An efficient constraint method for solving planning problems under end-effector constraints

Yahao Wang, Zhen Li, Yanghong Li and Erbao Dong
Department of Precision Machinery and Precision Instrumentation, University of Science and Technology of China, Hefei, China

Abstract

Purpose – In response to the challenge of reduced efficiency or failure of robot motion planning algorithms when faced with end-effector constraints, this study aims to propose a new constraint method to improve the performance of the sampling-based planner.

Design/methodology/approach – In this work, a constraint method (TC method) based on the idea of cross-sampling is proposed. This method uses the tangent space in the workspace to approximate the constrained manifold pattern and projects the entire sampling process into the workspace for constraint correction. This method avoids the need for extensive computational work involving multiple iterations of the Jacobi inverse matrix in the configuration space and retains the sampling properties of the sampling-based algorithm.

Findings – Simulation results demonstrate that the performance of the planner when using the TC method under the end-effector constraint surpasses that of other methods. Physical experiments further confirm that the TC-Planner does not cause excessive constraint errors that might lead to task failure. Moreover, field tests conducted on robots underscore the effectiveness of the TC-Planner, and its excellent performance, thereby advancing the autonomy of robots in power-line connection tasks.

Originality/value – This paper proposes a new constraint method combined with the rapid-exploring random trees algorithm to generate collision-free trajectories that satisfy the constraints for a high-dimensional robotic system under end-effector constraints. In a series of simulation and experimental tests, the planner using the TC method under end-effector constraints efficiently performs. Tests on a power distribution live-line operation robot also show that the TC method can greatly aid the robot in completing operation tasks with end-effector constraints. This helps robots to perform tasks with complex end-effector constraints such as grinding and welding more efficiently and autonomously.

Keywords Path planning, Autonomous robots

Paper type Research paper

1. Introduction

As robotics continues to evolve, field service robots are gradually venturing into increasingly complex scenarios to undertake challenging tasks across various domains. Given the unstructured nature of the scenarios and the distinct characteristics of the tasks, robots must swiftly respond to generate motions that adhere to constraints. This scenario necessitates stringent efficiency standards for the robot's motion planner. Moreover, the imperative to execute manipulation tasks that involve additional geometric constraints is a practical consideration that cannot be disregarded (Brock and Khatib, 2002). In Figure 1, the robot's end-effector is constrained during grinding, welding and field tasks (Qiu and Cao, 2018). In Figure 1(a), we constructed a power distribution live-line operation robot (PDLOR) that needs to perform tasks related to connecting power lines in overhead power lines. During the operation, the PDLOR has to move a wire that is attached to a lightning arrester at one end. This task requires the designer to consider additional end effector constraints.

In that case, the robot's motion planner faces challenges. First, planning in a high-dimensional space is inherently difficult, especially because of its spatial complexity (LaValle, 2006). Additionally, the introduction of end-effector constraints reduces the dimensionality of the free configuration space (CS), forcing the base version of the planner to reassess its validity (Kingston *et al.*, 2018).

Sample-based planning is commonly acknowledged to sustain optimal performance in high-dimensional spaces (Choset *et al.*, 2005). This strategy bypasses the need to compute the entire free CS by verifying each sampling point's validity (Shang *et al.*, 2023). However, conducting direct sampling of valid configurations is usually unfeasible because manifolds are implicitly defined by constraints and lack analytical formulas (Palleschi *et al.*, 2022; Kingston and Kavraki, 2023). Some researchers have proposed incorporating constraints into the sampling-based planner by adding a constraint module (Kingston *et al.*, 2018, 2019). A modular planner design enables the easy integration of

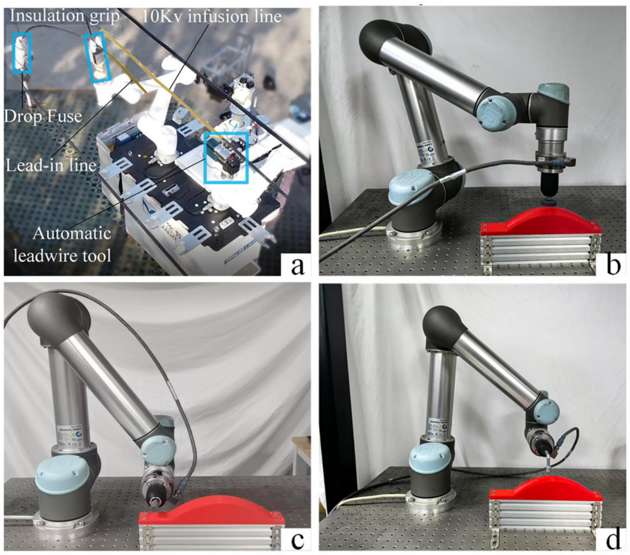
The current issue and full text archive of this journal is available on Emerald Insight at: **https://www.emerald.com/insight/0143-991X.htm**

Industrial Robot: the international journal of robotics research and application
51/3 (2024) 413–423

[DOI 10.1108/IR-10-2023-0251]

Received 23 October 2023
Revised 10 December 2023
20 January 2024
Accepted 31 January 2024

Figure 1 The cases that would be constrained by the end-effector



Notes: (a) PDLOR that completes power-line connection task; (b) cutting robot; (c) grinding robot; (d) welding robot
Source: Authors' own work

constraint methods into the algorithm without affecting its core computational process. In recent research, Projection (Lamiraux and Mirabel, 2022) and Atlas (Bordalba *et al.*, 2018, 2021; Jaillet and Porta, 2013) have emerged as two promising constraint methods. Although these two methods have been demonstrated to be effective (Qureshi *et al.*, 2022), the process of iterating through the Jacobi inverse matrix multiple times imposes significant computational complexity. It is challenging to achieve the required planning speed for field operations.

Certain techniques can directly produce end-effector trajectories in the workspace, resulting in greater efficiency (Rakita *et al.*, 2019). Constraints can be sampled in the constraint subspace within the workspace because they can be described analytically in this area (Rakita *et al.*, 2021). However, it is important to note that the workspace is fundamentally distinct from the CS. This method cannot ensure completeness and may not meet the exacting planning success rate standards required for field operations.

We therefore propose a new constraint method, the transformation cross-sampling method (TC method), which offers significant efficiency advantages. Unlike other constraint methods, it operates crosswise in workspace and CS. The expansion process of the planner is moved from the CS to the workspace for constraint processing with minimal computational expense. The constraint-processed data is subsequently conveyed back to the CS. This prevents the substantial computational burden of multiple iterations of the Jacobi inverse matrix for processing constraints solely in the CS and the potential lack of completeness in planning when subject to constraints in the workspace.

2. Preliminaries

To more systematically describe our work, this section presents the mathematical background of motion planning under end-effector constraints and related terminology.

