aerospace

MDPI

Article

A Novel Method for Inverse Kinematics Solutions of Space Modular Self-Reconfigurable Satellites with Self-Collision Avoidance

Jiping An, Xinhong Li *, Zhibin Zhang, Guohui Zhang, Wanxin Man, Gangxuan Hu, Junwei He and Dingzhan Yu

Department of Aerospace Science and Technology, Space Engineering University, Beijing 101416, China; ajp112233@alumni.sjtu.edu.cn (J.A.); zhangzhibinseu@163.com (Z.Z.); zhangguohui123@stu.xjtu.edu.cn (G.Z.); manwanxin@126.com (W.M.); hgx0609@163.com (G.H.); junwei.online@gmail.com (J.H.); 18811138591@139.com (D.Y.)

* Correspondence: 13366159269@189.cn

Abstract: Space modular self-reconfigurable satellites (SMSRSs) are a new type of satellite with reconfigurable structures and adjustable functions. The inverse kinematics of the hyper-redundant structure of SMSRSs are difficult to solve by traditional methods. In this paper, the inverse kinematics of SMSRS is transformed into an optimization problem and solved using the optimization method. Moreover, the avoidance of self-collision is implemented in the optimization process. Firstly, the kinematic model of SMSRS is established. Then, to find the more accurate inverse kinematics solutions, a novel Segmented Hybrid CMA-ES and PSO (SHCP) algorithm is proposed. The algorithm is used for three cases of inverse kinematic problems, and the optimization results prove the optimization method is effective to solve the inverse kinematic problem with self-collision avoidance. Compared to the results of PSO variants, meta-heuristic algorithms, and hybrid algorithms, the novel algorithm has higher accuracy, proving its better performance on the inverse kinematics problem of SMSRS.

Keywords: inverse kinematics; hybrid algorithm; self-reconfigurable satellite; self-collision avoidance; meta-heuristic algorithm

check for updates

Citation: An, J.; Li, X.; Zhang, Z.; Zhang, G.; Man, W.; Hu, G.; He, J.; Yu, D. A Novel Method for Inverse Kinematics Solutions of Space Modular Self-Reconfigurable Satellites with Self-Collision Avoidance. *Aerospace* **2022**, *9*, 123. https://doi.org/10.3390/aerospace9030123

Academic Editor: George Z.H. Zhu

Received: 29 December 2021
Accepted: 24 February 2022
Published: 28 February 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The structure of conventional spacecraft is usually permanently fixed to perform a single task and cannot be reconfigured for other tasks. At the same time, current spacecraft have difficulty responding quickly to unexpected events in space due to the limitations of long development cycles and launch windows [1]. To address this limitation, future spacecraft are expected to be multipurpose and be able to respond to unforeseen events. The research object of this paper, space modular self-reconfigurable satellites (SMSRSs), is proposed to meet this expectation. An SMSRS is a space system with morphological adaptability and functional reconfigurability. It carries multiple types of task subsystems, and these subsystems are arranged and reorganized at different space orientations to complete multiple types of space tasks. For example, when an SMSRS carries multiple space cameras, it could change the space orientations of these cameras through joint motions, achieve an expanded imaging area by stitching together the field of view of these cameras or achieve the reconnaissance of multiple targets.

An SMSRS is a chain structure composed of different functional modules and joints. In Figure 1, an SMSRS with five modules is shown. Each module has a standard interface and regular shape, and it is the integration of payloads or satellite subsystems. Modules of the SMSRS are scalable, and they are functionally independent and physically separated from each other. The standard interfaces are used to connect with modular joint modules for ground or in-orbit extension according to the type of space mission. The type and number of modules carried by the SMSRS are configured to mission requirements. The modular design of the SMSRS facilitates the reconfiguration in space and the folding and packaging before launch [2]. The modules of the SMSRS are connected by three rotating joints whose axes are orthogonal to each other to form degrees of freedom (DOFs) relative to rotation, rolling, and yaw directions so that the SMSRS can adaptively adjust its configuration. Unlike other types of reconfigurable satellites that adjust their configuration through robotic arms, external service robots, or autonomous flight, the SMSRS adjusts its configuration through joint motion.

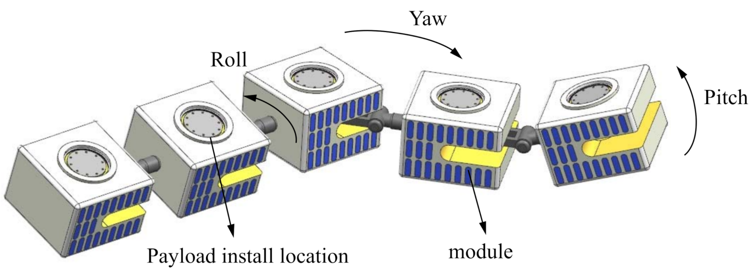


Figure 1. Model of SMSRS with five modules.

The SMSRS switches the task capability by structural reconfiguration. During task switching, the SMSRS should be converted from the current configuration to target configuration by joint rotations, where a key aspect is to calculate the angle of rotation required of each joint. Mathematically, it involves converting the desired module poses of the SMSRS in Cartesian space to the joint angles in joint space, also known as the inverse kinematics problem. The solution of inverse kinematics is indispensable for the motion analysis and control of the SMSRS [3].

There are various methods to solve inverse kinematics of SMSRSs, such as the geometric method [4], algebraic method, iterative method [3], and optimization method [5]. Among them, the geometric and algebraic methods are collectively called analytical methods. The geometric method is to obtain the geometric relationship between the poses of the links and the joint angles by solving the trigonometric functions. However, the geometric method is only suitable for systems with few DOFs, so it is not generalized. The idea of the algebraic method is to convert the kinematic equations into high-order polynomials, and then solve for all the roots of these polynomials. However, the algebraic method usually does not yield a clear indication of the validity of these solutions. Users usually have to rely on intuition to select the correct solutions [4]. Despite its fast solving speed and high accuracy, the analytical method is only applicable to manipulators with simple structures. For complex, geometrically structured systems, the solution of their nonlinear trigonometric equations is too complex and does not always have analytical expressions, so the analytical solution cannot be guaranteed [6]. In this case, it is necessary to use the iterative method and optimization method.

The iterative method is mainly based on the Jacobian matrix [7] and is regardless of the number of DOFs. However, as the number of DOFs increases, iterative methods become increasingly expensive to compute [8] and also suffer from singularity problems [9].

For the inverse kinematic problems that are not suitable for the above-mentioned traditional methods, they can be solved by the optimization method, where the optimal joint of inverse kinematic problems is found by optimization algorithms. The optimization method is particularly suitable for the complex inverse kinematic problems because its calculation cost and time is not significantly increased due to the increase in DOFs, and it is based on the forward kinematics equation, so joint singularities can be avoided [7]. When adding additional constraints to the multi-solution inverse kinematics, the corresponding optimal solutions for different objectives can be obtained. Many researchers have attempted to solve inverse kinematics with a variety of meta-heuristic algorithms, including the Genetic algorithm (GA) [10,11], the Artificial Bee Colony algorithm (ABC) [12,13], the Particle Swarm

Optimization algorithm (PSO) [14,15], the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [16] and the Differential Evolution algorithm (DE) [17,18], et al.

In inverse kinematics of SMSRS, different from other self-reconfiguration systems (e.g., space manipulators), the SMSRS is concerned with the pose of multiple modules and not only the pose of the end-effector. In addition, due to the large size of the modules, it is necessary to consider the avoidance of self-collision of the SMSRS. Along with the redundant DOFs, the inverse kinematics of SMSRS is too difficult to be solved by traditional methods, so we used the optimization method to solve it. In addition, due to the specificity of the space task, the accuracy of the SMSRS solution is required to be higher. To solve the inverse kinematics of SMSRS with higher accuracy, we propose a new segmented hybrid CMA-ES and PSO (SHCP) algorithm in this paper and consider the self-collision avoidance of the SMSRS in the inverse kinematics problem.

The rest of the paper is organized as follows. In Section 2, the kinematic model of SMSRS is established. In Section 3, the objective function of inverse kinematics with self-collision avoidance of SMSRS is established, and detailed optimization steps are proposed. The SHCP algorithm is proposed in Section 4. In Section 5, the SHCP algorithm is used to optimize the inverse kinematic problem of SMSRS with different requirements on module poses, and the optimization results are compared with other PSO variants, meta-heuristic and hybrid algorithms. Section 6 discusses and summarizes the work.

2. Kinematics Modeling of SMSRS

Kinematics describes the relationship between the coordinates of the joint space and the task space. The forward kinematics map the joint space to the task space, while the inverse kinematics is the opposite mapping [12]. For an SMSRS, once a task is started, the task planning system determines its task configuration and communicates the reconfiguration requirements to the control system of the SMSRS. These requirements are usually given in the form of the absolute or relative pose of task modules. This process depends on the establishment of forward kinematics and the solution of inverse kinematics.

Forward Kinematiccs of SMSRS

An SMSRS consists of a series of rigid bodies and its modules and inter-module connection structures can be regarded as links, so it is a typical multi-link system. In forward kinematics, these links are connected by joints to form a motion chain. The pose of each link can be presented directly in the homogeneous transformation matrix by link parameters and joint variables. The number of joints N_j defines the DOFs of the SMSRS. The homogeneous transformation matrix of the $\boldsymbol{i}$th link relative to the base coordinate system Σ_b is:

$$
{}^0_iT = \begin{bmatrix} n_{xi} & o_{xi} & a_{xi} & p_{xi} \\ n_{yi} & o_{yi} & a_{yi} & p_{yi} \\ n_{zi} & o_{zi} & a_{zi} & p_{zi} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \boldsymbol{n}_i & \boldsymbol{o}_i & \boldsymbol{a}_i \end{bmatrix} & \boldsymbol{p}_i \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}_i & \boldsymbol{p}_i \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{1}
$$

where, $\boldsymbol{R}_i$ represents the desired attitude matrix and $\boldsymbol{p}_i$ presents the desired position vector of the $\boldsymbol{i}$th link. The relative poses of non-adjacent links can be obtained by the successive multiplications of adjacent link homogeneous transformation matrices [7]. Equation (2) shows the pose of the $\boldsymbol{i}$th link relative to Σ_b.

$$
{}^0_iT = {}^0_1T(\theta_1){}^1_2T(\theta_2){}^2_3T(\theta_3)\ \cdots {}^{i-1}_{i}T(\theta_i) = \prod_{k=1}^{i} {}^{k-1}_{k}T(\theta_k) \tag{2}
$$

where, $\theta_i \ \in 1, 2, 3, \ldots, N_j$ represents the $\boldsymbol{i}$th joint angle. The first step to obtaining the homogeneous transformation matrix is to establish the multi-link coordinate system according to certain rules, then select the parameters to represent coordinate system relationships. Among various methods for establishing multi-link coordinate systems, the

Denavit-Hartenberg (D-H) [19] method has clear principles and can obtain the homogeneous transformation matrix by simple and fast iteration, which is used in this paper. According to its principles, each link needs to fix a coordinate system, and four parameters are selected between adjacent coordinate systems to indicate their relative relationship, which are linked length A, link twist α, link offset d, and joint angle θ. The four parameters present four-step motions from the $(i-1)$th coordinate system to the $\boldsymbol{i}$th coordinate system. In the forward kinematics of SMSRSs, any simple joint motion can be expressed by a homogeneous transformation matrix, and the complex motions can be expressed by multiplying these homogeneous transformation matrices, so the homogeneous transformation matrix of the four-step motions can be expressed in Equation (3).

$$\begin{aligned}
{}^{i-1}_{i}T &= Rot(x,\alpha_{i-1})\cdot Tran(A_{i-1},0,0)\cdot Rot(z,\theta_i)\cdot Tran(0,0,d_i)\\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & c\alpha_{i-1} & -s\alpha_{i-1} & 0 \\ 0 & s\alpha_{i-1} & c\alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} 1 & 0 & 0 & A_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} c\theta_i & -s\theta_i & 0 & 0 \\ s\theta_i & c\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\cdot\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}\\
&= \begin{bmatrix} c\theta_i & -s\theta_i & 0 & A_{i-1} \\ s\theta_i c\alpha_{i-1} & c\theta_i c\alpha_{i-1} & -s\alpha_{i-1} & -s\alpha_{i-1}d_i \\ s\theta_i s\alpha_{i-1} & c\theta_i s\alpha_{i-1} & c\alpha_{i-1} & c\alpha_{i-1}d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
\end{aligned} \tag{3}$$

On the contrary, given ${}^{0}_{N_j}T$ by Equations (2) and (3), the joint variables $\boldsymbol{q} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \cdots & \theta_{N_j} \end{bmatrix}^T$ cannot be calculated directly forward; thus they are called inverse kinematics.

