate visually the
effect of different values of $\mu_A$ on the shape of the end-effector paths of manipulators A and B, respectively, in the form of multicolored graphs. In Figures 17 and 18, the horizontal axis is the coordinate of the end of the corresponding manipulator under the X axis of the WCS, and the vertical axis is the coordinate of the end of the corresponding manipulator under the Y axis of the WCS. Due to the constraint $\mu_A + \mu_B = 1$, when $\mu_A$ increases, $\mu_B$ decreases. Observing Figures 17 and 18 together, as $\mu_A$ gradually increases from 0.1 to 0.9, the maximum offset of the path planned at the end of manipulator arm A on the Y-axis is incrementally reduced, and the opposite is true for manipulator B. From the geometric properties of the path curve, the curvature of the path of manipulator A enlarges and the curvature of the path of manipulator B diminishes.

![Figure 17](image1.png)

**Figure 17.** For Case 1 in Figure 13, the paths of the manipulator A EEF are simulated for different values of the positive circumventing weighting factors $\mu_A$ and $\mu_B$.

![Figure 18](image2.png)

**Figure 18.** For Case 1 in Figure 13, the paths of the manipulator B EEF are simulated for different values of the positive circumventing weighting factors $\mu_A$ and $\mu_B$.  

7. Conclusions

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

This paper inherits the main architecture of the SbOSMS algorithm, which also combines the map search method and the time-sampling method, and merely upgrades the local point map in a single sampling period to a global spherical map. To this end, a system of multivariate vector inequality equations was constructed based on the spatial geometry information and motion properties of the dual manipulator system. However, after preliminary simplifications, the initial set of objective functions and constraint equations for vector inequalities are not directly convertible into a set of equations with the same variables. This paper converts all the equations into a system of equations with common vector variables by proposing a new minimum distance prediction method to linearly fit the minimum distance curve of a dual manipulator system. The three-dimensional properties of the variable vectors are then used to spatially geometrify the entire set of equations, using spatial geometry to reduce the entire set of vector programming equations to scalar non-linear optimization equations that can be easily solved. By comparing the simulation results of single manipulator avoidance and dual manipulator avoidance for various scenarios, the proposal in this paper can obviously be superior to SbUOSM and the earlier map search method from different perspectives such as the avoidance of path complexity degree, deviation, algorithm execution time, and the number of planning steps.

However, the paper has not probed further into how much the linear fitting treatment of the minimum distance curve for the dual manipulator system actually affects the entire planning result, which could be one of the future research points. At the same time, the application of this proposal to multi-manipulator systems is also a direction in which work needs to be invested.

Author Contributions: C.S. conceived of the study, designed the study, and wrote the manuscript. J.X. provides guidance on ideas and mathematical treatment. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by Bosch (China) Investment Ltd. “Smart Manufacturing” international cooperation projects with Huazhong University of Science & Technology (Grant No.: 0232100008).

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

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Caccavale, F.; Uchiyama, M. Cooperative Manipulation. In Springer Handbook of Robotics; Siciliano, B., Khatib, O., Eds.; Springer: Cham, Switzerland. [CrossRef]
2. Jonas, S.; Verena, K.; Christin, H.; Klaus, B. Human Centered Assistance Applications for the working environment of the future. Occup. Ergon. 2015, 12, 83–95.
3. Iina, A.; Timo, S.; Ilari, M. Refining levels of collaboration to support the design and evaluation of human-robot interaction in the manufacturing industry. In Proceedings of the 51st CIRP Conference on Manufacturing Systems, Stockholm, Sweden, 16–18 May 2018; Volume 72, pp. 93–98.
4. Choi, M.; Lim, S.; Lee, J. Cooperative Path Planning for Redundant Dual-Arm Robot Using Low-Dimensional Sample-Based Algorithm. Mechatron. Syst. 2010, 43, 701–708. [CrossRef]
5. Nash, A.; Koenig, S.; Tovey, C. Lazy theta*: Any-angle path planning and path length analysis in 3D. In Proceedings of the Third Annual Symposium on Combinatorial Search, Atlanta, GA, USA, 8–10 July 2010; Volume 2, pp. 153–154.
6. Jia, Q.; Chen, G.; Sun, H.; Zheng, H. Path planning for space manipulator to avoid obstacle based on A* algorithm. Chin. J. Mech. Eng. 2010, 46, 109–115. [CrossRef]
7. Yu, N.; Wang, Z. Collision avoidance planning of manipulator based on C-space layered search arithmetic. In Proceedings of the International Conference on Electronic and Mechanical Engineering and Information Technology (EMEIT), Harbin, China, 12–14 August 2011; pp. 3258–3262.