The aim of this work is to address the problem of motion planning under end-effector constraints for operational robots, which involves finding feasible paths for the robot to avoid collisions with obstacles while ensuring that the end-effector is in the desired attitude. Regardless of the degree of freedom of a robotic system and the complexity of its structure, it is always modelled as a point q in the CS. The dimension of this space is equal to the robot's degree of freedom (or rather the number of joints):

$$q \in CS : \mathbb{R}^k \tag{1}$$

Any point in this space is reachable by the robot without considering certain constraints, such as collisions. Consequently, the space is bounded. Furthermore, we use Euclidean distances instead of calculating the distance between any two points in that space. Accordingly, the CS is a complete metric space. Usual planning occurs in the CS rather than the workspace to ensure completeness.

In basic motion planning, obstacle avoidance constraints must be considered. The position and shape of obstacles are always described in the work space (WS). The robot system under the obstacle avoidance constraint can no longer reach the obstacle region. At this point, a subspace CS_{free} under a CS can be determined as follows:

$$CS_{free} = \{q | q \in CS - CS_{ob}\}. \tag{2}$$

This concept indicates that the robot system satisfies the obstacle avoidance constraints at point q. Given that the position and shape of obstacles are always described in the workspace, the process of projecting them into the CS and partitioning the subspace in this way comes with a substantial computational cost. The rejection sampling strategy is typically used to ensure that the robotic system satisfies the obstacle avoidance constraints. This strategy discards the sampling points that do not satisfy the constraints and ensures that the retained sampling points satisfy the obstacle avoidance constraints. The goal of basic motion planning is to find a path from the initial position $q_{initial}$ to a specified position q_{goal}. The general motion planning problem can be described as finding the curve from the initial point $qinital$ to the target point q_{goal} within CS_{free}, which is a continuous injective map σ:[0,1] $\rightarrow CS_{free}$.

However, this work aims at solving the motion planning problem under end-effector constraints for operational robots, which involves finding feasible paths for the robot to avoid collisions with obstacles while ensuring that the end-effector is in the desired attitude. Here, constraints are often represented through the constraint function $H(X)$, then the constrained manifold M_W in the workspace can be obtained as:

$$X = [x, y, z, \alpha, \beta, \gamma]^T$$
$$M_X = \{X \in WS | H(X) = 0\}. \tag{3}$$

where the 6-D vector $X \in WS$: $se(3)$ denotes the end-effector of the robotic system. As shown in Figure 2, the constraint function in the CS can be obtained $F(q)$: $\mathbb{R}^n \rightarrow \mathbb{R}^k$, and we can define a constrained manifold MC:

$$F(q) = H(FK(q))$$
$$MC = \{q \in CS_{free} | F(q) = 0\}. \tag{4}$$

where $FK()$ denotes the positive kinematics of the robot system. At this point the motion planning problem under the end-effector can be defined as: σ:[0,1] $\rightarrow MC$.

The end-effector constrained manifold MC is a zero-measurement manifold. Nevertheless, the random sampling point happens to lie in the constrained manifold. Consequently, the strategy of rejecting sampling almost fails. The main focus of this study is how to ensure that the random tree satisfies the bit-wise constraint when expanding towards the random sampling point. One effective solution is to project each extension of the random tree onto the constrained manifold. Although the Jacobi pseudo-inverse projection techniques are generally accepted for motion planners with orientation or motion closure constraints, these methods encounter several technical challenges, such as avoiding joint constraints and singularities, iteration and computational efficiency.

3. Our algorithm

In this section, we provide an example using the rapid-exploring random trees (RRT) algorithm to illustrate the integration of the constraint module in the planner. Subsequently, we explain how the TC method is implemented.

3.1 Cross-sampling in the configuration space and workspace

To methodically introduce the execution process of our constraint method, we use the RRT algorithm as an example of how to find a path connecting the initial point q_{intial} and the goal point q_{goal} on M_W defined under the end-effector constraint C.

First, the random tree *Tree* is initialized with q_{rand}. Then the random configure q_{rand} is obtained by randomly sampling in the unexplored *CS*. Please note that q_{rand} may not necessarily lie on M_W. As the description of the constrained manifold type display is not available in the CS, it cannot be randomly sampled on M_W.

The *Tree* is then traversed to find the node $Node_{near}$ closest to q_{rand}. Next, the *Tree* extends towards q_{rand} from $Node_{near}$. A configuration q_{new} is obtained by ExtendTree($Node_{near}$, q_{rand}). This extension is assumed to be successful, and this configuration q_{new} satisfies the end-effector c constraints. ExtendTree() is the core function that performs the constraint processing, which will be described in detail below. To determine if the *Tree* is successfully extended, the validity of q_{new} must be checked, primarily for collision detection. If it passes the test, q_{new} should be included in the *Tree*. The abovementioned process is then repeated until the *Tree* reaches the target point q_{goal}. Collision constraints are introduced using the technique of rejection sampling.

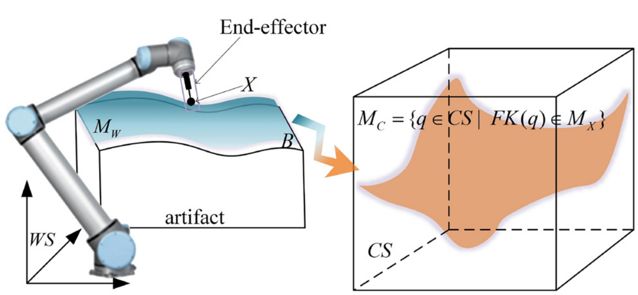
Algorithm 1 TC-RRT($q_{initial}, q_{goal}, C$)

```
Tree ← InitialTree(q_initial)
while Distance(Tree, q_goal) > d_0 do
    q_rand ← Random()
    Node_near ← findNear(Tree)
    motion ← TreeExtend(Node_near, q_rand)
    q_new ← TCmodule(motion, η, ε)
    if IsValidity(q_new) then
        Tree ← Update(Tree, q_new)
    end if
end while
Path ← FindPath(Tree)
```

In Figure 3, ExtendTree() serves to grow the random tree towards q_{rand} to obtain new nodes. When q_{rand} is constantly updated, the *Tree* can be guided to explore the CS. Under the end-effector constraint, the *Tree* is required to grow along the geodesic line pointing to q_{rand}. In that case, the guiding role of q_{rand} is preserved to explore the constrained manifold. The constraint module TCmodule() must be added to realize this function.