3. The Objective Function for Solving Inverse Kinematics of SMSRS

3.1. Pose Error of Single Module

We use the optimization method to solve optimal $\boldsymbol{q} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \ldots & \theta_{N_j} \end{bmatrix}^T$ for the given module pose (the inverse kinematics of SMSRS). The optimization goal is to minimize the error between the desired poses and the estimated poses of modules, which is usually expressed mathematically by searching for the minimum value of the Euclidean distance between the desired poses and the estimated poses of modules [7]. For the $\boldsymbol{i}$th module, we define its estimated homogeneous transformation matrix:

$$\widehat{{}^{0}_{i}T} = \begin{bmatrix} \hat{n}_{xi} & \hat{o}_{xi} & \hat{a}_{xi} & \hat{p}_{xi} \\ \hat{n}_{yi} & \hat{o}_{yi} & \hat{a}_{yi} & \hat{p}_{yi} \\ \hat{n}_{zi} & \hat{o}_{zi} & \hat{a}_{zi} & \hat{p}_{zi} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \widehat{\boldsymbol{n}}_i & \widehat{\boldsymbol{o}}_i & \widehat{\boldsymbol{a}}_i \end{bmatrix} & \widehat{\boldsymbol{p}}_i \\ \boldsymbol{0} & \boldsymbol{1} \end{bmatrix} = \begin{bmatrix} \widehat{\boldsymbol{R}}_i & \widehat{\boldsymbol{p}}_i \\ \boldsymbol{0} & 1 \end{bmatrix} \tag{4}$$

where, $\widehat{\boldsymbol{R}}_i$ represents the estimated attitude matrix and $\widehat{\boldsymbol{p}}_i$ presents the estimated position vector of the $\boldsymbol{i}$th link. And ${}^{0}_{i}T$ in Equation (1) presents the desired homogeneous transformation matrix.

Position error: The objective function to minimize the error between $\widehat{\boldsymbol{p}}_i$ and $\boldsymbol{p}_i$ can be defined as:

$${}^{i}f_p = \frac{\|\boldsymbol{p}_i - \widehat{\boldsymbol{p}}_i\|}{\|\widehat{\boldsymbol{p}}_i\|} \tag{5}$$

Attitude error: The objective function to minimize the error between $\widehat{\boldsymbol{R}}_i$ and $\boldsymbol{R}_i$ can be defined as:

$${}^{i}f_R = \frac{\|\boldsymbol{R}_i - \widehat{\boldsymbol{R}}_i\|}{\|\widehat{\boldsymbol{R}}_i\|} \tag{6}$$

Objective function: Defining the objective function of minimum pose error of $\boldsymbol{i}$th module:

$$f_{pR}\left({}^{0}_{i}T, {}^{0}_{i}\hat{T}\right) \ = \ \varepsilon \cdot {}^{i}f_{p} + \ (1-\varepsilon) \cdot {}^{i}f_{R} \tag{7}$$

The first term on the right side of Equation (7) represents the requirements for minimum position error and the second term represents the requirements for minimum attitude error. ε is the weight factor to measure their contribution and is defined as:

$$\varepsilon = \alpha e^{-\delta} + \beta \tag{8}$$

where, $\delta = {}^{i}f_{R}$ and $\alpha = 0.7, \ \ \beta = 0.3$ were selected experimentally. If the value of ${}^{i}f_{R}$ is large ($\varepsilon \approx 0.3$), the weight factor gives priority to ${}^{i}f_{R}$. On the other hand, if the value of ${}^{i}f_{R}$ is small ($\varepsilon \approx 1$), the weight factor gives priority to ${}^{i}f_{p}$. When only the minimum position or attitude error in different tasks is required, the objective function may be only one term of Equation (7), that is:

$$f_{pR}\left({}^{0}_{i}T, {}^{0}_{i}\hat{T}\right) \ = {}^{i}f_{p} \ or \ {}^{i}f_{R} \tag{9}$$

3.2. Pose Error of Multi-Modules

When establishing the multi-link coordinate system of the SMSRS, the Σ_b is usually established at the center of mass of the attitude control module to facilitate the attitude control, and then the multi-link coordinate systems of the SMSRS is divided into a and b sides by Σ_b. When there is a requirement for the minimum pose error of multiple SMSRS modules, we define them as task modules. Consider a space task that has m task modules, n modules on the a side of Σ_b (the module where Σ_b is located is grouped into the a side) and m-n modules on the b side of Σ_b. Taking Σ_b as the counting starting point, the a_jth module is located at the $3\left(a_j-1\right)$th link coordinate system on the a side. The b_jth module is located in the $3\left(b_j-1\right)$ th link coordinate system on the b side. As shown in Figure 2, the n task modules on a side are numbered $\eta^a_1, \ \cdots, \ \eta^a_n$ link, which are displayed in dark blue. The $m-n$ task modules on the b side are numbered $\eta^b_1, \ \cdots, \ \eta^b_{m-n}$, and they are shown in light blue. The task modules are not necessarily adjacent to each other. After the task planning, since SMSRSs are free-floating and have no fixed base, the desired pose of each module is given relative to the inertial system Σ_I, which are ${}^{I}_{g}T, \ \ g = \eta^a_1, \ \cdots, \ \eta^a_n, \ \eta^b_1, \ \cdots, \ \eta^b_{m-n}$.

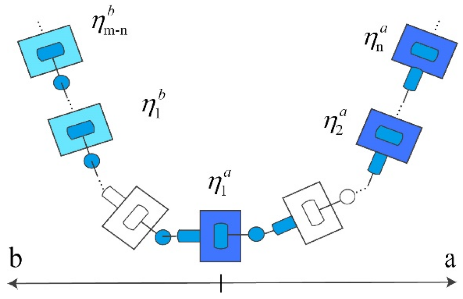


Figure 2. Schematic diagram of task module on both sides of the base coordinate system.

To establish the objective function of minimum pose error of the m task modules, the relative homogeneous transformation matrices of every two adjacent task modules should be calculated in a certain order. There are two possible relationships between the two adjacent modules: on the same side and or both sides of Σ_b.

(1) When two adjacent modules are on the same side of Σ_b: taking the $\eta^a_{g-1}, \ \eta^a_g$ link on a side as an example, their homogeneous transformation matrices relative to Σ_I are:

$$ {}^{I}_{\eta^a_{g-1}}T = {}^{I}_{0}T \left({}^{0}_{1}T\right)_a \left({}^{1}_{2}T\right)_a \cdots \left({}^{\eta^a_{g-1}-1}_{\eta^a_{g-1}}T\right)_a \tag{10}$$

$$ {}^{I}_{\eta^a_{g}}T = {}^{I}_{0}T \left({}^{0}_{1}T\right)_a \left({}^{1}_{2}T\right)_a \cdots \left({}^{\eta^a_{g}-1}_{\eta^a_{g}}T\right)_a \tag{11}$$

Then, the relative homogeneous transformation matrix between them can be calculated:

$$ {}^{\eta_{g-1}^{a}}_{\eta_{g}^{a}}T = \left({}^{I}_{\eta_{g-1}^{a}}T\right)^{-1} {}^{I}_{\eta_{g}^{a}}T = \left({}^{\eta_{g-1}^{a}}_{\eta_{g-1}^{a}+1}T\right)\left({}^{\eta_{g-1}^{a}+1}_{\eta_{g-1}^{a}+2}T\right)\cdots\left({}^{\eta_{g}^{a}-1}_{\eta_{g}^{a}}T\right) \tag{12} $$

(2) When the two adjacent links are located on different sides of Σ_b, they are η_1^a, η_1^b link of SMSRS, and their homogeneous transformation matrices relative to Σ_I are:

$$ {}^{I}_{\eta_1^a}T = {}^{I}_{0}T\left({}^{1}_{0}T\right)_a\left({}^{1}_{2}T\right)_a\cdots\left({}^{\eta_1^a-1}_{\eta_1^a}T\right)_a \tag{13} $$

$$ {}^{I}_{\eta_1^b}T = {}^{I}_{0}T\left({}^{1}_{0}T\right)_b\left({}^{1}_{2}T\right)_b\cdots\left({}^{\eta_1^b-1}_{\eta_1^b}T\right)_b \tag{14} $$

The relative homogeneous transformation matrix between them is:

$$ {}^{\eta_1^a}_{\eta_1^b}T = \left({}^{I}_{\eta_1^a}T\right)^{-1}{}^{I}_{\eta_1^b}T = \left({}^{\eta_1^a}_{\eta_1^a-1}T\right)_a\cdots\left({}^{1}_{0}T\right)_a\left({}^{0}_{1}T\right)_b\cdots\left({}^{\eta_1^b-1}_{\eta_1^b}T\right)_b \tag{15} $$

Accumulating the objective function of minimum pose error between each two adjacent task modules, the objective function of the m task modules are:

$$ f_{pose} = \sum_{i=1}^{n-1} f_{pR}\left({}^{\eta_{i+1}^a}_{\eta_i^a}T, {}^{\eta_{i+1}^a}_{\eta_i^a}\hat{T}\right) + \sum_{j=1}^{m-n-1} f_{pR}\left({}^{\eta_j^b}_{\eta_{j+1}^b}T, {}^{\eta_j^b}_{\eta_{j+1}^b}\hat{T}\right) + f_{cR}\left({}^{\eta_1^a}_{\eta_1^b}T, {}^{\eta_1^a}_{\eta_1^b}\hat{T}\right) \tag{16} $$

3.3. Self-Collision Avoidance

In an SMSRS, the modules have large volumes but are small in spacing, and their relative positions can also change in a wide range by joint motions, so they can easily collide with other modules. Therefore, it is necessary to avoid self-collision when solving the inverse kinematics of an SMSRS. In the optimization method, if there exists interference between any two modules under one solution, this solution needs to be excluded.

For an SMSRS, Ψ is defined as the set of all modules. We select two different modules $y, r \in \Psi\ (y \neq r)$ and calculate the 2-norm of the difference between their position vectors d_{yr}. When the two modules have not collided, d_{yr} needs to satisfy:

$$ d_{yr} = \|\boldsymbol{p}_y - \boldsymbol{p}_r\| > 2RC \tag{17} $$

where RC is the body diagonal of the module. As shown in Figure 3, an SMSRS with three modules is taken as an example. Draw three spheres with the centers of the modules as the origins and the body diagonal lengths as the radii. According to the judgment criteria in Equation (17), it is obvious that module 2 collided with 3, and module 1 has no contact with modules 2 and 3. From Equation (1), we find that the position vectors of the ith link related to Σ_b can be obtained by ${}^{0}_{i}T$, and the ${}^{0}_{i}T$ takes the $\boldsymbol{q} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \cdots & \theta_{N_j} \end{bmatrix}^T$ as the variable.

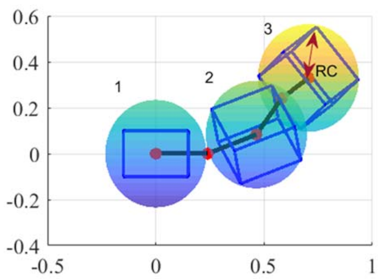


Figure 3. Schematic diagram of module collision and non-collision.

When using the optimization method to achieve self-collision avoidance, we define the objective function for self-collision avoidance of SMSRS:

$$f_{ca}(\boldsymbol{q}) = \begin{cases} 0, \ \forall y, r \in \Psi, d_{yr} > 2RC \\ \Delta, \ \ else \end{cases} \tag{18}$$

where Δ is a large value. When the module collides, the value of $f_{ca}(\boldsymbol{q})$ is large, forcing the optimization algorithm to constantly find the solution which satisfies $f_{ca}(\boldsymbol{q}) = 0$.

3.4. Objective Function

The objective function of the inverse kinematic of SMSRS can be defined as:

$$f(\boldsymbol{q}) = f_{pose}(\boldsymbol{q}) + f_{ca}(\boldsymbol{q}) \tag{19}$$

It is the combination of the objective function for minimum pose error and self-collision avoidance. In Equation (19), $f_{pose}(\boldsymbol{q})$ shown in Equation (16) represents the difference between the desired pose and the actual pose, and $f_{pose}(\boldsymbol{q})$ shown in Equation (18) is used to determine if there are collisions between modules in the SMSRS with joint angle as $\boldsymbol{q}$. The essence of the inverse kinematic issue is to find the optimal joint angles $\boldsymbol{q} = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \cdots & \theta_{N_j} \end{bmatrix}^T$ by optimization algorithms to achieve the minimum difference between the desired and actual pose of modules and guarantee that no collisions occur. When the optimization algorithm uses $f(\boldsymbol{q})$ as the fitness function, it will continuously find a more suitable $\boldsymbol{q}$ to make $f(\boldsymbol{q})$ smaller and smaller, thus achieving minimum pose error and self-collision avoidance.

