IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2026, Vienna, Austria. Cite as RA-L paper.

F-RRT: an Efficient Algorithm for Semi-Constrained Path Planning Problems

Guillaume de Mathelin de Papigny[1], Franco Gassibe[2] and Vincent Padois[3]

***Abstract*—This paper addresses the challenging problem of Semi-Constrained End-Effector Path Planning for robotic manipulators. This problem arises when complex specifications restrict the end-effector's motion during the execution of industrial tasks. Traditional path planning algorithms often struggle with such problems due to the difficulty of exploring the robot's valid configuration space, or constrained manifold, under these conditions. In this work, we propose a novel sampling-based approach that efficiently navigates the constrained manifold by exploring an alternative space representing the end-effector's degrees of freedom, such as process-related tolerances, throughout the task. This method retains the simplicity of sampling-based techniques. Building on this approach, we introduce the F-RRT algorithm, an adaptation of the renowned RRT planner [1]. F-RRT demonstrates enhanced speed and robustness compared to existing solutions, particularly in complex and cluttered environments.**

***Index Terms*—Constrained Motion Planning, Motion and Path Planning, Industrial Robots.**

I. Introduction

WHILE advances in robotics have significantly improved path planning techniques [2], many industrial tasks remain difficult to plan due to complex specifications of the possible end-effector motions. Industrial tasks such as water or sand blasting, arc welding, or gluing are typical examples. They belong to the class of semi-constrained end-effector tasks where the motion of the end-effector is not strictly constrained but defined as an interval for some task-specific directions. This generally leaves more degrees of freedom for path planning. For example, the 3D gluing task illustrated in Fig. 1 depicts such a case where the end-effector is semi-constrained in orientation inside an incidence cone. This type of path planning situation is commonly referred to as the *Semi-Constrained End-Effector Path Planning* (SCEEPP) problem or Semi-Constrained Cartesian Path Planning problem [3], [4]. The challenge in solving SCEEPP-problems lies in the effective exploration of the robot's valid configuration space, or constrained manifold, defined by the end-effector's constraints. The objective of this paper is to present a new sampling-based approach for effectively handling complex SCEEPP-problems that traditional path planning algorithms struggle with, while maintaining a low computational complexity with few hyperparameters to tune. The approach is based on representing the available degrees of freedom of the task as an axis-aligned object and exploring the valid constrained manifold of the configuration space through this axis-aligned representation. This approach avoids differentiating the constraints, maintains the simplicity of sampling-based methods, uses only a few hyperparameters, and reduces the risk of getting trapped in local minima in cluttered environments. To demonstrate the effectiveness of our approach, we present a new planner, F-RRT, which is an adaptation of the well-known RRT (Rapidly-exploring Random Trees) algorithm [1].

The structure of this paper is as follows. The next section presents an overview of related works about solving SCEEPP-problems. Section III introduces the Semi-Constrained End-Effector Path Planning problem. Then, the proposed approach and its underlying planner are described in section IV and compared with existing solutions in section V. Finally, future directions and conclusions are discussed in sections VI and VII.

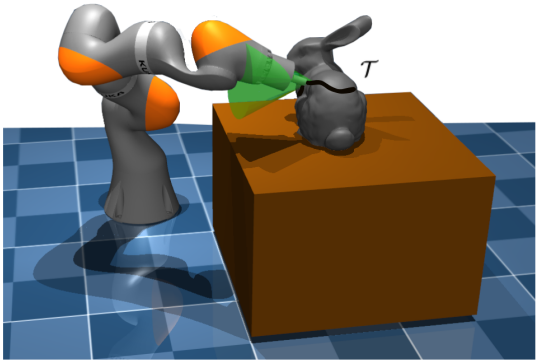

Fig. 1. The gluing task is a classic semi-constrained end-effector task where the end-effector is semi-constrained in orientation but is fully position constrained. The tip of the end-effector must follow the defined Cartesian path $\mathcal{T}$ inside an orientation cone.