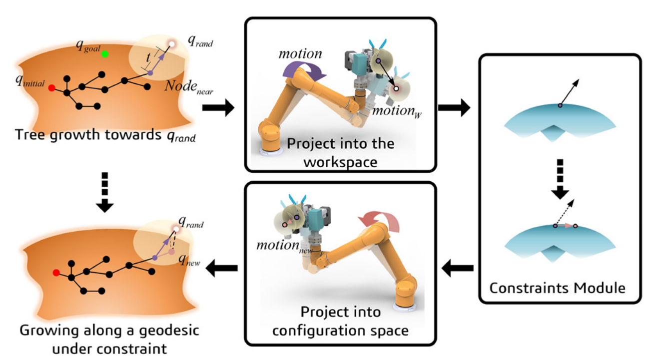
The nodes of the existing *Tree* satisfy the constraints. The node $Node_{near}$ in the *Tree* expanding once toward *qrand* in step *t*

Figure 2 The end of the grinding robot is constrained to the workpiece surface M_X, while the constraint creates a constrained manifold in the configuration space MC



Source: Authors' own work

Figure 3 The extend of the random tree is projected into the workspace and corrected under constraints via the constraint module



Note: The mapping back to the configuration space guides the extend
Source: Authors' own work

Industrial Robot: the international journal of robotics research and application
Volume 51 · Number 3 · 2024 · 413–423

corresponds to the robot system, which can be equated to the robot system undergoing a differential motion:

$$\begin{aligned} v &= \frac{q_{rand} - node_{near}}{|q_{rand} - node_{near}|} \\ motion &= v * t. \end{aligned} \quad (5)$$

The corresponding end-effector also undergoes a motion $motion_W$. When t is small enough, we can get a linear approximation of it by forward kinematics (FK):

$$\begin{aligned} v_W &= Jacobi(node_{near}) * v \\ \lim_{t \to 0} motion_W &= v_W * t. \end{aligned} \quad (6)$$

$motion_W$ is the projection of the expanded motion of the *Tree* into the *WS*, which clearly does not satisfy the constraints. Displaying the constraints in the *WS* allow for it to be constrained. By extracting the section that satisfies the constraints, the bootstrapping effect of q_{rand} can be maintained. Hence, the original properties of the planning method can be preserved. The detailed processing (TCmodule()) is described in the next section. This work is assumed that the constraint processing has been successfully performed, and potential node X_{new} in the *WS* is obtained. This node represents the ideal pose of the end-effector after the constraint processing, and the inverse kinematics (IK) can be used to obtain the configuration q_{new} of the robot in this posture X_{new}. At this point, q_{new} is considered to be the former potential node obtained from this extend, which is located on M_W. Finally, *Tree* can be guided to explore M_W by q_{rand} constant updating by incorporating a constraints module in the planning algorithm. Accordingly, a path can be found on the M_W connecting the initial point q_{intial} and the goalpoint q_{goal}. Pseudo-code for RRT based on TC constraint approach is given in Algorithm 1.

3.2 Constraint compliance in the workspace

The growth process of the randomized tree must be processed with corrections under the constraints to incorporate end-effector constraints in the planner. The previous section describes how our approach crosses over in the CS. Meanwhile, this section describes in detail how the processing TCmodule() under constraints takes effect in *WS*.

As previously discussed, a single extend of a random tree can be projected into the workspace, described by $motion_W$. $motion_W$ is projected onto a constrained manifold M_W in the workspace. Meanwhile, obtaining the projection of $motion_W$ onto M_W becomes a process of finding the geodesics. Although M_X in the *WS* can be made explicit, the process still requires a large computational overhead. Accordingly, a local linear approximation is used here to reduce the computational cost. A constraint function C:$H(X)$ that can explicitly describe the M_W is given, where the $X \in se(3)$ expression corresponds to the bit position of the end-effector of the robot system. This expression can be further processed as follows:

$$\begin{aligned} &C: H(X) = 0 \\ &H(X_{near}) + \mathrm{d}H(X_{near}) * \mathrm{d}X = 0. \end{aligned} \quad (7)$$

When considering X_{near} as the end-effector position of the robot system that corresponds to a node in the *Tree*, the pose satisfies the constraint $H(X_{near}) = 0$. The primary motion of the end-effector must satisfy equation (7). The position of the end-effector after the motion is guaranteed to satisfy the constraints. Thereafter, the following expression is obtained:

$$\mathrm{d}H(X_{near}) * \mathrm{d}X = 0. \quad (8)$$

In Figure 5, the constrained manifold pattern is replaced by a hyperplane P in the X_{near} neighbourhood. The hyperplane P is a subspace in the se(3) space, which can be defined by $\boldsymbol{\eta}$:

$$\begin{gathered} \boldsymbol{\eta} = [\eta_1, \eta_2 \cdots \eta_{n-k}], \mathbf{k} = [k_1, k_2 \cdots k_{n-k}] \\ \lim_{X \to X_0} M_X = P(X_{near}) = \{X_{near} + \mathbf{k} * \boldsymbol{\eta}\} \\ C: X \in M_W \Rightarrow C: \boldsymbol{\eta}. \end{gathered} \quad (9)$$

where $\boldsymbol{\eta}$ is the underlying solution system of G(X). $n-k$ represents the accumulation of constraints, with larger representing more complex constraints. Then the constraint can be expressed as C:$\boldsymbol{\eta}$. Thereafter, the projection of v_W on each basis is separately computed and superimposed to extract the part of v_{new} in $motion_W$ that satisfies the constraints:

$$\begin{aligned} v_{new} &= \sum_{i=1}^{n-k} v_W * \eta_i \\ motion_{new} &= v_{new} * t. \end{aligned} \quad (10)$$

The feasible nodes in the workspace resulting from constraint processing can be acquired through an affine transformation. Afterwards, the feasible node in the CS can be obtained by inverse kinematics.

$$\begin{aligned} X_{new} &= \mathrm{Transf}(X_{near}, motion_{new}) \\ q_{rand} &= \mathrm{IK}(X_{new}, X_{near}) \end{aligned} \quad (11)$$

This node is considered to satisfy the constraints and evaluate the extent to which the potential node deviates from the constraints by parameter ε in the workspace.

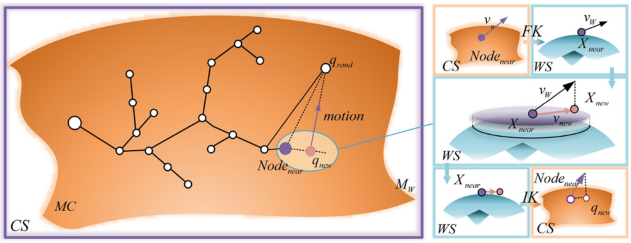
Algorithm 2 TCmodule($motion, \boldsymbol{\eta}, \varepsilon$)

```
[v, Node_near, d_0] ← Motion
v_W ← Jacobi(Node_near) * v
X_near ← FK(Node_near)
v_new = Σ v_W * η_i
motion_new = v_new * t
X_new ← Transf(X_near, motion_new)
q_new ← IK(X_new)
if ErrorCheck(q_new, ε) then
    return q_new
    return NULL
end if
```

This deviation is introduced by the linear approximation, which is also considered an error introduced by the constraining process. However, this error can be accumulated. Accordingly, the accumulation of deviations from the constraint is limited by introducing a constraint tolerance error ε, similar to the other constraint methods (Fusco *et al.*, 2019; Bonilla *et al.*, 2017).