3.5. How to Solve the Inverse Kinematic Problem

While using intelligent algorithms to solve inverse kinematics problems of SMSRSs, every individual contains a set of estimated joint variables $\widehat{\boldsymbol{q}}_i = \begin{bmatrix} \hat{\theta}_1 & \hat{\theta}_2 & \hat{\theta}_3 & \cdots & \hat{\theta}_{N_j} \end{bmatrix}^T$ *with* $i \in 1, 2, 3, \ldots., N$, and N is the total number of individuals, N_j is the DOFs of the SMSRS. Each set of joint variables can be brought into Equation (19) to obtain a fitness value of $f(\boldsymbol{q})$. The best solution $\widehat{\boldsymbol{q}}_{\text{best}}$ is obtained by minimizing the fitness value of $f(\boldsymbol{q})$ in each iteration, until it meets the stop criteria. We proceed to solve the inverse kinematic of the SMSRS as follows:

(1) *Initialization*. Every individual $\widehat{\boldsymbol{q}}_i$ is initialized randomly according to:

$$\widehat{\boldsymbol{q}}_i = \boldsymbol{I}_b + (\boldsymbol{u}_b - \boldsymbol{I}_b)\boldsymbol{r} \tag{20}$$

$\boldsymbol{r} = \left[rand_1, rand_1, \cdots rand_{N_j}\right]$ is a random vector with $\boldsymbol{r} \in [0, 1]$, and $\boldsymbol{I}_b, \boldsymbol{u}_b$ are lower and upper bound of $\widehat{\boldsymbol{q}}_i$, which are defined as:

$$\begin{aligned} \boldsymbol{I}_b &= \begin{bmatrix} \theta_{1_{min}} & \theta_{2_{min}} & \theta_{3_{min}} & \cdots & \theta_{N_{jmin}} \end{bmatrix}^T \\ \boldsymbol{u}_b &= \begin{bmatrix} \theta_{1_{max}} & \theta_{2_{max}} & \theta_{3_{max}} & \cdots & \theta_{N_{jmax}} \end{bmatrix}^T \end{aligned} \tag{21}$$

where $\theta_{i_{min}}, \theta_{i_{max}} i \in 1, 2, 3, \ldots, N_j$ are the minimum and maximum angle of $\boldsymbol{i}$th joint.

(2) *Fitness calculation*. For every $\widehat{\boldsymbol{q}}_j$, we compute its forward kinematics using Equations (1)–(3) to obtain ${}^I_g\hat{T}$, $g = \eta^a_1, \cdots, \eta^a_n, \ \eta^b_1, \cdots, \eta^b_{m-n}$, and then calculate the estimated relative homogeneous transformation matrices between two adjacent modules of m task modules by Equations (12) and (15). We use these estimated relative homogeneous matrices that contain estimated relative position and attitude to compute the fitness value in Equation (19).

The best solution $\widehat{\boldsymbol{q}}_{\text{best}}$ is found by solving an optimization problem:

$$\begin{aligned} &minimize : f(\widehat{\boldsymbol{q}}) \\ &subject\ to : \boldsymbol{l}_b \leq \widehat{\boldsymbol{q}}_i \leq \boldsymbol{u}_b \end{aligned} \quad (22)$$

(3) *Stop criteria.* The optimization algorithm stops when it achieves the maximum iterations, or the fitness value reaches a value of tolerance.

4. Segmented Hybrid PSO and CMA-ES Algorithm

According to the No Free Lunch (NFL) theorem [20], no optimizer is perfect and can always be the best optimizer to solve any problem [21]. Usually, an optimizer can obtain the exact optimal result of a certain kind of problem, but it may also converge to the local optimal solution or wrong solution when dealing with another kind of problem. This means that the structure and operation of the optimizer need to be adapted according to the specific problem. However, this does not mean that we always need to develop new methods to obtain higher-quality solutions for a new problem. The NFL theorem encourages us to further modify the existing optimizers to solve specific problems. In the inverse kinematic problems of SMSRSs, we envisaged to hybridize PSO with CMA-ES algorithms to obtain more accurate solutions.

The PSO algorithm has been broadly used to solve engineering optimization problems due to its simplicity, few parameters, and fast convergence speed, so we select it as one of the hybridized algorithms [22]. The concept of the PSO is based on the assumption that a particle is a potential solution to the optimization problem. The particle flies through the search space and adjusts its velocity and position following the experience of individual and surrounding particles until it finds the optimal position [23]. However, traditional PSO algorithms cannot focus on convergence speed, exploitation, and exploration performance simultaneously. Typically, the PSO algorithm has good global search performance and can quickly reduce the fitness value in the initial stage. However, it may be limited in local exploration performance and is prone to be premature. When the PSO algorithm falls into local optima, more iterations cannot reach a better solution.

Unlike PSO, the CMA-ES has good local exploration performance but poor global exploitation performance. CMA-ES is an outstanding representative of various variants of ES. CMA-ES handles the pairwise dependencies between sample points by the covariance matrix and generates new samples with a multivariate normal distribution [24]. Therefore, the CMA-ES algorithm is often used to improve the global search performance of other algorithms [21,25]. Selecting it as the other algorithm for hybridization, we propose a new Segmented Hybrid PSO and CMA-ES (SHPC) algorithm.

4.1. PSO Framework

In PSO, every potential solution of a D-dimensional problem is regarded as a particle. Each particle has several important parameters, which are current velocity, current position, and the current optimal position of particle swarm, and they are both D-dimensional vectors [26]. The position and velocity of the $\boldsymbol{i}$th particle can be expressed as:

$$x_i = (x_{i1}, x_{i2}, \ldots, x_{iD}) \quad (23)$$

$$v_i = (v_{i1}, v_{i2}, \ldots, v_{iD}) \quad (24)$$

The PSO uses the velocity updating formula and the position updating formula to guide the kth update of each particle:

$$v_i(k+1) = wv_i(k) + c_1 r_1 (P_{best}(k) - x_i(k)) + c_2 r_2 (G_{best}(k) - x_i(k)) \quad (25)$$

$$x_i(k+1) = x_i(k) + v_i(k+1) \quad (26)$$

where w is inertia weight, P_{best} is the optimal position of the $\boldsymbol{i}$th particle, G_{best} is the global optimal position of the population. c_1, c_2 are learning factors in [0.5, 2]; r_1 and r_2 are random values in (0, 1). ω is an important parameter for tuning the performance of PSO, and its widely used forms include constant value, linear and nonlinear, etc. In this paper, the PSO with linear inertia weight (LIW-PSO) is chosen as the hybridized algorithm. In LIW-PSO, the linear inertia weight can balance global exploration and local exploitation better, a larger ω is used to achieve a good global exploration in the early stage, and at the end of the algorithm, a smaller ω is used to achieve good local exploitation [27].

The right side of Equation (25) has three parts. The first part is the inertia part, which shows the influence of the velocity of the last movement of the particle. The second is the individual cognition part, which determines the particle velocity towards the individual optimal position. The third is the social recognition part, which determines the particle velocity towards the global optimal position. These three parts guide the motion of particles together. In PSO, the updates of P_{best} and G_{best} are evaluated by the fitness function and performed once in each iteration.

4.2. CMA-ES Framework

The CMA-ES is usually used to solve complex nonlinear, discontinuous convex optimization problems [21]. The three essential operations of the CMAES algorithm are:

(1) Sampling operation: firstly, the CMAES algorithm selects a solution in the solution space at random, which is used as a centroid to generate the population using a normal distribution. As shown in:

$$X_i(k+1) \sim \boldsymbol{m}(k) + \sigma(k)\boldsymbol{N}(0, \boldsymbol{C}(k)), i = 1, 2, \ldots, popsize \tag{27}$$

where X_i is the $\boldsymbol{i}$th individual in the population. popsize is the population size. $\boldsymbol{m}$ is the centroid of the population. σ is the step length. $\boldsymbol{N}$ is the multinomial normal distribution, and $\boldsymbol{C}$ is the covariance matrix of the population, which is the vital element of the CMA-ES algorithm. According to the above multivariate normal distribution, the population generated has an isodense surface that is a hyperellipsoid, and the eigenvector of $\boldsymbol{C}$ corresponds to the population distribution of the long axis of an ellipsoid. The initial $\boldsymbol{C}$ can be set to a D $\times$ D unit matrix.

(2) Selection and reorganization operation: this operation selects a part of the optimal solution as a subpopulation in the generated population. The new $\boldsymbol{m}$ is as follows:

$$\boldsymbol{m}(k+1) = \sum_{i=1}^{\mu} \rho_i * X_i(k+1), \tag{28}$$

where μ is the subpopulation size. X_i is the $\boldsymbol{i}$th individual selected among the *popsize* individuals in the population. ρ is the weight, and the sum of ρ is 1. This rule indicates that the distribution center of the next-generation population will shift to the subpopulation.

(3) Update operation: in this operation, $\boldsymbol{C}$ is updated to guide the population to search for global optimal solutions, including two operation modes: Rank$-\mu-$update and Rank$-1-$update. The Rank$-\mu-$update uses the deviation between the relative expectations of μ individuals, and the latter uses the deviation between the expectations of two adjacent generations [21].

$$\begin{aligned}\boldsymbol{C}(k+1) = & \left(1 - cr_1 - cr_\mu\right) \cdot \boldsymbol{C}(k) + cr_1 \cdot P_c(t+1) \cdot (P_c(k+1))^T \\ & + cr_\mu \cdot \sum_i^{\mu} \omega_i \cdot Y_i(k+1) \cdot (Y_i(k+1))^T\end{aligned}, \tag{29}$$

The second part of the summation on the right side of Equation (29) is the Rank$-1-$update, the third part is the Rank$-\mu-$update, and cr_1 and cr_μ are the learning rates of two update modes, $cr_1 = 2/D^2$, $cr_\mu = min\left(\mu_{eff}/D^2, 1 - cr_1\right)$, μ_{eff} is the variance-influencing choice set:

$$\mu_{eff} = \left(\sum_{i=1}^{\mu} \rho_i^2\right)^{-1}, \tag{30}$$

And, $Y_i(k+1) = (X_i(k+1) - m(k))/\sigma(k)$, c_c is the learning rate of the evolutionary path P_c. The step length σ is updated as follows:

$$\sigma(k+1) \ = \sigma(k) exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{P_\sigma(k+1)}{E\|N(0,\boldsymbol{I})\|}-1\right)\right), \tag{31}$$

where $E(\cdot)$ is the expectation function. $\boldsymbol{I}$ is the unit matrix. c_σ is the learning rate of the step, d_σ is the step update of the damping coefficient. P_σ is the evolutionary path of the step size, and its initial value is zero. After the update, it is:

$$P_\sigma(k+1) \ = \ (1-c_c)\cdot P_\sigma(k) \ + \ \sqrt{c_c\cdot\ (2-c_c)\cdot\mu_{eff}}\boldsymbol{C}(k)^{-\frac{1}{2}} \times \ \left[\frac{m(k+1)-m(k)}{\sigma(k)}\right], \tag{32}$$

Most of the parameters in the CMA-ES are independent. When the initial parameters are set, similar to PSO, the CMA-ES iterates through the above operations and gradually searches for the global optimal solution.