8. Li, L.; Shi, Z.; Guan, Y.; Zhao, C.; Zhang, J.; Wei, H. Formal verification of a collision-free algorithm of dual-arm robot in HOL4. In Proceedings of the IEEE International Conference on Robotics & Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 1380–1385.

9. Harada, K.; Foissotte, T.; Tsubi, T.; Nagata, K.; Yamanobe, N.; Nakamura, A.; Kawai, Y. Pick and place planning for dual-arm manipulators. In Proceedings of the IEEE International Conference on Robotics and Automation, Saint Paul, MN, USA, 14–18 May 2012; pp. 2281–2286.

10. Lee, S.; Moradi, H.; Yi, C. A real-time dual-arm collision avoidance algorithm for assembly. In Proceedings of the IEEE International Symposium on Assembly and Task Planning, Marina del Rey, CA, USA, 7–9 August 1997; pp. 7–12.

11. Larsen, L.; Pham, V.L.; Kim, J.; Kupke, M. Collision-free path planning of industrial cooperating robots for aircraft fuselage production. In Proceedings of the IEEE International Conference on Robotics and Automation, Seattle, WA, USA, 25 May 2015; p. 2042.

12. Cohen, B.; Chitta, S.; Likhachev, M. Single- and dual-arm motion planning with heuristic search. *Int. J. Robot. Res.* 2014, 33, 305. [CrossRef]

13. Chiddarwar, S.S.; Ramesh, B.N. Conflict free coordinated path planning for multiple robots using a dynamic path modification sequence. *Robot. Auton. Syst.* 2011, 59, 508. [CrossRef]

14. Afaghani, A.Y.; Atyama, Y. Advanced-collision-map-based online collision and deadlock avoidance between two robot manipulators with PTP commands. In Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE), New Taipei, Taiwan, 18–22 August 2014; pp. 1244–1251.

15. Rodríguez, C.; Montano, A.; Suarez, R. Planning manipulation movements of a dual-arm system considering obstacle removing. *Robot. Auton. Syst.* 2014, 62, 1816. [CrossRef]

16. Korayem, M.H.; Nekoo, S.R. The SDRE control of mobile base cooperative manipulators: Collision free path planning and moving obstacle avoidance. *Robot. Auton. Syst.* 2016, 86, 86–105. [CrossRef]

17. Yang, L.; Qi, J.; Song, D.; Xiao, J.; Han, J.; Xia, Y. Survey of Robot 3D Path Planning Algorithms. *J. Control. Sci. Eng.* 2016, 1, 22. [CrossRef]

18. Zhu, X.; Yan, B.; Yue, Y. Path Planning and Collision Avoidance in Unknown Environments for USVs Based on an Improved D* Lite. *Appl. Sci.* 2021, 11, 7863. [CrossRef]

19. Faria, M.; Marin, R.; Popović, M.; Maza, I.; Viguria, A. Efficient Lazy Theta* Path Planning over a Sparse Grid to Explore Large 3D Volumes with a Multirotor UAV. *Sensors* 2019, 19, 174. [CrossRef] [PubMed]

20. Luo, M.; Hou, X.; Yang, J. Surface Optimal Path Planning Using an Extended Dijkstra Algorithm. *IEEE Access* 2020, 8, 147827. [CrossRef]

21. Schömer, P.; Hüneberg, M.T.; Zöllner, J.M. Optimization of Sampling-Based Motion Planning in Dynamic Environments Using Neural Networks. In Proceedings of the IEEE Intelligent Vehicles Symposium (IV), Las Vegas, NV, USA, 19 October–13 November 2020; pp. 2110–2117.

22. Wang, J.; Li, C.X.-T.; Chi, W.; Meng, M.Q.-H. Tropistic RRT*: An Efficient Planning Algorithm via Adaptive Restricted Sampling Space. In Proceedings of the IEEE International Conference on Information and Automation (ICIA), Wuyishan, China, 11–13 August 2018; 2018; pp. 1639–1646.

23. Yang, S.M.; Lin, Y.A. Development of an Improved Rapidly Exploring Random Trees Algorithm for Static Obstacle Avoidance in Autonomous Vehicles. *Sensors* 2021, 21, 2244. [CrossRef] [PubMed]

24. Heo, Y.J.; Kim, D.; Lee, W.; Kim, H.; Park, J.; Chung, W.K. Collision Detection for Industrial Collaborative Robots: A Deep Learning Approach. *IEEE Robot. Autom. Lett.* 2019, 4, 740–746. [CrossRef]

25. Casalino, A.; Brameri, A.; Zanchettin, A.M.; Rocco, P. Adaptive swept volumes generation for human-robot coexistence using Gaussian Processes. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; p. 3823.

26. Lumelsky, V.J. On fast computation of distance between line segments. *Inf. Processing Lett.* 1985, 21, 55–61. [CrossRef]