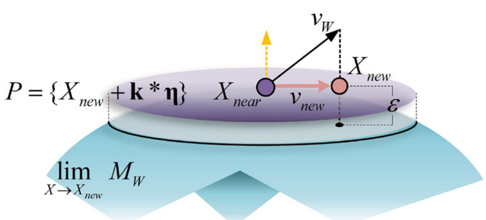
In Figure 4, the process of correcting the random tree growth process under the constraints by the abovementioned processing was completed. The method proposed in this work

Figure 4 Process of expanding a random tree



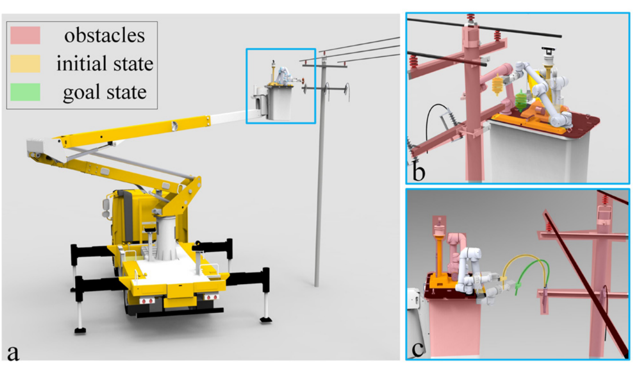
Notes: Expansion operations under constraints are performed by intersecting the reconfiguration space and the workspace, ensuring that the extend is performed along the geodesic of the constrained manifold pattern
Source: Authors' own work

Figure 5 Neighbourhood of X_{near} on the constrained manifold pattern in the workspace; the constraint manifold can be approximated as a hyperplane P



Source: Authors' own work

Figure 6 The PDLOR in the simulation environment needs to avoid obstacles and operate the object to reach the target location



Notes: (a) Simulated overhead lines; (b) moving the arrester task, (c) moving the conductor task
Source: Authors' own work

crosses in the *CS* and the *WS*, preserving the bootstrapping role of q_{rand} in the *WS*. Meanwhile, the constraint processing in the *WS* avoids the huge computational effort of multiple iterations using the Jacobi inverse matrix in the CS. The pseudo-code of this method is provided in Algorithm 2.

4. Simulation

In the simulation experiments, the performance of the TC constraint method is tested including: planning speed test and greediness lowercase test. To evaluate the efficiency of the TC-constrained method, Projection and Atlas were chosen for comparison, which are the most widely used and maintain high efficiency (Kingston *et al.*, 2019). During testing, we implemented three constraint planners – the TC method, the Projection method and the Altas method – into the RRT planner. To maintain fairness, all planners for the benchmarking were run on the same workstation using Intel processors and the same set of planners utilized for comparison used the same planner parameters. The workstation was configured with a CoreTM i7-8700K processor and 16 GB 2400 MHz DDR 4 RAM.

As shown in Figure 6, two planning tasks with different levels of difficulty are given in a known distribution overhead line environment. PDLOR is required to move lightning arrestors in Task 1. In addition to avoiding collisions with obstacles, the robot must ensure that the lightning arrester's position remains steady to prevent any potential chemical leakage. In Task 2, the PDLOR is required to move the wire, and in addition to satisfying the obstacle avoidance constraints, the fixed position of the end of the wire is considered to be a ball-hinge constraint. The level of the end-effector constraints imposed by the two tasks are 2 and 3, and the two constraints are expressed through C_1 and C_2:

$$C_1 : H(X) = 0 = \begin{cases} \alpha - \alpha_0 \\ \beta - \beta_0 \end{cases}$$

$$C_2 : H(X) = 0 = \begin{cases} x - \sqrt{r^2 - z^2 - y^2} \\ \gamma - \mathrm{atan}(x, y) \\ \beta - cross(\gamma, \alpha) \end{cases}. \tag{12}$$

To observe the constraint method planner's greedy nature and the impact of constraint tolerances on the planner, we conducted experiments with three tolerance errors (0.003, 0.002 and 0.001) and three target deviations. Two tasks were used to test the performance of each of the three planners under these parameter settings. The parameter settings for the three planners are shown in Table 1.

4.1 Planning speed test

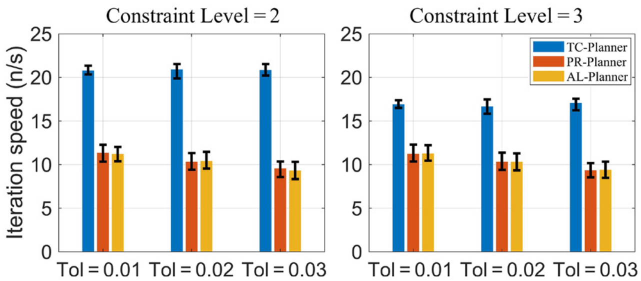
Figure 7 records the planning time results and planner iteration speeds for each of the three planners at three tolerance errors when the target deviation is 0.1. In Task 1, the TC-Planner exhibits the greatest iteration speed, followed by the Pr-Planner and the AL-Planner demonstrates the slowest iteration speed when the tolerance error is 0.001 and the target deviation is 0.1. Furthermore, the TC-Planner outperforms the Pr-Planner by 50%. This advantage remains constant as the tolerance error increases. Furthermore, for each planner individually, the iteration speed exhibits a certain degree of decrease as the restricted tolerance error increases. When the tolerance error was adjusted between 0.01 and 0.03, the TC-Planner observed

Table 1 Experimental setup

Planner	Method	Bias	Step	Tolerance error
TC-Planner	TC + RRT	(0.1,0.2,0.3)	0.06	(0.003, 0.002, 0.001)
Pr-Planner	Projection + RRT			
Al-Planner	Atlas + RRT			

Source: Authors' own work

Figure 7 Variation of iteration speed with tolerance error for three planners under different tasks



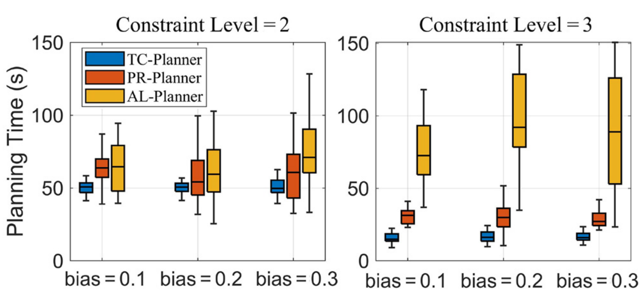
Source: Authors' own work

a 5% reduction in iteration speed, whereas the other two planners recorded a decrease of over 10%. This phenomenon is due to the fact that the amount of computation involved in the iterative operation process of the TC-planner remains unchanged despite the increase in tolerance. The only change is the probability of finding valid nodes. In contrast, the other two planners incur significantly increased computational cost in the iteration process. As the constraint level increases, the constraints become more intricate. In Task 2, the rate of iteration reduces when using the TC constraint strategy. In a complex constraint flow pattern, the probability of the algorithm discovering a valid node per iteration is lower. Nonetheless, in Task 2, the TC-Planner is just 20% faster than the Projection planning periodizer in acquiring valid nodes.