4.3. SHPC Algorithm

To address the need for the high-precision inverse kinematic solution of SMSRS, the SHPC algorithm hybridizes the global exploitation performance of the PSO algorithm with the local exploration performance of CMA-ES, which can give full play to their superiorities to obtain more accurate solutions. The novelty is that the hybrid approach of CMA-ES and PSO is segmented rather than structured. Structural hybrids will increase the complexity and computing time of the algorithm. The basic idea of the segmented hybrid is to divide the whole optimization process into global and local search stages. On the global stage, the PSO algorithm is used for a fast approach to the optimal solution. In the local search stage, CMA-ES and PSO algorithms are alternately used to achieve better local exploitation. To define the boundary of the two stages, we propose a fitness gradient function:

$$f_h(k) \ = \ |(f_{best}(k) \ - \ f_{best}(k-h))/f_{best}(k-h)|, \tag{33}$$

where $f_{best}(k)$ is the best fitness value obtained in the kth iteration, and h is the step width, which is a constant value set as 30 in the global search stage, $f_h(k)$ represents the relative change of the fitness value after the h iterations. In each iteration, we compare the value of $f_h(k)$ with the expected fitness gradient ξ, which is set as:

$$\xi = 0.2/log_{10}(k) \tag{34}$$

In the beginning, to give the PSO algorithm an adaptation period to fully utilize its global search capability, the SHCP algorithm runs the operation of the PSO algorithm in the first 50 iterations. After 50 iterations, we compare the values of $f_h(k)$ with ξ in each iteration; when $f_h(k) \ \geq \xi$, the PSO algorithm is considered to be still in the global search stage. Once $f_h(k) \ < \xi$, we judge that the PSO algorithm has changed from the global exploitation stage to the local exploration stage, where the CMA-ES and PSO algorithm is used to find optimal solutions one after another, the so-called segmented hybrid.

The first step in the local exploration stage is to use the best position $\hat{G}_{best}(k)$ obtained by the PSO algorithm as the $\boldsymbol{m}(k + 1)$ at the start of the CMA-ES algorithm. During the runtime of the CMA-ES algorithm, the fitness gradient function $f_h(k)$ is used again to judge whether the CMA-ES searches for the optimal solution effectively. The difference is the expected fitness gradient value ξ is set as 0.01, and the step width h is set as 10 at the local search stage. When $f_h(k) \ \geq \xi$, the current algorithm will still be used for optimization; when $f_h(k) \ < \xi$, instead of rejecting it to give the current algorithm more opportunities to show its talent, we propose a counter H with an initial value of 0. In each iteration with $f_h(k) \ < \xi$, H = H + 1; otherwise, H = 0. When the value of H exceeds 100, it indicates that after a period of iterations, the CMA-ES algorithm cannot find a better solution, so the algorithm switches to the PSO algorithm and takes the latest $\boldsymbol{m}$ obtained by the CMA-ES algorithm as the initial particles' positions. There is also a counter K in the PSO algorithm, and its initial value is 0. In each iteration with $f_h(k) \ < \xi$, K = K + 1; otherwise, K = 0. When the value of H exceeds 100, it indicates the PSO algorithm loses its effectiveness; then, the

SHCP algorithm switches to the CMA-ES algorithm again. The two counters control the segmented operation of the PSO and CMA-ES algorithms according to the changes of the $f_h(k)$. The pseudo-code of SHPC is as follows (Algorithm 1):

Algorithm 1. SHCP algorithm

```
Set $k$:= 0, H = 0, K = 101
Randomly initialize positions and velocities of all particles of PSO
WHILE (the termination conditions are not met)
    WHILE ($k$ < 50 or $f_h(k) \geq \xi$)
    Step1: Exploitation stage (Execute PSO algorithm)
      FOR (each particle $i$ in the swarm)
        Calculate fitness: Calculate the fitness value of the current particle: $f(\boldsymbol{x_i})$.
        Update $\boldsymbol{P_{best}}$: Compare the fitness value of $P_{best}$ with $f(\boldsymbol{x_i})$.
        If $f(\boldsymbol{x_i})$ is better than the fitness value of $P_{best}$, then set $P_{best}$ to the current position $\boldsymbol{x_i}$;
        Update $\boldsymbol{G_{best}}$: If $f(\boldsymbol{x_i})$ is better than the fitness value of $\boldsymbol{G_{best}}$, then $\boldsymbol{G_{best}}$ is set to the position of the current particle $\boldsymbol{x_i}$;
        Update velocities: Calculate velocities $\boldsymbol{v_i}$ using Equation (25). If $\boldsymbol{v_i} > \boldsymbol{v_{max}}$ then $\boldsymbol{v_i} = \boldsymbol{v_{max}}$.If $\boldsymbol{v_i} < \boldsymbol{v_{min}}$ then $\boldsymbol{v_i} = \boldsymbol{v_{min}}$;
        Update positions: Calculate positions $\boldsymbol{x_i}$ using Equation (26);
      END FOR
    ELSE
    Step 2: Exploitation stage (Execute CMA-ES and PSO algorithm)
      WHILE (H < 100 & K > 100) (Execute CMA-ES algorithm)
      Initialize population of CMA-ES (set $\boldsymbol{G_{best}}$ as the $\boldsymbol{m}$ at CMA-ES)
         FOR (each individual $i$)
           Update $\boldsymbol{x_i}$: Generating new individuals using the Gaussian distribution by Equation (27).
           Calculate fitness: Calculate the fitness value of the current individuals: $f(\boldsymbol{x_i})$.
           If $f(\boldsymbol{x_i})$ is better than the best fitness value, then set best fitness value as $f(\boldsymbol{x_i})$;
           Update $\boldsymbol{m}$: Updating $\boldsymbol{m}$ by the best $\mu$ individual in Equation (28)
           Update $\boldsymbol{C}$ and $\boldsymbol{\sigma}$: Covariance matrix $\boldsymbol{C}$ and Step $\boldsymbol{\sigma}$ are updated by Equations (29) and (31).
        END FOR
        If $f_h(k) \geq \xi$
           H = 0.
        ELSE
           H = H + 1.
        END IF
      ELSE (Execute PSO algorithm)
      Initialize positions of all particles as best position obtained by CMA-ES and set K = 0, H = 0 in the first iteration. Execute PSO algorithm same as Step 1
        If $f_h(k) \geq \xi$
           K = 0.
        ELSE
           K = K + 1.
        END IF
      END WHILE
    Set $k$:= $k$ + 1;
END WHILE
```

5. Experimental Settings

5.1. Settings of SMSRS

In this paper, we choose an SMSRS with nine modules as the study object. The middle module is selected to establish the Σ_b, so there are 4 modules on both the *a* and *b* side of Σ_b. For some space systems, such as space dual-arm robots, the joint axis on both sides of Σ_b are always designed to be symmetrical in direction, so the D-H parameters on both sides are equal. However, the three joints between the modules are orthogonal, which causes the asymmetrical D-H parameters on both sides of Σ_b. To solve the violation, we proposed the virtual joint coordinate system *Jv*. *Jv* is a coordinate system established between the Σ_b and the first joint coordinate system on the *b* side. Its insertion could successfully solve the problem of asymmetrical D-H parameters on both sides. *Jv* increases the DOF of the SMSRS by 1, but does not affect the mass and size of the SMSRS when its mass and size are set to 0. Then, the calculation of the pose matrix of the link on the *b* side only needs to multiply the homogeneous transformation matrix from the *Jv* to Σ_b between the first joint coordinate system and Σ_b. More details can be seen in reference [2]. Then, the DOF of the *a* side is 12, and, the DOF of the *b* side is 13 due to the existence of virtual joints. The D-H parameters of SMSRS are shown in Table 1.

Table 1. D-H parameters of SMSRS.

D-H Parameters of *a* Side					**D-H Parameters of *b* Side**				
Link	$\theta_i^a(deg)$	$d_i^a(m)$	$\alpha_{i-1}^a(deg)$	$A_{i-1}^a(m)$	**Link**	$\theta_i^b(deg)$	$d_i^b(m)$	$\alpha_{i-1}^b(deg)$	$A_{i-1}^b(m)$
1	0	0	0	0	v-0	90	0	90	0
2	90	0	90	0.198	1-v	0	−0.243	90	0
3	−90	0.243	−90	0	2	−90	0	90	0
4	−90	0	−90	0	3	90	0	−90	0.198
5	90	0	90	0.198	4	−90	−0.243	−90	0
6	−90	0.243	90	0	5	−90	0	90	0
7	−90	0	−90	0	6	90	0	−90	0.198
8	90	0	90	0.198	7	−90	−0.243	−90	0
9	−90	0.243	90	0	8	−90	0	90	0
10	−90	0	−90	0	9	90	0	−90	0.198
11	90	0	90	0.198	10	−90	−0.243	−90	0
12	−90	0.243	90	0	11	−90	0	90	0
-	-	-	-	-	12	90	0	−90	0.198

The lower and upper limits of the joint angle are set as:

$$\begin{aligned} \boldsymbol{l}_b &= \begin{bmatrix} -90 & -90 & -90 & \cdots & -90 \end{bmatrix}^T \\ \boldsymbol{u}_b &= \begin{bmatrix} 90 & 90 & 90 & \cdots & 90 \end{bmatrix}^T \end{aligned} \tag{35}$$

5.2. Cases of Inverse Kinematic Problems

In this paper, we will use the SHPC algorithm to solve three cases of inverse kinematics problems of SMSRS, in which the minimum position and attitude error of two modules, the minimum position and attitude error of three modules, the minimum attitude error of four modules are searched respectively. Self-collision avoidance between modules is considered in all three cases; the nine module SMSRSs with task modules marked in red are shown in Figure 4.

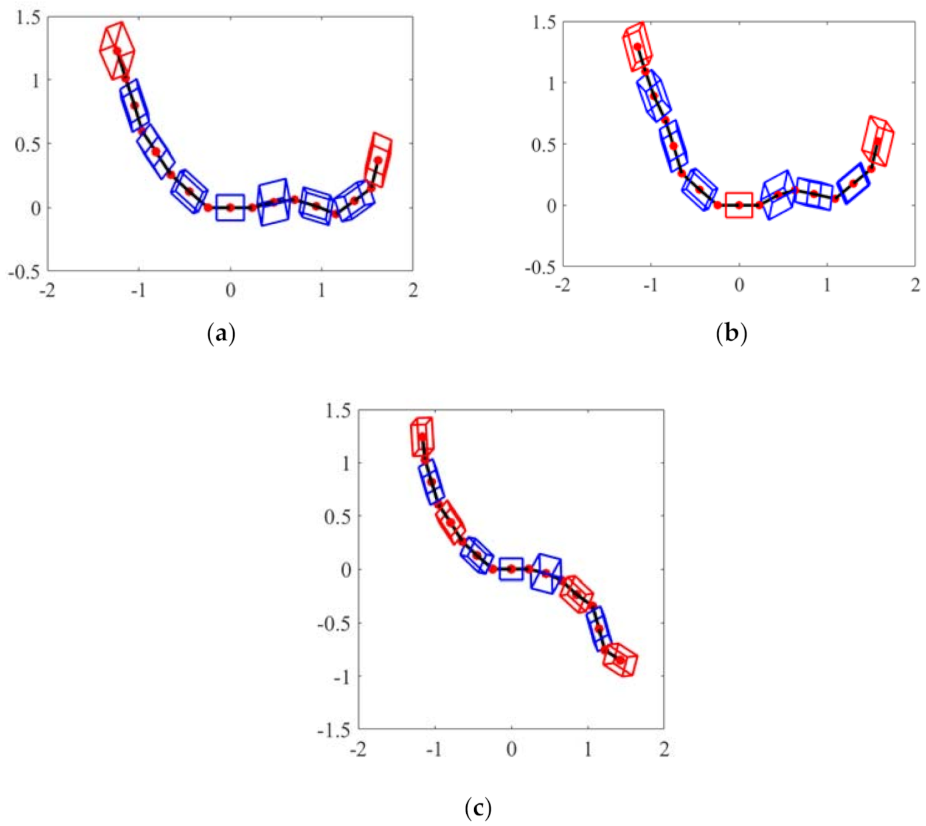


Figure 4. Nine module SMSRS and the task modules of three cases: (**a**) case 1, (**b**) case 2, and (**c**) case 3.