II. Related Work

SCEEPP-problems are a specific subcategory of the Constrained Planning problem. A complete review of sampling-based planning methods for solving general Constrained Planning problems is provided in [5], where different methodologies are categorized. However, this review does not explore the SCEEPP-problems in detail. From [5], in the Open Motion Planning Library (OMPL) [6], the authors establish a common

Received 1 September 2025; accepted 28 December 2025. Date of publication 21 January 2026; date of current version 2 February 2026. This article was recommended for publication by Associate Editor Takuya Kiyokawa and Editor Olivier Stasse upon evaluation of the reviewers' comments. This work was supported by Inria through Program "Plan France Relance" in collaboration with the Aerospline Company (Plan de Relance AUCTUS - Aerospline Project).

(Corresponding author: Guillaume de Mathelin de Papigny.)

[1]Guillaume de Mathelin de Papigny is with the Automation and Control Institute, TU Wien, 1040 Vienna, Austria demathelin@acin.tuwien.ac.at

[2]Franco Gassibe is with Aerospline, 33520 Bordeaux, France rd.f.gassibe@aerospline.eu

[3]Vincent Padois is with Inria, Auctus team, 33400 Talence, France vincent.padois@inria.fr

framework [7] that incorporates three key methodologies for solving general constrained planning problems: the Projection Methodology [8], the Tangent Space Methodology [9] and the Atlas Methodology [10]. The Projection Methodology uses the Jacobian of the constraints for projecting a sample node of the configuration space to the constrained manifold. The other two methodologies use the Jacobian for computing tangent spaces to the constrained manifold and approximating it. Even if these common methodologies are well adapted in most cases due to the global search made by their sampling-based planning methods, they are not well-adapted for SCEEPP-problems. These methodologies need differentiable constraints. As elaborated in the subsequent sections, writing differentiable constraints in the case of SCEEPP-problems increases the dimension of the configuration space, and the Jacobian computation can only be done numerically. This approach leads to a large computational increase and is not reliable for producing solutions.

To tackle this limit, [11] and [12] propose a dual sampling-based planner strategy for solving SCEEPP-problems with redundant manipulators. The first planner, referred to as the "hard" or greedy task planner, initiates the process by planning along a nominal Cartesian path of the task, dealing with the redundancy of the robot without taking care of the task tolerances. For [12], this hard planner is implemented following the method presented in full task-constrained planning scenarios [13], [14]. Once this planner is blocked, a second "soft" task planner locally explores the tolerance axes around the nominal task-path pose at which the first planner stalled to bypass the obstacle. As local exploration operates only in the neighborhood of the blocked pose, this approach is effective when obstacles along the intended task path are widely spaced. However, it is not appropriate for finding solutions inside a confined or cluttered space.

Another approach, presented in [15], formulates a SCEEPP-problem as an objective function to minimize. This method is extremely interesting in an online motion planning context due to its fast computations. Nevertheless, it suffers from local minima and its success depends on many non-trivial hyper-parameters.

A brute-force algorithm to SCEEPP-problems, named Descartes, is introduced in [16] and in [17]. However, this approach is strongly impacted by the dimensionality of the problem. It is worth noting that two-step solutions, such as those in [18], [3], [19], which involve first constructing a graph with a significant number of valid joint nodes and edges before applying a graph-search method, are quite similar to a brute-force approach and also suffer from the dimensional curse.

Recently, methods utilizing a neural representation of the constrained manifold have been proposed to solve Constrained Planning problems [20], [21], [22], [23]. These methods learn a representation of the constrained manifold, enabling efficient planning and motion generation. However, their success is highly dependent on the accuracy and quality of the training process, which is challenging to achieve.

In this paper, we propose a novel approach, the F-space approach, that explores a space named the "tolerance space" using a sampling-based planner. In this space, dimensions represent tolerances on the possible end-effector motion, while one dimension represents the path parametrization of the Cartesian path. This exploration inherently induces an exploration of the constrained manifold. Our approach can be classified under the Reparameterization Methodology as outlined in [5]. Reparameterization indirectly sets the exploration graph of the configuration space via an exploration graph built by any sampling-based planner inside an alternate space. For example, [24] finds a free-collision path in the configuration space by exploring a reachable space made with a Minkowski sum space. In [25], constrained bimanual planning with two 7-DOF manipulator arms is addressed by exploring a 7+1 DOF space, denoting the joint space of one robot combined with the null-space dimension of the other. Note that [26] introduced a concept similar to our method by considering the task space as an alternate space. However, at each edge extension in task space, the resulting end-effector motion may fall outside its valid motion range and must be projected back. In our alternate space, the end-effector motion is inherently confined to the valid range, eliminating the need for any projection or reconfiguration.