Based on the observation result of the planning time data recorded in Figure 8, the advantage in planning time in Task 2 is more pronounced than in Task 1. In Task 1, the TC-Planner with the lowest average planning time is 15% less than the Pr-Planner and 25% less than the Atlas. By contrast, in Task 2, these two parameters are improved to 50% and 75%. This situation arises because the constraint flow patterns become distorted as the constraints become complex. Then, the original approximation of the metric results in a decrease in the effectiveness of the exploration of the Pr-Planner and Al-Planner. This notion indicates that the TC constraint approach is equally efficient under complex constraints.

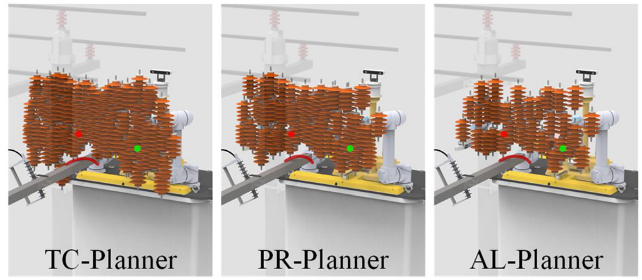
Figures 9 and 10 demonstrate the three planners exploring the constrained manifold at a target deviation of 0.1 and a tolerance error of 0.01 for the three tolerance errors. The nodes

Figure 8 Variation of planning time with tolerance error for three planners in different tasks



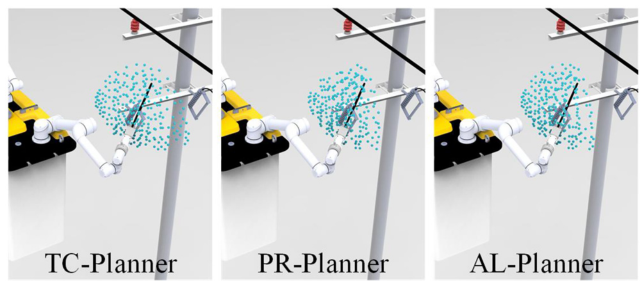
Source: Authors' own work

Figure 9 Configurations explored by different planners in one planning session in Task 1



Notes: Here, the configuration is explicitly represented as the arrester position corresponding to the configuration
Source: Authors' own work

Figure 10 Configurations explored by different planners in one planning session in Task 2



Notes: Here, the configuration is explicitly represented as the position of the tool gripping point corresponding to the configuration
Source: Authors' own work

generated by all three planners are guaranteed to be end-effector-constrained while not colliding with obstacles. The area of the nodes generated by the TC-Planner to cover the constrained manifold during the planning process is higher than that of the other planners. Covering a larger area of the constrained manifold means that a higher chance of finding a path connecting the start and goal points.

Table 2 provides the path length and planning success rate for each group of tests in the experiment. The results indicate that, for TC-Planner with the same parameter settings, the planning success rate in task one is the highest. This is to some extent the effect of iteration speed on the planning success rate. However, with the same parameters, TC-Planner, which has the fastest iteration speed compared to other planners, does not have the highest success rate. This may be caused by the randomness of the planning algorithm or the loss of greediness due to the TC constraint approach. Therefore, a further discussion is needed on the preservation of greediness by TC constraint methods. The variation between the path lengths of the three planners is within 5%.

Moreover, the TC method boasts lower computational complexity than other methods, which facilitates faster iterations for exploring multiple constrained manifold patterns. The trait of being more tolerant to tolerance errors also

Table 2 Experimental result of planning speed

Planner	Set	Path Length	Nodes	Success
TC-Planner	Task1, TOL=0.01	300.3	19.6	*19/20*
	Task1, TOL=0.02	290.2	19.3	*19/20*
	Task1, TOL=0.03	308.3	19.5	*18/20*
	Task2, TOL=0.01	50.1	16.4	*17/20*
	Task2, TOL=0.02	50.8	16.3	*18/20*
	Task2, TOL=0.03	52.1	16.2	*16/20*
Pr-Planner	Task1, TOL=0.01	297.2	10.2	*19/20*
	Task1, TOL=0.02	310.2	9.3	*19/20*
	Task1, TOL=0.03	280.2	7.9	*19/20*
	Task2, TOL=0.01	53.1	9.6	*19/20*
	Task2, TOL=0.02	54.1	8.5	*19/20*
	Task2, TOL=0.03	51.1	7.3	*19/20*
Al-Planner	Task1, TOL=0.01	301.2	9.2	*19/20*
	Task1, TOL=0.02	297.5	8.1	*19/20*
	Task1, TOL=0.03	302.4	6.6	*19/20*
	Task2, TOL=0.01	52.1	9.1	*19/20*
	Task2, TOL=0.02	51.6	8.5	*19/20*
	Task2, TOL=0.03	51.5	7.2	*19/20*

Source: Authors' own work

Table 3 Experimental result of greediness

Task	Set	Path Length	Iteration	Success
Task1	Bais=0.1, TOL=0.01	300.3	21.6	*19/20*
	Bais=0.2, TOL=0.01	272.2	21.2	*18/20*
	Bais=0.3, TOL=0.01	261.8	21.4	*20/20*
	Bais=0.1, TOL=0.02	290.2	21.1	*19/20*
	Bais=0.2, TOL=0.02	282.2	21.2	*20/20*
	Bais=0.3, TOL=0.02	258.7	21.2	*20/20*
	Bais=0.1, TOL=0.03	308.3	21.0	*18/20*
	Bais=0.2, TOL=0.03	252.4	20.9	*19/20*
	Bais=0.3, TOL=0.03	240.3	21.1	*20/20*
Task2	Bais=0.1, TOL=0.01	50.1	19.4	*19/20*
	Bais=0.1, TOL=0.01	45.8	19.2	*19/20*
	Bais=0.3, TOL=0.01	38.2	19.2	*20/20*
	Bais=0.1, TOL=0.02	50.8	19.2	*19/20*
	Bais=0.2, TOL=0.02	46.9	18.7	*19/20*
	Bais=0.3, TOL=0.02	42.8	18.4	*19/20*
	Bais=0.1, TOL=0.03	52.1	19.1	*18/20*
	Bais=0.2, TOL=0.03	41.2	17.7	*29/20*
	Bais=0.3, TOL=0.03	33.9	18.3	*20/20*

Source: Authors' own work

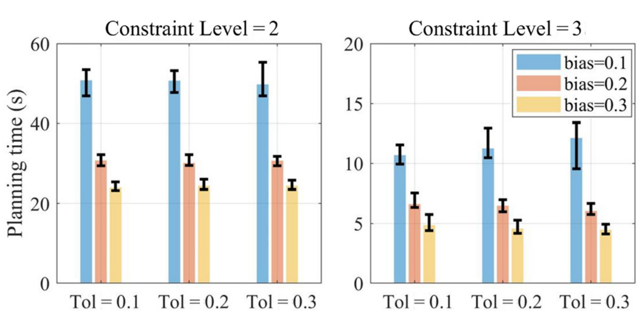
facilitates the TC method's ability to identify appropriate parameters for usage.