5.2.1. Case 1: Minimum Position and Attitude Error of Two Modules

As shown in Figure 4a, we select the end modules of *a* side and *b* side as the task modules, that is m = 1, n = 1, η_1^a= 12, η_1^b= 13 in Section 3.2. The pose matrix of η_2^a, η_1^b relative to Σ_I is:

$$ {}^{I}_{\eta_1^a}T = {}^{I}_{0}T\left({}^{0}_{1}T\right)_a\left({}^{1}_{2}T\right)_a\cdots\left({}^{11}_{12}T\right)_a \tag{36}$$

$$ {}^{I}_{\eta_1^b}T = {}^{I}_{0}T\left({}^{0}_{1}T\right)_b\left({}^{1}_{2}T\right)_b\cdots\left({}^{12}_{13}T\right)_b \tag{37}$$

Calculating their relative pose matrix:

$$ {}^{\eta_1^a}_{\eta_1^b}T = \left({}^{I}_{\eta_1^a}T\right)^{-1}{}^{I}_{\eta_1^b}T = \left({}^{12}_{11}T\right)_a\left({}^{11}_{10}T\right)_a\cdots\left({}^{1}_{0}T\right)_a\left({}^{0}_{1}T\right)_b\left({}^{1}_{2}T\right)_b\cdots\left({}^{12}_{13}T\right)_b \tag{38}$$

At this time, the objective function is:

$$ f_1(\boldsymbol{q}) \ = f_{pR}\left({}^{\eta_1^a}_{\eta_1^b}T, {}^{\eta_1^a}_{\eta_1^b}\hat{T}\right) \ + \ f_{ca}(\boldsymbol{q})\Big) \tag{39}$$

The desired homogeneous transformation matrices of the two modules are set as:

$$ {}^{I}_{\eta_1^a}\hat{T} = \begin{bmatrix} -0.791 & -0.320 & -0.522 & 0.447 \\ -0.314 & 0.519 & 0.794 & 1.150 \\ -0.52 & 0.792 & 0.310 & 0.644 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{40}$$

$$
{}^{I}_{\eta_1^b}\hat{T} = \begin{bmatrix} 0.754 & -0.133 & 0.642 & -1.060 \\ 0.173 & 0.984 & 0.000 & -0.828 \\ -0.633 & 0.112 & 0.766 & 0.685 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{41}
$$

5.2.2. Case 2: Minimum Position and Attitude Error of Three Modules

As shown in Figure 4b, we select the end module of a side, base module, and end module of b side as the task modules, that is $m = 2$, $n = 1$, $\eta_1^a = 0$, $\eta_2^a = 12$, $\eta_1^b = 13$ in Section 3.2. The pose matrix of η_2^a, η_1^b in Σ_I is the same as Equations (36) and (37). The pose matrix of the base module in Σ_I is ${}^{I}_{0}T$, which is given by sensors. The relative homogeneous transformation matrix ${}^{\eta_2^a}_{\eta_1^a}T$, ${}^{\eta_1^a}_{\eta_1^b}T$ are calculated, respectively, as:

$$
{}^{\eta_2^a}_{\eta_1^a}T = \left({}^{12}_{11}T\right)_a \left({}^{11}_{10}T\right)_a \cdots \left({}^{1}_{0}T\right)_a \tag{42}
$$

$$
{}^{\eta_1^a}_{\eta_1^b}T = \left({}^{0}_{1}T\right)_b \left({}^{1}_{2}T\right)_b \cdots \left({}^{12}_{13}T\right)_b \tag{43}
$$

At this time, the objective function is:

$$
f_2(\boldsymbol{q}) = f_{pR}\left({}^{\eta_1^a}_{\eta_1^b}T, {}^{\eta_1^a}_{\eta_1^b}\hat{T}\right) + f_{pR}\left({}^{\eta_2^a}_{\eta_1^a}T, {}^{\eta_2^a}_{\eta_1^a}\hat{T}\right) + f_{ca}(\boldsymbol{q}) \tag{44}
$$

The desired homogeneous transformation matrices of the three modules are set as:

$$
{}^{I}_{\eta_1^a}\hat{T} = \begin{bmatrix} 1.000 & 0.000 & 0.000 & 0.000 \\ 0.000 & 1.000 & 0.000 & 0.000 \\ 0.000 & 0.000 & 1.000 & 0.000 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{45}
$$

$$
{}^{I}_{\eta_2^a}\hat{T} = \begin{bmatrix} -0.242 & -0.775 & 0.583 & 1.667 \\ 0.198 & -0.627 & -0.752 & 0.116 \\ 0.949 & -0.066 & 0.306 & 0.270 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{46}
$$

$$
{}^{I}_{\eta_1^b}\hat{T} = \begin{bmatrix} 0.052 & 0.366 & 0.928 & -1.153 \\ -0.745 & 0.628 & -0.206 & -0.099 \\ -0.659 & -0.685 & 0.307 & 1.293 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{47}
$$

5.2.3. Case 3: Minimum Attitude Error of Four Modules

As shown in Figure 4c, we select the third and fifth modules of a side, second and fourth modules of b side as the task modules, that is $m = 2$, $n = 2$, $\eta_1^a = 6$, $\eta_2^a = 12$, $\eta_1^b = 7$, $\eta_2^b = 13$ in Section 3.2. The pose matrix of η_2^a, η_2^b in Σ_I is the same as case 1. The pose matrix of η_1^a and η_1^b in Σ_I is:

$$
{}^{I}_{\eta_1^a}T = {}^{I}_{0}T\left({}^{0}_{1}T\right)_a \left({}^{1}_{2}T\right)_a \cdots \left({}^{5}_{6}T\right)_a \tag{48}
$$

$$
{}^{I}_{\eta_1^b}T = {}^{I}_{0}T\left({}^{0}_{1}T\right)_b \left({}^{1}_{2}T\right)_b \cdots \left({}^{6}_{7}T\right)_b \tag{49}
$$

Calculating the relative matrix between them in turn:

$$ {}^{\eta_2^a}_{\eta_1^a}T = \left({}^{12}_{11}T\right)_a \left({}^{11}_{10}T\right)_a \cdots \left({}^{7}_{6}T\right)_a \tag{50} $$

$$ {}^{\eta_1^a}_{\eta_1^b}T = \left({}^{6}_{5}T\right)_a \left({}^{5}_{4}T\right)_a \cdots \left({}^{1}_{0}T\right)_a \left({}^{0}_{1}T\right)_b \left({}^{1}_{2}T\right)_b \cdots \left({}^{6}_{7}T\right)_b \tag{51} $$

$$ {}^{\eta_1^b}_{\eta_2^b}T = \left({}^{7}_{8}T\right)_b \left({}^{8}_{9}T\right)_b \cdots \left({}^{12}_{13}T\right)_b \tag{52} $$

In case 3, the objective function is:

$$ f_3(\boldsymbol{q}) = f_{pR}\left({}^{\eta_1^a}_{\eta_1^b}T, {}^{\eta_1^a}_{\eta_1^b}\hat{T}\right) + f_{pR}\left({}^{\eta_2^a}_{\eta_1^a}T, {}^{\eta_2^a}_{\eta_1^a}\hat{T}\right) + f_{pR}\left({}^{\eta_1^b}_{\eta_2^b}T, {}^{\eta_2^a}_{\eta_1^a}\hat{T}\right) + f_{ca}(\boldsymbol{q}) \tag{53} $$

The desired homogeneous transformation matrices of the four modules are set as:

$$ {}^{I}_{\eta_1^a}\hat{T} = \begin{bmatrix} -0.139 & -0.587 & 0.796 & 0.895 \\ 0.981 & -0.190 & 0.031 & 0.058 \\ 0.133 & 0.786 & 0.603 & 0.269 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{54} $$

$$ {}^{I}_{\eta_2^a}\hat{T} = \begin{bmatrix} -0.289 & -0.719 & 0.630 & 1.447 \\ 0.420 & -0.687 & -0.591 & 0.474 \\ 0.859 & 0.093 & 0.501 & 0.815 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{55} $$

$$ {}^{I}_{\eta_1^b}\hat{T} = \begin{bmatrix} -0.151 & 0.699 & 0.698 & -0.887 \\ -0.962 & -0.266 & 0.058 & -0.026 \\ 0.227 & -0.662 & 0.713 & 0.278 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{56} $$

$$ {}^{I}_{\eta_2^b}\hat{T} = \begin{bmatrix} 0.191 & 0.767 & 0.612 & -1.638 \\ -0.959 & 0.015 & 0.280 & 0.220 \\ 0.205 & -0.641 & 0.739 & 0.839 \\ 0.000 & 0.000 & 0.000 & 1.000 \end{bmatrix} \tag{57} $$

5.3. Settings of Compared Algorithms

To test the performance of the SHPC algorithm in solving inverse kinematic problems of SMSRSs, we selected several famous algorithms to compare it with. These algorithms include several PSO variants, several meta-heuristic algorithms, and several hybrid PSO algorithms. The PSO variants are standard PSO with constant inertia weight (SPSO), the standard PSO with linear inertia weight (LIW-PSO) [28], the standard PSO with nonlinear inertia weight (NLIW-PSO) [29], the PSO with constriction factor (CPSO) [30], the iterative PSO (IPSO) [31], and the PSO with the passive congregation (PSOPC) [32]. The selected meta-heuristic algorithms are DE [33], GWO [34], BA [35], BOA [36], and ABC [37] algorithms. The hybrid algorithms are HPSOBOA [36], PSOBOA [36], and PSOGWO [22]. The parameter settings of all selected algorithms are shown in Table 2, which are referenced in original literature or selected after extensive calculations.

Table 2. Parameter settings of compared algorithms.

Algorithm	Parameter Setting
SPSO	$c_1 = c_2 = 2, w = 0.5$
LIW-PSO [28]	$c_1 = c_2 = 2,\ \omega_{max} = 0.9,\ \omega_{min} = 0.4$
NLIW-PSO [29]	$c_1 = c_2 = 2, w = 0.3, \mu = 1.00002$
CPSO [30]	$c_1 = c_2 = 2.05, \chi = 0.73$
IPSO [31]	$c_1 = c_2 = 0.01, \lambda = c_1\left(1 - e^{-c_1 k}\right)$
PSOPC [32]	$c_1 = c_2 = 0.5, c_3 = [0.4,\ 0.6]$
SHCP	$c_1 = c_2 = 2, \omega_{max} = 0.9, \omega_{min} = 0.4$
DE [33]	$CR = 0.9$
GWO [34]	$\boldsymbol{a}$ decreased linearly from 2 to 0
BA [35]	$f_{\max} = 2, f_{\min} = 0, A^0 \in [1,2], r^0 \in [0,0.2], \alpha = 0.9, \gamma = 0.9$
ABC [37]	limit = 100
BOA [36]	$a = 0.1, c(0) = 0.01, p = 0.6$
PSOBOA [36]	$a = 0.1, c(0) = 0.01, p = 0.6,\ c_1 = c_2 = 0.5$
HPSOBOA [36]	$a_{first} = 0.1, a_{final} = 0.3, c(0) = 0.01, p = 0.6, x(0) = 0.315, \rho = 0.295,\ c_1 = c_2 = 0.5$
PSOGWO [22]	ω= 0.7298, $c_1 = c_2$= 1.49445, $\boldsymbol{a}$ decreased linearly from 2 to 0

The terminal condition is the maximum iteration and is set to 1000. All algorithms were implemented in MATLAB 2012a and run on the same machine with an Intel(R) Core(TM) i5-8300H CPU @ 2.30 GHz (Intel, Santa Clara, CA, USA) and 8 G memory.

6. Experiments and Comparison Results

6.1. Comparison Results in Solution Accuracy

The algorithms in Table 2 iteratively search for the minimum values of objective functions for the three cases. To perform a fair comparison of these algorithms, each algorithm runs 10 times in each case, and five statistical indicators: the average value, the best value, the worst value, median, and standard deviation are recorded for the 10 times optimizations. The results between SHPC and PSO variants, meta-heuristic algorithms, and hybrid algorithms are recorded in Tables 3–5.

Table 3. Comparisons of experimental results between SHPC and PSO variants.

FUN		SHPC	SPSO	LIW-PSO	NLIW-PSO	CPSO	IPSO	PSOPC
f_1	Mean	$8.29\ 10^{-3}$	5.00×10^3	3.00×10^3	1.74×10^{-1}	6.00×10^4	7.16×10^{-1}	1.31
	Best	7.21×10^{-12}	6.52×10^{-12}	9.40×10^{-7}	6.25×10^{-3}	4.00×10^4	5.06×10^{-1}	1.31
	Worst	4.12×10^{-2}	3.00×10^4	3.00×10^4	5.77×10^{-1}	9.00×10^4	1.15	1.31
	Std	1.66×10^{-2}	1.08×10^4	9.49×10^3	1.82×10^{-1}	1.41×10^4	2.10×10^{-1}	2.34×10^{-16}
	median	5.20×10^{-6}	1.88×10^{-3}	3.24×10^{-4}	9.11×10^{-2}	6.00×10^4	7.06×10^{-1}	1.31
f_2	Mean	3.92×10^{-2}	8.00×10^3	3.45×10^{-1}	4.57×10^{-1}	6.70×10^4	2.00×10^3	1.62
	Best	1.48×10^{-2}	8.39×10^{-2}	8.13×10^{-2}	2.41×10^{-1}	5.00×10^4	6.40×10^{-1}	1.62
	Worst	8.47×10^{-2}	3.00×10^4	9.02×10^{-1}	6.69×10^{-1}	9.00×10^4	2.00×10^4	1.62
	Std	1.90×10^{-2}	1.32×10^4	2.75×10^{-1}	1.55×10^{-1}	1.57×10^4	6.32×10^3	0.00
	median	3.97×10^{-2}	2.36×10^{-1}	2.62×10^{-1}	4.13×10^{-1}	6.50×10^4	1.42	1.62
f_3	Mean	1.21×10^{-2}	1.26×10^{-2}	1.76×10^{-1}	1.70×10^{-2}	4.80×10^4	1.45	2.22
	Best	9.97×10^{-16}	9.35×10^{-16}	1.10×10^{-15}	1.02×10^{-8}	2.00×10^4	6.09×10^{-1}	2.22
	Worst	6.99×10^{-2}	1.25×10^{-1}	1.08	6.24×10^{-2}	7.00×10^4	2.61	2.22
	Std	2.58×10^{-2}	3.96×10^{-2}	3.58×10^{-1}	2.18×10^{-2}	1.69×10^4	5.73×10^{-1}	0.00

Table 4. Comparisons of experimental results between SHPC and meta-heuristic algorithms.