Note that this paper focuses on the SCEEPP problem, where the end-effector must follow a predefined Cartesian path with specific constrained degrees of freedom; this implies exploring a constrained manifold. Other topics such as unconstrained end-effector motion planning methods (e.g., [27])—which implicitly assume any collision-free joint configuration results in an authorized end-effector pose—are beyond the scope of this work.

III. Formulation of a SCEEPP-problem

A. Description of the Environment

Throughout the task execution, a n_q-DOF manipulator arm, characterized by its configuration $\mathbf{q} \in \mathbb{R}^{n_q}$, must place its end-effector along the task path. The pose of the end-effector is defined by the forward kinematics function $f(\mathbf{q}) \in \mathrm{SE3}$.

As illustrated in Fig. 2a, a task is firstly defined as a Cartesian Path $\mathcal{T}(\sigma)$ to be followed in SE3, where $\sigma \in [0,1]$ represents the path parameterization. Then, if the task is not fully constrained, navigation along various freedom axes or tolerances is possible. These tolerances, as shown in Fig. 2a, can be represented as relative motions from a nominal task pose $\mathcal{T}(\sigma)$. Generally, these relative motions can be described as combinations of classical Euler angle rotations and/or linear translations. Consequently, the tolerances can be fully defined as a set of n ranges $[\delta_{i,\min}, \delta_{i,\max}] \subset \mathbb{R}$, where n is the number of tolerances. Each range denotes possible rotation angles or distances along the axes of the Cartesian path. The relative motions associated to the tolerances are denoted $T(\delta) \in \mathrm{SE3}$ where δ is an element of the set of tolerances $\Delta = \prod_{i=1}^{n} [\delta_{i,\min}, \delta_{i,\max}]$. Therefore, during the task execution, for every configuration $\mathbf{q}$ taken by the manipulator arm, there should exist $\delta \in \Delta$ and $\sigma \in [0,1]$ such as:

$$f(\mathbf{q}) = \mathcal{T}(\sigma) T(\delta) \tag{1}$$

Fig. 2b illustrates an example of Δ for a blasting task. It is

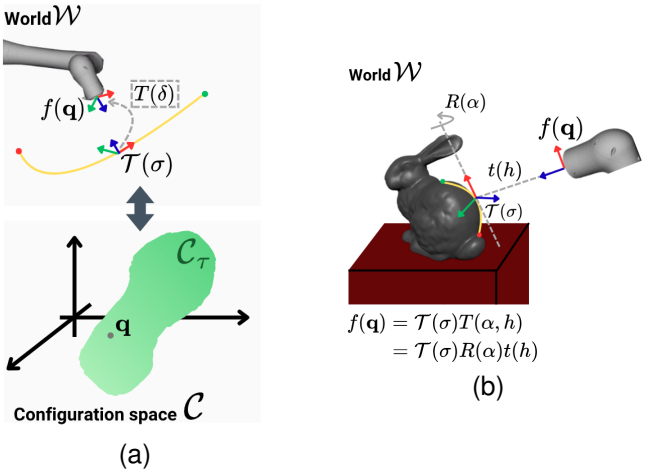

Fig. 2. (**a**) The end-effector pose is described as a combination of a nominal task pose $\mathcal{T}(\sigma)$ and a relative pose, or tolerance, $T(\delta)$. When the end-effector is in a valid pose, the joint configuration $\mathbf{q}$ lies in the constrained manifold $\mathcal{C}_\mathcal{T}$, and vice versa, as in Eq. (4). (**b**) Blasting task application, where the relative pose includes rotation around the path axis $R(\alpha)$ and translation along the end-effector axis $t(h)$.

characterized by two key tolerances:

- The incidence angle α of the end-effector, ranging from α_{min} to α_{max}.
- The distance h between the Cartesian path and the end-effector, ranging from h_{min} to h_{max}.

Given these parameters, we can define the set of tolerances Δ as $\Delta = [h_{\text{min}}, h_{\text{max}}] \times [\alpha_{\text{min}}, \alpha_{\text{max}}]$ where the relative motion $T(\delta) = T(h, \alpha) = t(h)R(\alpha) \in \text{SE3}$ is the combination of a rotation $R(\alpha) \in \text{SE3}$ along the tangential axis of the Cartesian path and a translation $t(h) \in \text{SE3}$ along the z-axis of the end-effector.