4.2 Greediness test

In this section, we discuss the preservation of planner greediness by the TC constraint approach. Figure 11 depicts the variation of the TC-planner's planning time with target deviation for the different tolerance errors in Tasks 1 and 2 to observe the effect of the TC method on the characteristics of the planning algorithms. Table 3 presents the speed of iterations, the path length and the planning success rate of the planners for each set of tests.

The speed of iteration by each group of planners does not show significant fluctuations as the goal deviation varies. Furthermore, the planning time of the TC-Planner goal planner significantly decreased as the goal deviation increases. The reason for this phenomenon is that the TC constraint method respects the bootstrapping effect of random points, thus preserving the greedy nature of the planning algorithm itself. Therefore, the TC method does not affect the asymptotic optimality of the planner.

Figure 11 Variation of the TC-Planner's planning time with target deviation for different tolerance errors in the two tasks



Source: Authors' own work

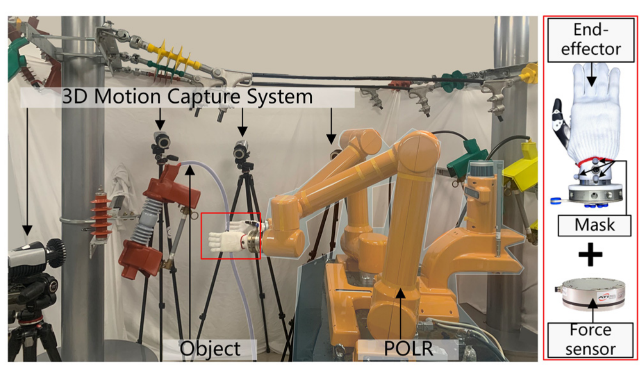
5. Experiment

5.1 Constrained compliance error experiment

Despite the tremendous advances in robotics(Wang *et al.*, 2023; Li *et al.*, 2023), PDLORs typically use teleoperation in moving wire tasks when performing power-line connection tasks. This phenomenon occurs because the actual robotic system will not fully enforce accurate compliance with the constraints due to the discrete data modelling of the sampling-based planner, which may lead to task failure. Accordingly, the TC constraint method is first tested in a physical test system for constraint enforcement errors. Then the feasibility of the method is verified on an overhead line.

In Figure 12, the experiments were conducted in a laboratory where an overhead line was simulated, to prevent electromagnetic interference and ease data collection. In this study, the PDLOR system is utilized to execute the planning results of three different

Figure 12 The motion trajectory of the moving wire of the PDLOR is captured by using 3 D motion capture recorders fixed at three mask positions on the end-effector



Source: Authors' own work

planners in simulation experiments aimed at accomplishing the moving wire task. The PDLOR comprises two UR10 robotic arms, along with multiple binocular cameras and LiDAR. The experiments utilized a 3D motion capture system and force sensors to accurately capture the real-time execution of constraints during runs. Additionally, the robot's end load was limited to 5 N, and if the stress surpassed this threshold, the task was deemed unsuccessful.

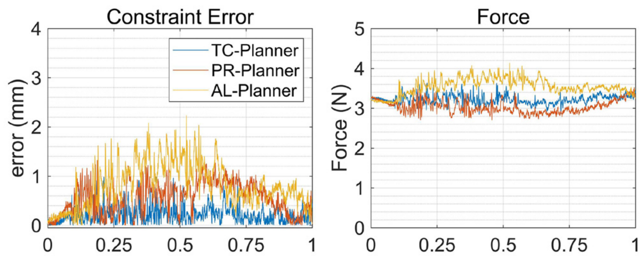
During the operation, the wire is modelled as a rigid body fixed at one end on a ball hinge. The position of the ball hinge is sensed by camera recognition and encoded to the three planners. Thereafter, the robotic system accurately executes the planning results given by the planners. Figure 13(a) shows the trajectories of the six joints, the trajectories of the end-effector in the workspace during the execution of the planning results of the three planners by the PDLOR. The trajectories in the CS are jittery in the absence of the trajectory optimisation process due to the discrete sampling nature of the RRT. Figure 13(b) shows the trajectories of the three target balls on the end-effector. The position of the end-effector can be calculated from the positions of the three target balls to further obtain the execution error of this motion on the constraint:

$$\begin{aligned} &\{P_1, P_2, P_3\} \Rightarrow Posture_{tool} = \begin{bmatrix} Position \\ Attitude \end{bmatrix} \\ &Attitude \Rightarrow Position_{SA} \\ &error = |Position_{SA}| \end{aligned} \tag{13}$$

where P_1, P_2 and P_3 denote the real-time recordings of the real-time positions of the three targets. The ideal position of the end-effector can be obtained from the attitude of the end-effector. Then error can be expressed as the distance between the actual and the ideal positions.

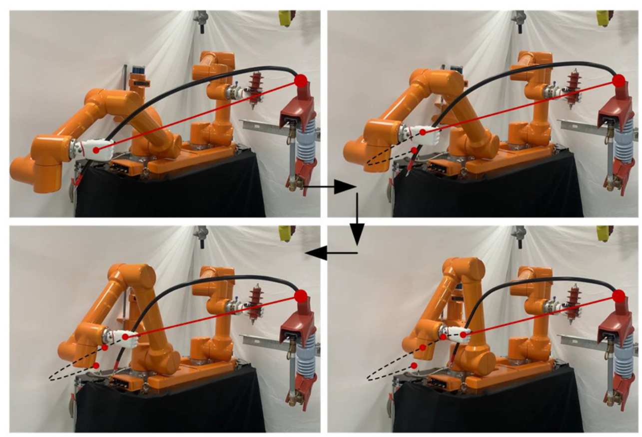
Figure 13 records the trajectories of the six joints during the execution of the planning results of the three planners by PDLOR and the trajectories of the end-effector in the workspace.