FUN		MY-PSO	DE	GWO	BA	ABC	CMA-ES	BOA
f_1	Mean	8.29×10^{-3}	4.38×10^{-2}	1.31	9.50×10^{4}	3.90×10^{-1}	5.63×10^{-2}	1.32×10^{-1}
	Best	7.21×10^{-12}	7.53×10^{-3}	1.31	5.00×10^{4}	2.30×10^{-1}	1.62×10^{-2}	4.37×10^{-2}
	Worst	4.12×10^{-2}	8.67×10^{-2}	1.31	1.40×10+05	5.09×10^{-1}	1.63	2.33×10^{-1}
	Std	1.66×10^{-2}	2.57×10^{-2}	2.34×10^{-16}	3.81×10^{4}	8.44×10^{-2}	4.60×10^{-2}	6.20×10^{-2}
	median	5.20×10^{-6}	4.85×10^{-2}	1.31	9.00×10^{4}	4.00×10^{-1}	3.89×10^{-2}	1.32×10^{-1}
f_2	Mean	3.92×10^{-2}	3.23×10^{-2}	1.62	9.20×10^{4}	9.93×10^{-1}	5.82×10^{-2}	6.16×10^{-1}
	Best	1.48×10^{-2}	1.45×10^{-2}	1.62	9.00×10^{4}	7.96×10^{-1}	3.06×10^{-2}	3.68×10^{-1}
	Worst	8.47×10^{-2}	7.69×10^{-2}	1.62	1.10×10^{5}	1.15	1.06	8.09×10^{-1}
	Std	1.90×10^{-2}	1.78×10^{-2}	0.00	6.32×10^{3}	1.25×10^{-1}	2.06×10^{-2}	1.40×10^{-1}
	median	3.97×10^{-2}	2.88×10^{-2}	1.62	9.00×10^{4}	1.03	5.63×10^{-2}	6.26×10^{-1}
f_3	Mean	1.21×10^{-2}	3.00×10^{-2}	2.22	9.50×10^{4}	1.79	7.02×10^{-2}	9.99×10^{-1}
	Best	9.97×10^{-16}	1.21×10^{-2}	2.22	5.00×10^{4}	1.57	4.27×10^{-2}	3.56×10^{-1}
	Worst	6.99×10^{-2}	4.51×10^{-2}	2.22	1.40×10+05	2.03	1.04×10^{-1}	1.58
	Std	2.58×10^{-2}	1.08×10^{-2}	0.00	2.92×10^{4}	1.75×10^{-1}	2.06×10^{-2}	4.05×10^{-1}
	median	1.21×10^{-15}	3.05×10^{-2}	2.22	9.00×10^{4}	1.78	6.63×10^{-2}	9.50×10^{-1}

Table 5. Comparisons of experimental results between SHPC and hybrid algorithms.

FUN		MY-PSO	PSOBOA	HPSOBOA	PSOGWO
f_1	Mean	8.29×10^{-3}	7.49×10^{-1}	4.88×10^{-1}	2.00×10^{3}
	Best	7.21×10^{-12}	5.17×10^{-1}	4.64×10^{-1}	1.19×10^{-4}
	Worst	4.12×10^{-2}	8.84×10^{-1}	5.48×10^{-1}	2.00×10^{4}
	Std	1.66×10^{-2}	1.23×10^{-1}	2.70×10^{-2}	6.32×10^{3}
	median	5.20×10^{-6}	7.91×10^{-1}	4.74×10^{-1}	5.11×10^{-3}
f_2	Mean	3.92×10^{-2}	1.08	9.70×10^{-1}	2.37×10^{-1}
	Best	1.48×10^{-2}	9.81×10^{-1}	9.46×10^{-1}	2.12×10^{-2}
	Worst	8.47×10^{-2}	1.34	9.93×10^{-1}	1.39
	Std	1.90×10^{-2}	1.03×10^{-1}	1.26×10^{-2}	4.23×10^{-1}
	median	3.97×10^{-2}	1.07	9.71×10^{-1}	6.54×10^{-2}
f_3	Mean	1.21×10^{-2}	1.55	1.85	1.74×10^{-1}
	Best	9.97×10^{-16}	1.12	1.72	3.05×10^{-4}
	Worst	6.99×10^{-2}	1.89	1.91	8.86×10^{-1}
	Std	2.58×10^{-2}	2.54×10^{-1}	6.48×10^{-2}	3.51×10^{-1}
	median	1.21×10^{-15}	1.59	1.87	6.73×10^{-3}

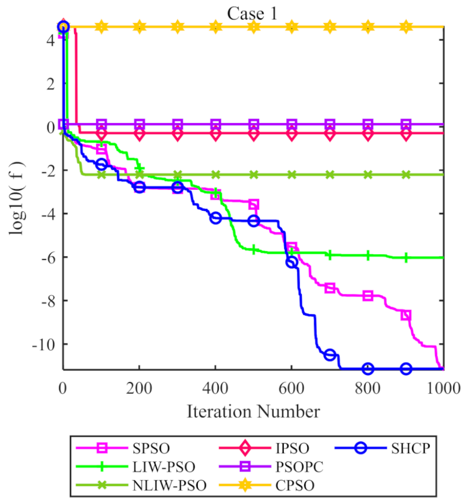
From the comparison results between SHCP and other PSO variants in Table 3, it can be seen that the SHPC algorithm has outstanding performance in solving the inverse kinematics problems of all three cases. Although some statistical indicators of SHCP are not optimal, such as the best value in case 1 is not as good as that of SPSO, the difference between them is small, while all other statistical indicators of SPSO are worse than the SHCP algorithm. In general, all statistical indicators of SHCP algorithms exceed any PSO variant. In particular, the result of SHCP outperforms the LIW-PSO algorithm, which proves the introduction of CMA-ES into LIW-PSO can greatly improve the algorithm performance. It is worth noting that some statistical indicators in Table 3 exceed 10,000, such as the mean values of SPSO, NLIW-PSO, and CPSO in case 1, the mean values of CPSO and SPSO in case 2, which reflect that these algorithms cannot find suitable solutions to avoid self-collision. Fortunately, the SHPC algorithm can find solutions to avoid self-collision perfectly in every solution and achieves high solution accuracy in all three cases, which fully satisfies the requirements of space tasks. It can be concluded that the SHCP algorithm is a powerful and effective method for finding the high-precision solutions to the inverse kinematics problem of SMSRS.

From the comparison results between SHCP and other meta-heuristic algorithms in Table 4, we find that the SHCP algorithm shows great advantages in general. In case 1, all the statistical indicators outperform other meta-heuristic algorithms. In case 2, all the statistical indicators of SHCP algorithms outperform all meta-heuristic algorithms except the DE algorithm. Although the results are not better than the DE algorithm, they are very close. In case 3, all the statistical indicators of SHCP algorithms outperform all algorithms except for the worst value and standard deviation which are worse than the DE algorithm, but the results of the SHCP algorithm are still better than the result of the DE algorithm in general. There is no doubt that the DE algorithm is excellent. In particular, the result of the SHCP algorithm is better than that of the CMA-ES algorithm, which proves the hybrid with LIW-PSO can greatly improve the performance of the CMA-ES algorithm. All statistical indicators of BA algorithms exceed 10,000, which proves that it cannot find a solution to avoid self-collision.

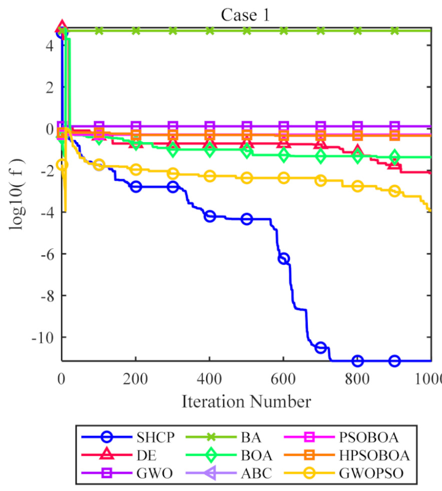
From the results in Table 5, it can be seen that the SHPC algorithm has comprehensive advantages over other hybrid algorithms. All the statistical indicators of the SHCP algorithm exceed other hybrid algorithms, indicating the segmented hybrid mechanism of SHPC is an effective operation. In addition, the performance of the hybrid BOA algorithm is inferior to the non-hybrid algorithm in some cases, but the performance of the PSOGWO algorithm outperforms the GWO algorithm, indicating that the hybrid algorithm may or may not achieve better solution results, and needs to be adjusted to local conditions.

6.2. Result Comparisons on Convergence Curves

To analyze the algorithm performance from the convergence process, for each algorithm, the optimal convergence curves among 10 runs in the three cases are drawn in Figure 5. The fitness value is calculated logarithmically based on ten. We can find the curve of the SHCP in each case is basically near the bottom of the figure, which indicates the SHCP algorithm can always maintain its advantages throughout the iterations. The algorithms with almost horizontal fitness curves, such as the CPSO, PSOPC, IPSO, BOA, BA, and PSOBOA, show signs of prematureness. The curve of the SHCP algorithm maintains a downward trend overall throughout the iteration, which indicates that it can effectively avoid prematureness.

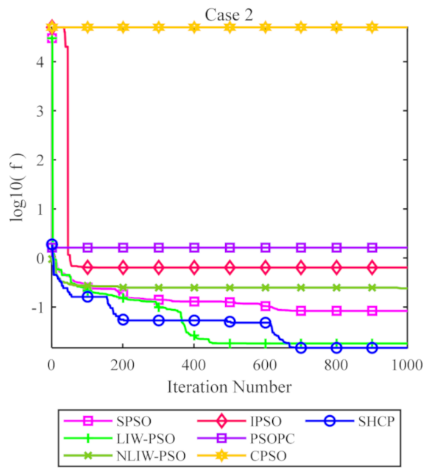


(**a**)

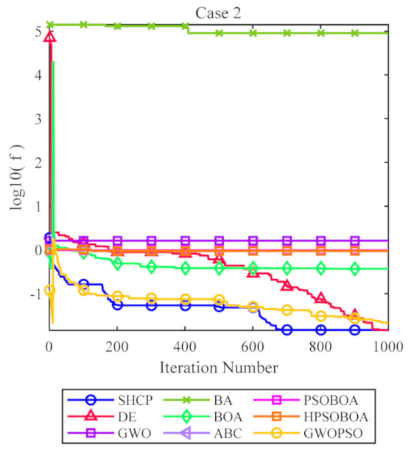


(**b**)

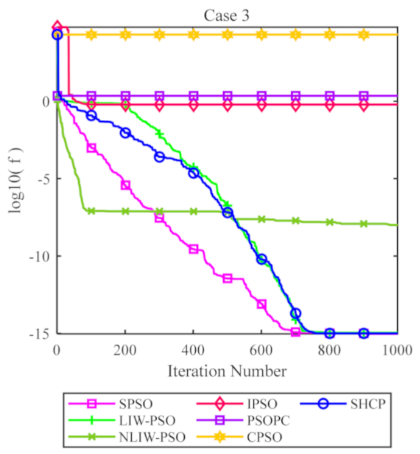
Figure 5. *Cont.*



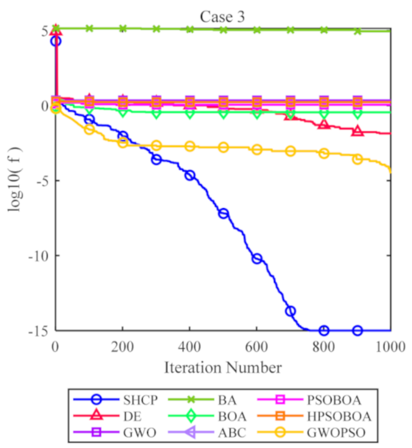
(**c**)



(**d**)



(**e**)



(**f**)

Figure 5. Convergence curves of algorithms. (**a**) Convergence curves of SHCP and PSO variants for Case 1; (**b**) Convergence curves of SHCP and meta-heuristic algorithms for Case 1; (**c**) Convergence curves of SHCP and PSO variants for Case 2; (**d**) Convergence curves of SHCP and meta-heuristic algorithms for Case 2; (**e**) Convergence curves of SHCP and PSO variants for Case 3; (**f**) Convergence curves of SHCP and meta-heuristic algorithms for Case 3.

6.3. Self-Collision Avoidance

As shown in Figure 6a–c, the configurations of the SMSRS of case 1-3 respectively under the final solutions obtained by the SHCP algorithm are drawn and used to judge whether there are collisions between the modules. It can be seen that there is no contact between all modules, which meets the requirements for no self-collision in all three cases. It demonstrates the correctness of the proposed self-collision avoidance strategy and the effectiveness of the optimization method in achieving self-collision avoidance of the SMSRS.

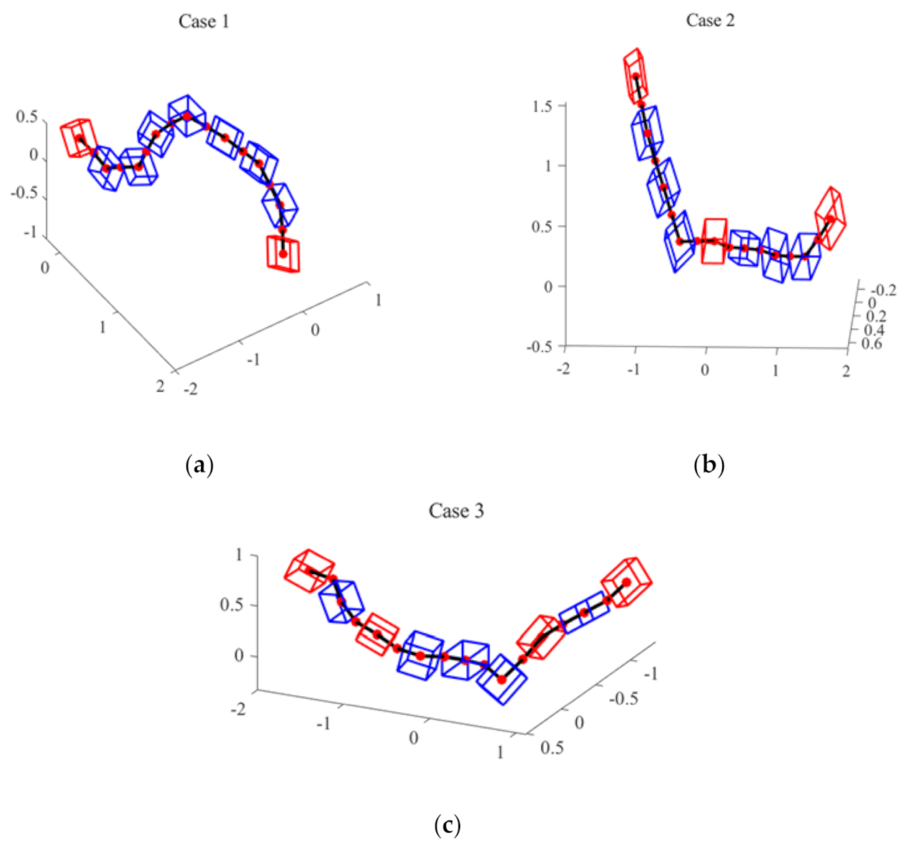


Figure 6. Configurations of the SMSRS under the solutions of the SHCP algorithm. (**a**) Case 1; (**b**) Case 2; (**c**) Case 3.

6.4. Feasibility Analysis of the Optimization Method

To analyze the feasibility of the optimization method to solve the inverse kinematics, configurations of SMSRS obtained by different algorithms for one case are drawn in one figure; these algorithms are selected due to they have higher accuracy solutions among all compared algorithms. The results can be seen in Figure 7a–c.

In Figure 7a, the solution results of CMA-ES, SHCP, GWOPSO, and LIW-PSO algorithms are drawn in case 1. Although the entire configurations of SMSRS obtained by these algorithms are different, the relative position and attitude of their two task modules are basically the same, which proves that they are all trying to find the pose of the task modules to meet the requirements for case 1. The difference in solution accuracy causes the two end modules to not completely overlap, but the basic principle is that the higher the accuracy of the algorithm, the more it can meet the mission requirements.

Case 2 has a request for the relative pose of the three task modules. From the configurations obtained by the CMA-ES, SHCP, LIW-PSO, and DE algorithms in Figure 7b, we find the poses of the three task modules are basically the same, and the difference in the solution accuracy cause their slight pose differences.

In Figure 7c, the solution results of the SHCP, CMA-ES, GWOPSO, and LIW-PSO algorithms are selected in case 3 for drawing. Case 3 requires solutions to meet the relative attitude requirements of the four task modules, although the positions of the four task modules are different, their relative attitudes are basically the same. All the optimization algorithms are trying to find the solutions to meet the set attitude requirements.

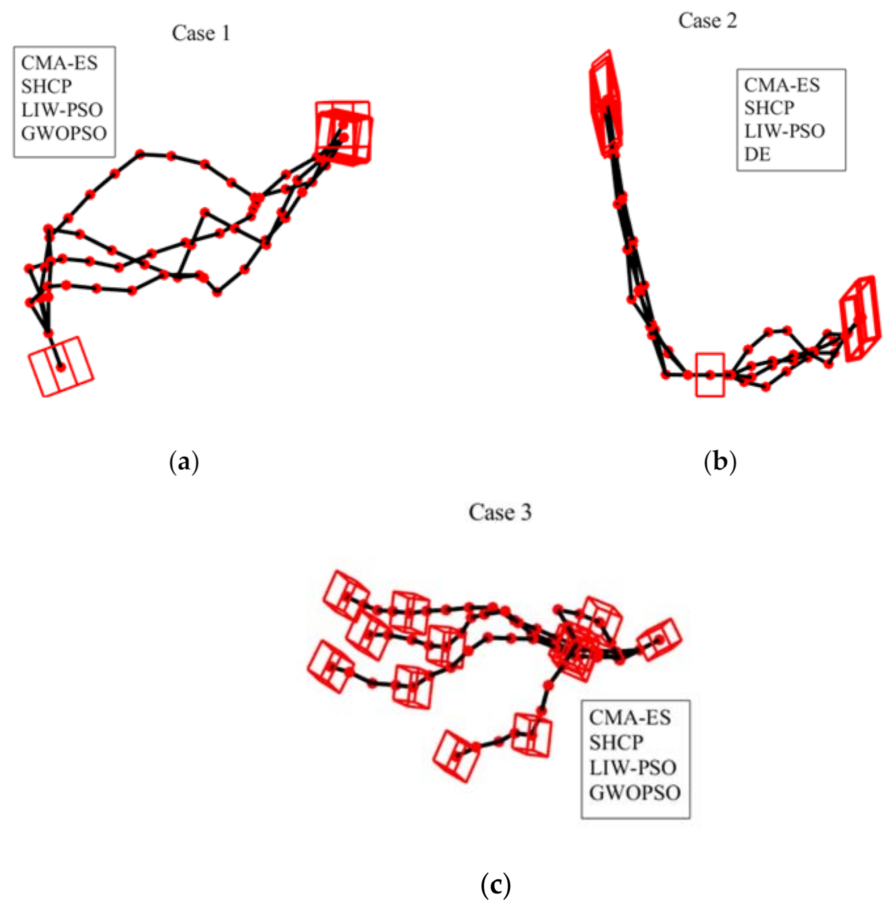


Figure 7. The configurations obtained by different algorithms of SMSRS for three cases. (**a**) Case 1; (**b**) Case 2; (**c**) Case 3.

The analysis result in this section proves the correctness of the kinematics model established in this paper and the feasibility of using the optimization algorithm to solve the inverse kinematics problem of SMSRSs.

6.5. Effect Analysis of Segmented Hybrid

To analyze the mechanism of the segmented hybrid, we draw the fitness curves of the SHCP algorithm and the segments where they are located. We then compare the fitness curves of the SHCP algorithm with those of the CMA-ES and LIW-PSO algorithm, which are the original algorithms of the SHCP algorithm. As shown in Figure 8, the segments of the fitness curve of the SHCP algorithm marked by the arrow run the CMA-ES algorithm, and the other segments run the PSO algorithm.

(1) Case 1: the first segment in Figure 8a is the global search stage of the SHCP algorithm. At this stage, the curves of SHCP and LIW-PSO algorithms keep decreasing, but CMA-ES has stagnated, verifying that the LIW-PSO algorithm does have advantages in global search. After the continuous decline, the curve of the LIW-PSO algorithm in the fourth segment has gradually stabilized. At this segment, the SHCP algorithm runs the CMA-ES algorithm and its fitness value still fluctuates and decreases, breaking the stagnant trend of the original algorithm.

(2) Case 2: in the first segment, the three curves decrease rapidly. After the SHCP algorithm enters the CMA-ES segment, the fitness value decreases rapidly after a period of the adaptation period, then trends to stagnation like the CMA-ES algorithm, while the curve of the LIW-PSO algorithm continues to decline. Whereas the LIW-PSO algorithm also tends to converge, the SHCP algorithm breaks through the bottleneck in the fifth segment and decreases rapidly until it exceeds the LIW-PSO algorithm. Case 2

indicates that when the convergence speed in the global search stage is unsatisfactory, the SHCP algorithm switches to the CMA-ES algorithm immediately to accelerate the fitness curve decline. It also demonstrates that the SHCP algorithm is constantly activated in the switching between LIW-PSO and CMA-ES algorithms, thus increasing the vitality of the algorithm.

(3) Case 3: in this case, the LIW-PSO algorithm has shown excellent performance from the beginning. Therefore, the SHCP algorithm is always in the LIW-PSO segment until it obtains high-precision solutions, rendering CMA-ES useless. This phenomenon reflects that the SHCP algorithm could make full use of the excellent performance of the LIW-PSO algorithm, where LIW-PSO is applicable. It is also proven that the segmentation strategy can adaptively adjust the boundary of the global exploration stage and local exploitation stage.

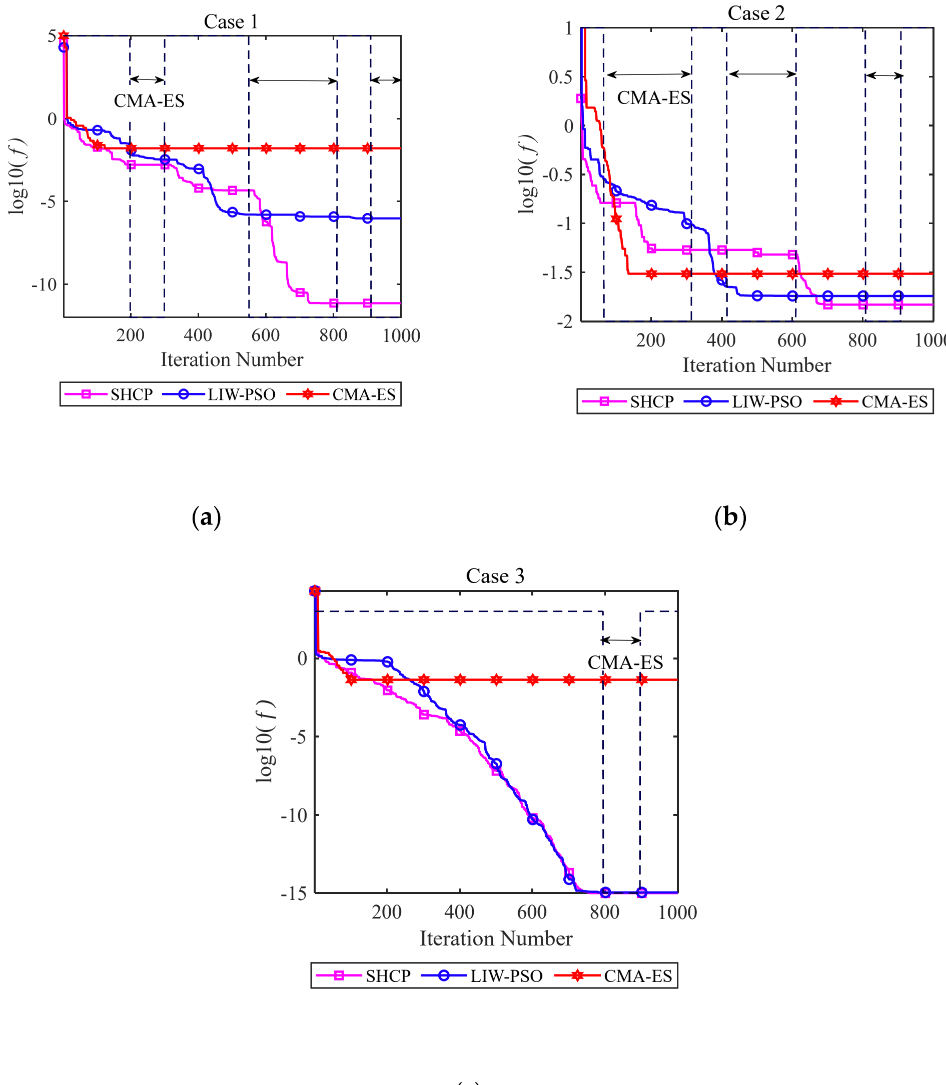


Figure 8. Segments and fitness curves of SHCP algorithm compared with original algorithms. (**a**) Case 1; (**b**) Case 2; (**c**) Case 3.