Figure 14 Change in constraint error and change in end force during the experiment



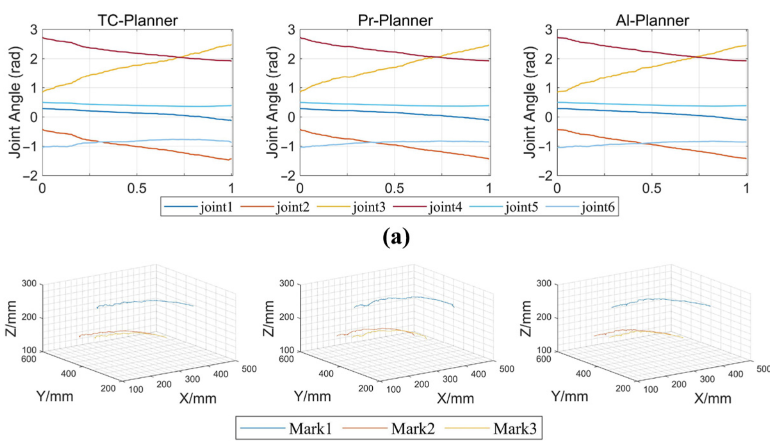
Note: 0 and 1 indicate task start and task completion
Source: Authors' own work

Figure 15 Case of running the planning results of TC-Planner on PDLOR



Source: Authors' own work

Figure 13 Result of the constraint error experiment



(a)

(b)

Notes: (a) Joint angle changes of PDLOR; (b)The end-effector trajectories of PDLOR during Experiment. 0 and 1 indicate task start and task completion
Source: Authors' own work

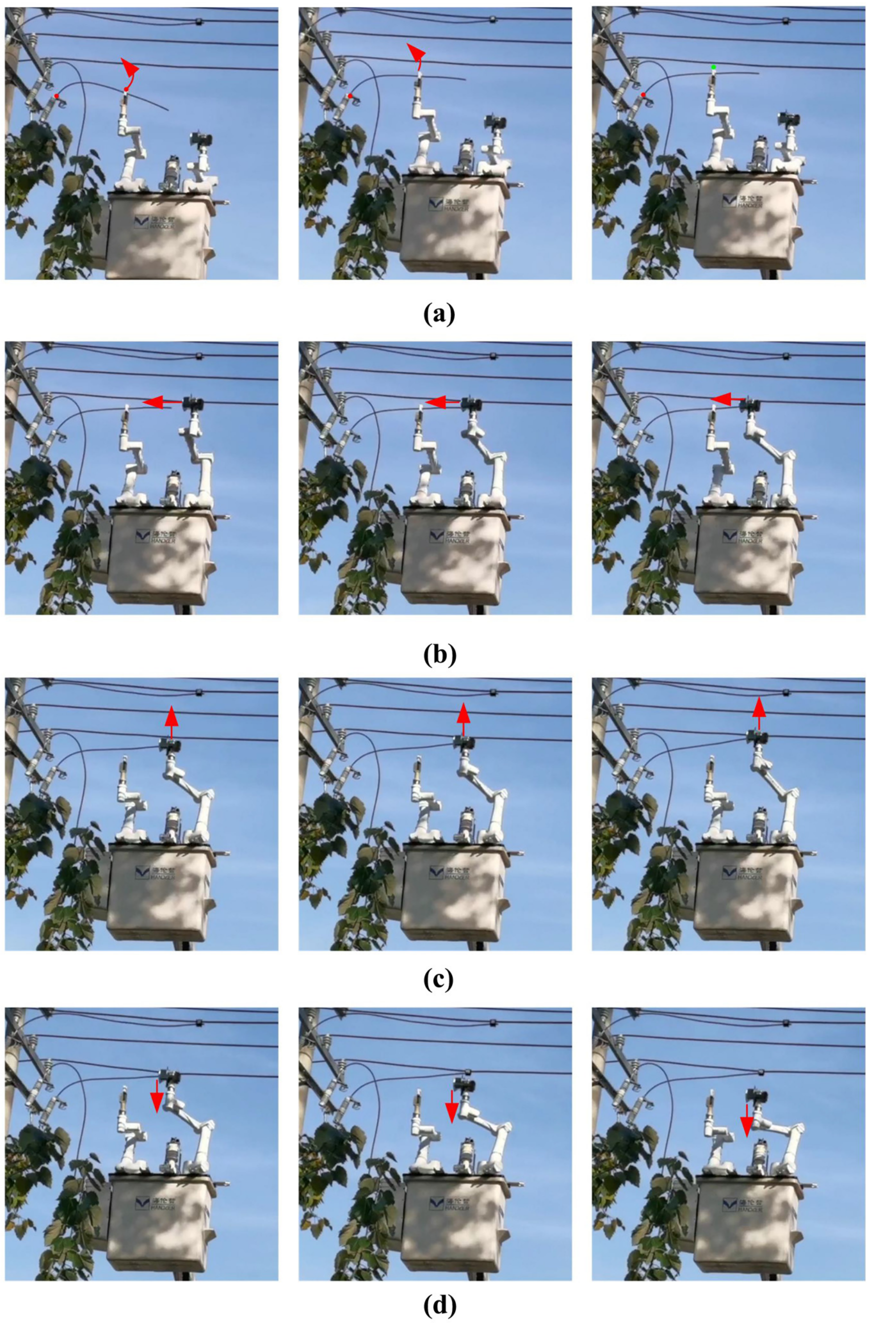
Industrial Robot: the international journal of robotics research and application
Volume 51 · Number 3 · 2024 · 413–423

The trajectories in the CS are jittery in the absence of the trajectory optimization due to the discrete sampling nature of the RRT. When this concept is reflected in the workspace, the jitter is more pronounced during the motion of the gripping tool.

Figure 14 shows the actuator error on the constraints during the execution of the planning results of the three planners by PDLOR and the forces on the end. The constraint execution errors of all three planners are managed within 3 mm. TC-Planner's execution error is managed within 1 mm. The stresses caused during motion do not surpass the load limit at the end. Furthermore, the TC-Planner generates the most stable result, and the stress variation caused by it is kept within 1 N. Meanwhile, the case of running the planning results of the TC-Planner on PDLOR is shown in Figure 15.

This work demonstrated that the TC-Planner has an efficiency that far exceeds that of similar algorithms. In addition, physical experiments indicated that performing the moving wire task for the planner generation by the TC-

Figure 16 Example of an autonomous operating system using the TC constraint method to complete power-line connection task



Source: Authors' own work

Planner avoids the generation of stresses beyond the limits.

5.2 Experiment on overhead line

Further in a PDLOR autonomous operating system is implemented through TC-Planner to perform distribution network maintenance work in real overhead lines. The semi-autonomous operating system (Chen *et al.*, 2023) is also compared to explore whether the TC-Planner helps PDLOR completes power-line connection tasks in a fully autonomous manner.

Twenty power-line connection tasks are executed with the semi-autonomous operating system and autonomous operational scenario. Figure 16 illustrates the workflow of autonomous operations. For executing power-line connection tasks, PDLOR moves the wire to the target position using the left arm and inserts the clamped wire into the automatic piercing connector installation tool held by the right arm via threading manoeuvre. Afterwards, the automatic piercing connector installation tool is lifted, and the wire is secured to the main conductor by the tool, ensuring that the current is conducted. Finally, the tool is lowered to complete the operation.

In the autonomous operating system employing TC-Planner, the vision system identifies the operational object whilst the constraints are integrated into TC-Planner to yield the required motion for every subtask. In contrast, the proposed semi-autonomous operating system performs wire movement task via teleoperation. The other subtasks are generated by paths projected from the workspace into the CS, though this approach does not fulfil completeness.