By analyzing three cases, it is found that the SHCP algorithm can automatically adjust its operation according to the characteristics of the problem, realizing the adaptability to different task requirements and achieving solutions with higher accuracy. Therefore, the SHCP algorithm is proven to be an effective algorithm to solve the inverse kinematics problem of SMSRSs.

7. Conclusions

In this paper, we investigate how to use the optimization method to solve the inverse kinematics of hyper-redundant SMSRSs and propose the SHCP algorithm, which makes use of the global search advantages of PSO optimization and the local search advantages of CMA-ES algorithms by adopting the segmented hybrid concept to improve the algorithm performance. The SHCP algorithm is applied to solve inverse kinematics problems of SMSRSs. Compared with some PSO variants, hybrid algorithms, and meta-heuristic algorithms, SHCP has great advantages in general and can find higher quality solutions for desired poses of task modules, which proves the SHCP algorithm is adaptable for the inverse kinematics of SMSRSs. By analyzing the trends of fitness curves, it is found that the segmented hybrid mechanism of the SHCP can adaptively adjust the operation of the algorithm according to changes in the fitness value to obtain high accuracy solutions for completely different cases, which proves it to be a successful mechanism.

In addition, the self-collision avoidance considered in the inverse kinematics problem can be achieved by the optimization method. The SHCP algorithm is also an effective algorithm for finding solutions to avoid self-collision.

In this paper, the inverse kinematics of SMSRSs are studied for the first time; the study results prove that the SHCP algorithm is a recommended algorithm for solving the inverse kinematics of SMSRSs. The research methods and innovative algorithms have significance for future research. Improving the running speed of the SHCP algorithm will promote its onboard real-time operation, and future work will focus on optimizing the structure and improving the speed of the SHCP algorithm.

Author Contributions: Conceptualization, J.A. and X.L.; data curation, J.A.; formal analysis, Z.Z. and J.A.; funding acquisition, X.L. and D.Y.; investigation, J.A., W.M., J.H. and G.H.; methodology, J.A.; project administration, X.L.; supervision, X.L. and J.H.; writing—original draft, J.A and G.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Defense Science and Technology Innovation Zone of China, grant number 00205501.

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

References

1. Zhang, Y.; Wang, W.; Sun, J.; Chang, H.; Huang, P. A Self-Reconfiguration Planning Strategy for Cellular Satellites. *IEEE Access* **2019**, *7*, 4516–4528. [CrossRef]
2. An, J.; Li, X.; Zhang, Z.; Man, W.; Zhang, G. Joint Trajectory Planning of Space Modular Reconfigurable Satellites Based on Kinematic Model. *Int. J. Aerosp. Eng.* **2020**, *2020*, 8872788. [CrossRef]
3. Liang, X.; Takeda, Y. An iterative method for the inverse kinematics of lower-mobility parallel mechanism with three RS or SR chains based on kinematically equivalent mechanism. *Mech. Mach. Theory* **2019**, *141*, 40–51. [CrossRef]
4. Lee, C.S.G.; Ziegler, M. Geometric Approach in Solving Inverse Kinematics of PUMA Robots. *IEEE Trans. Aerosp. Electron. Syst.* **1984**, *6*, 695–706. [CrossRef]
5. Nearchou, A.C. Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm. *Mech. Mach. Theory* **1998**, *33*, 273–292. [CrossRef]
6. Chembuly, V.V.M.J.S.; Voruganti, H.K. An Optimization Based Inverse Kinematics of Redundant Robots Avoiding Obstacles and Singularities. In Proceedings of the Advances in Robotics, New Delhi, India, 28 June–2 July 2017.
7. Lopez-Franco, C.; Hernandez-Barragan, J.; Alanis, A.Y.; Arana-Daniel, N. A soft computing approach for inverse kinematics of robot manipulators. *Eng. Appl. Artif. Intel.* **2018**, *74*, 104–120. [CrossRef]
8. Jiokou Kouabon, A.G.; Melingui, A.; Mvogo Ahanda, J.J.B.; Lakhal, O.; Coelen, V.; Kom, M.; Merzouki, R. A Learning Framework to inverse kinematics of high DOF redundant manipulators. *Mech. Mach. Theory* **2020**, *153*, 103978. [CrossRef]
9. Ananthanarayanan, H.; Ordóñez, R. Real-time Inverse Kinematics of (2n + 1) DOF hyper-redundant manipulator arm via a combined numerical and analytical approach. *Mech. Mach. Theory* **2015**, *91*, 209–226. [CrossRef]
10. Bjoerlykhaug, E. A Closed Loop Inverse Kinematics Solver Intended for Offline Calculation Optimized with GA. *Robotics* **2018**, *7*, 7. [CrossRef]
11. Ahmed, E.-S.; Mostafa, A.E.; Amira, Y.H. A comparative study of soft computing methods to solve inverse kinematics problem. *Ain Shams Eng. J.* **2018**, *9*, 2535–2548.
12. El-Sherbiny, A.; Elhosseini, M.A.; Haikal, A.Y. A new ABC variant for solving inverse kinematics problem in 5 DOF robot arm. *Appl. Soft Comput.* **2018**, *73*, 24–38. [CrossRef]

13. Zhang, L.; Xiao, N. A novel artificial bee colony algorithm for inverse kinematics calculation of 7-DOF serial manipulators. *Soft Comput* **2017**, *23*, 3269–3277. [CrossRef]
14. Dereli, S.; Kker, R. A meta-heuristic proposal for inverse kinematics solution of 7-DOF serial robotic manipulator: Quantum behaved particle swarm algorithm. *Artif. Intell. Rev.* **2020**, *53*, 949–964. [CrossRef]
15. Sancaktar, I.; Tuna, B.; Ulutas, M. Inverse kinematics application on medical robot using adapted PSO method. *Eng. Sci. Technol. Int. J.* **2018**, *21*, 1006–1010. [CrossRef]
16. Fan, B.; Liang, Z. Omnidirectional kick in RoboCup3D simulation. In Proceedings of the 2014 IEEE International Conference on Mechatronics and Automation, Tianjin, China, 3–6 August 2014.
17. Nguyen, T.; Nguyen, H.; Dang, K.; Nguyen, P.; Pham, H.; Bui, A. *Simulation and Experiment in Solving Inverse Kinematic for Human Upper Limb by Using Optimization Algorithm*; Springer International Publishing: Cham, Switzerland, 2021; pp. 556–568.
18. Zhang, Q.; Wang, D.; Gao, L. Research on the inverse kinematics of manipulator using an improved self-adaptive mutation differential evolution algorithm. *Int. J. Adv. Robot. Syst.* **2021**, *18*, 172988142110144. [CrossRef]
19. Khaled Mohamed, H.E.; Elsharkawy, A. Dynamic analysis with optimum trajectory planning of multiple degree-of-freedom surgical micro-robot. *Alex Eng. J* **2018**, *57*, 4103–4112. [CrossRef]
20. Wolpert, D.H.; Macready, W.G. No Free Lunch Theorems for Optimization. *IEEE Trans. Evol. Comput.* **1997**, *1*, 67–82. [CrossRef]
21. Hu, J.; Chen, H.; Heidari, A.A.; Wang, M.; Zhang, X.; Chen, Y.; Pan, Z. Orthogonal learning covariance matrix for defects of grey wolf optimizer: Insights, balance, diversity, and feature selection. *Knowl.-Based Syst.* **2021**, *213*, 106684. [CrossRef]
22. Şenel, F.A.; Gökçe, F.; Yüksel, A.S.; Yiğit, T. A novel hybrid PSO–GWO algorithm for optimization problems. *Eng. Comput.* **2019**, *35*, 1359–1373. [CrossRef]
23. Niu, B.; Zhu, Y.; He, X.; Wu, H. MCPSO: A multi-swarm cooperative particle swarm optimizer. *Appl. Math. Comput.* **2007**, *185*, 1050–1062. [CrossRef]
24. Xu, P.; Luo, W.; Lin, X.; Qiao, Y.; Zhu, T. Hybrid of PSO and CMA-ES for Global Optimization. In Proceedings of the 2019 IEEE Congress on Evolutionary Computation (CEC), Wellington, New Zealand, 10–13 June 2019.
25. Xu, Y.; Ye, Q.; Hoorfar, A. Surface Reconstruction of Large Reflector Antennas Based on a Hybrid of CMA-ES and HIO Algorithms. In Proceedings of the 2019 IEEE International Symposium on Antennas and Propagation and USNC-URSI Radio Science Meeting, Atlanta, GA, USA, 7–12 July 2019.
26. Chander, A.; Chatterjee, A.; Siarry, P. A new social and momentum component adaptive PSO algorithm for image segmentation. *Expert Syst. Appl.* **2011**, *38*, 4998–5004. [CrossRef]
27. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the 1998 IEEE International Conference on Evolutionary Computation Proceedings, Anchorage, AK, USA, 4–9 May 1998.
28. Shi, Y.; Eberhart, R.C. Empirical study of particle swarm optimization. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99, Washington, DC, USA, 6–9 July 1999.
29. Jiao, B.; Lian, Z.; Gu, X. A dynamic inertia weight particle swarm optimization algorithm. *Chaos Soliton Fract.* **2008**, *37*, 698–705. [CrossRef]
30. Clerc, M. The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization. In Proceedings of the 1999 Congress on Evolutionary Computation-CEC99, Washington, DC, USA, 6–9 July 1999.
31. Lee, T.-Y.; Chen, C.-L. Unit commitment with probabilistic reserve: An IPSO approach. *Energy Convers. Manag.* **2007**, *48*, 486–493. [CrossRef]
32. He, S.; Wu, Q.; Wen, J.; Saunders, J.; Paton, R. A particle swarm optimizer with passive congregation. *Biosystems* **2004**, *78*, 135–147. [CrossRef]
33. Mao, B.; Xie, Z.; Wang, Y.; Handroos, H.; Wu, H.; Shi, S. A hybrid differential evolution and particle swarm optimization algorithm for numerical kinematics solution of remote maintenance manipulators. *Fusion Eng. Des.* **2017**, *124*, 587–590. [CrossRef]
34. Seyyedabbasi, A.; Aliyev, R.; Kiani, F.; Gulle, M.U.; Basyildiz, H.; Shah, M.A. Hybrid algorithms based on combining reinforcement learning and metaheuristic methods to solve global optimization problems. *Knowl.-Based Syst.* **2021**, *223*, 107044. [CrossRef]
35. Yang, Q.; Dong, N.; Zhang, J. An enhanced adaptive bat algorithm for microgrid energy scheduling. *Energy* **2021**, *232*, 121014. [CrossRef]
36. Zhang, M.; Long, D.; Qin, T.; Yang, J. A Chaotic Hybrid Butterfly Optimization Algorithm with Particle Swarm Optimization for High-Dimensional Optimization Problems. *Symmetry* **2020**, *12*, 1800. [CrossRef]
37. Yang, J.; Cui, J.; Zhang, Y.-D. Artificial bee colony algorithm with adaptive covariance matrix for hearing loss detection. *Knowl.-Based Syst.* **2021**, *216*, 106792. [CrossRef]