Twenty power-line connection tasks were carried out utilising the semi-autonomous operating system and the autonomous operational scenario provided by TC-Planner. Table 4 indicates the execution time and success rate of the two schemes in the experiment. The success rate of the autonomous operating system using TC-Planner in the experiments is higher than that of the semi-autonomous operating system, because the planning method generated directly in the workspace lacks completeness and fails to avoid singularities. Thus, it can lead to failure in generating trajectories. Also, the total operation time of the autonomous operating system using TC-Planner is better than the semi-autonomous system due to the time-consuming process of line grabbing using teleoperation.

6. Summary

This paper outlines a constraint-based approach for solving configurations that satisfy end-effector constraints. The approach incorporates the constraint compliance process in a hybrid space of the configuration and task spaces, thereby avoiding the significant computational burden associated with the use of multiple iterations of the Jacobi inverse matrix in the CS. The proposed constraint method is implemented as a constraint module in the entire planning process. Thus, the characteristics of the planning algorithm remain unchanged.

Furthermore, we compared our algorithm to the advanced constraint methods Altas and Projection through an experiment. As a result, the TC method planner shows superior performance under end-effector constraints. Moreover, our experimental results reveal that the TC-Planner is less sensitive to tolerance errors, which makes it more suitable for real robot systems. The physical experiments demonstrate that the TC-Planner does not result in excessive constraint errors that lead to task failure. Furthermore, tests conducted on field robots indicate the effectiveness of the TC-Planner in promoting robot autonomy for power-line connection tasks. Its outstanding performance further supports this notion.

Table 4 Execution time and success rate data for experiments on overhead line

	Time of subtask				
System	**Catch line**	**Thread**	**Hoist**	**Lowered**	**Success**
Auto	2.0min	2.1 min	2 0.4 min	2.3 min	*18/20*
Semi-auto	5.2min	3.1 min	1.6 min	0.5 min	*16/20*

Source: Authors' own work

References

Bonilla, M., Pallottino, L. and Bicchi, A. (2017), "Noninteracting constrained motion planning and control for robot manipulators", *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 4038-4043.

Bordalba, R., Ros, L. and Porta, J.M. (2018), "Randomized kinodynamic planning for constrained systems", *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 7079-7086.

Bordalba, R., Ros, L. and Porta, J.M. (2021), "A randomized kinodynamic planner for closed-chain robotic systems", *IEEE Transactions on Robotics*, Vol. 37 No. 1, pp. 99-115.

Brock, O. and Khatib, O. (2002), "Elastic strips: a framework for motion generation in human environments", *The International Journal of Robotics Research*, Vol. 21 No. 12, pp. 1031-1052.

Chen, Y.Y., Wang, X., Tang, K., Wu, S., Wu, R., Guo, Y., Feng, E. and Dong, N. (2023), "Intelligent power distribution live-line operation robot systems based on stereo camera", *High Voltage, Pages*, Vol. 8 No. 6, pp. 1-13.

Choset, H., Lynch, K., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L. and Thrun, S. (2005), *Principles of Robot Motion: Theory, Algorithms, and Implementations*, MIT Press, London, ISBN 0262033275.

Fusco, F., Kermorgant, O. and Martinet, P. (2019), "Improving relaxation-based constrained path planning via quadratic programming", in Strand, M., Dillmann, R., Menegatti, E. and Ghidoni, S. (Eds), *Intelligent Autonomous Systems*, Vol. 15, Springer International Publishing, Cham, pp. 15-26, ISBN 978-3-030-01370-7.

Jaillet, L. and Porta, J. (2013), "Efficient asymptotically-optimal path planning on manifolds", *Robotics and Autonomous Systems*, Vol. 61 No. 8, pp. 797-807. ISSN 0921-8890.

Kingston, Z. and Kavraki, L.E. (2023), "Scaling multimodal planning: using experience and informing discrete search", *IEEE Transactions on Robotics*, Vol. 39 No. 1, pp. 128-146.

Kingston, Z., Moll, M. and Kavraki, L.E. (2018), "Sampling-based methods for motion planning with constraints",

Annual Review of Control, Robotics, and Autonomous Systems, Vol. 1 No. 1, pp. 159-185.
Kingston, Z., Moll, M. and Kavraki, L. (2019), "Exploring implicit spaces for constrained sampling-based planning", *The International Journal of Robotics Research*, Vol. 38 No. 10-11, p. 027836491986853.
Lamiraux, F. and Mirabel, J. (2022), "Prehensile manipulation planning: modeling, algorithms and implementation", *IEEE Transactions on Robotics*, Vol. 38 No. 4, pp. 2370-2388.
LaValle, S.M. (2006), "Planning algorithms".
Li, Z., Yahao, W., Run, Y., Shaolei, W., Rui, G. and Dong, E. (2023), "An efficiently convergent deep reinforcement learning-based trajectory planning method for manipulators in dynamic environments", *Journal of Intelligent & Robotic Systems*, Vol. 107 No. 4.
Palleschi, A., Pollayil, G.J., Pollayil, M.J., Garabini, M. and Pallottino, L. (2022), "High-level planning for object manipulation with multi heterogeneous robots in shared environments", *IEEE Robotics and Automation Letters*, Vol. 7 No. 2, pp. 3138-3145.
Qiu, Q. and Cao, Q. (2018), "Task constrained motion planning for 7-degree of freedom manipulators with parameterized submanifolds", *Industrial Robot: An International Journal*, Vol. 45 No. 3, pp. 363-370.
Qureshi, A.H., Dong, J., Baig, A. and Yip, M.C. (2022), "Constrained motion planning networks x", *IEEE Transactions on Robotics*, Vol. 38 No. 2, pp. 868-886.
Rakita, D., Mutlu, B. and Gleicher, M. (2019), "Stampede: a discrete-optimization method for solving pathwise-inverse kinematics", International Conference on Robotics and Automation (ICRA), pp. 3507-3513.
Rakita, D., Shi, H., Mutlu, B. and Gleicher, M. (2021), "Collisionik: a per-instant pose optimization method for generating robot motions with environment collision avoidance", IEEE International Conference on Robotics and Automation (ICRA), pp. 9995-10001.
Shang, Y., Liu, F., Qin, P., Guo, Z. and Li, Z. (2023), "Research on path planning of autonomous vehicle based on RRT algorithm of q-learning and obstacle distribution", *Engineering Computations*, Vol. 40 No. 5.
Wang, Z., Liu, Y., Duan, Y., Li, X., Zhang, X., Ji, J., Dong, E. and Zhang, Y. (2023), "Ustc flicar: a sensors fusion dataset of Lidar-inertial-camera for heavy-duty autonomous aerial work robots", *The International Journal of Robotics Research*, Vol. 42 No. 11, pp. 1015-1047.

Corresponding author

Erbao Dong can be contacted at: ebdong@ustc.edu.cn