B. General Approach

The general sampling-based framework for constrained manifold path planning problems, as proposed in [7], is suboptimal for a SCEEPP-problem. Denoting $\mathcal{C}$ and $\mathcal{C}_{\text{obs}}$ as the configuration space and the obstacle configuration space, the objective is to find a continuous function $\mathcal{P}$ in the robot's configuration space $\mathcal{C}$, here, denoting the joint space $\mathbb{R}^{n_q}$, such that the path's start and end points correspond to the start joint configuration $\mathbf{q}_{\text{init}}$ and a joint configuration within the goal region $\mathcal{X}_{\text{end}}$, i.e.:

$$\mathcal{P}(0) = \mathbf{q}_{\text{init}} \text{ and } \mathcal{P}(1) \in \mathcal{X}_{\text{end}} \tag{2}$$

where $\mathcal{P}$ is free of collision ($\forall s \in [0,1] \quad \mathcal{P}(s) \in \mathcal{C} \backslash \mathcal{C}_{\text{obs}}$) and belongs to a constrained manifold $\mathcal{C}_\mathcal{T} = \{\mathbf{q} \in \mathcal{C} |\ F(\mathbf{q}) = \mathbf{0}\}$. F is known as the constraint function mapping into $\mathbb{R}^k$ ($k \in \mathbb{N}$); its explicit form for SCEEPP problems will be specified later in this section.

To successfully solve a SCEEPP problem, the end-effector must follow the Cartesian path from $\sigma = 0$ to $\sigma = 1$ while remaining within the allowed tolerance region Δ. Hence, with Eq. (4), the goal region $\mathcal{X}_{\text{end}}$ is defined as:

$$\mathcal{X}_{\text{end}} = \left\{\mathbf{q} \in \mathcal{C} \setminus \mathcal{C}_{\text{obs}} \;\middle|\; \exists\, \delta \in \Delta,\ f(\mathbf{q}) = \mathcal{T}(1)\, T(\delta)\right\}. \tag{3}$$

In classical sampling-based solutions, a planner typically explores the configuration space $\mathcal{C}$ by sampling and simultaneously filtering the samples to retain only the valid ones. However, in the case of constrained manifold path planning problems, the planners should directly explore the constrained manifolds $\mathcal{C}_\mathcal{T} \subset \mathcal{C}$ and avoid sampling mainly outside. For this purpose, the general framework proposes three methodologies: the Projection, the Tangent Space, and the Atlas methodologies. They are based on two main approaches: a projection approach, as in [8], and a tangent space approach, as in [9] and [10]. Both approaches essentially work by differentiating the function F. In the projection approach, each sample outside the constrained manifold is projected onto it. The direction of the projection is given by the Jacobian of F. For the tangent space approach, tangent spaces are created with the Jacobian of different samples belonging to the constrained manifolds for approximation purposes. Then, all steps inside the tangent spaces are projected onto the constrained manifolds with the function F and its Jacobian.

To define the constrained manifold and its associated function F in the case of SCEEPP-problems, we start with Eq. (1). As illustrated in Fig. 2a, Eq. (1) can be expressed as:

$$\mathbf{q} \in \mathcal{C}_\mathcal{T} \iff \exists \delta \in \Delta,\ \exists \sigma \in [0,1]\ \ f(\mathbf{q}) = \mathcal{T}(\sigma)T(\delta) \tag{4}$$

From Eq. (4), only two equality function candidates $F(\mathbf{q})$ are available. The first equality function is suggested by [28] and can be interpreted as follows: A configuration $\mathbf{q} \in \mathcal{C}$ lies on the constrained manifold if and only if the closest valid task-space pose to the end-effector pose $f(\mathbf{q}) \in \text{SE3}$ is exactly $f(\mathbf{q})$ itself. Mathematically, it is equivalent to:

$$\mathbf{q} \in \mathcal{C}, \quad F(\mathbf{q}) = \min_{\delta,\sigma} ||d\Big(f(\mathbf{q}), \mathcal{T}(\sigma)T(\delta)\Big)|| \tag{5}$$

where the operator d is the logarithm map from the Lie algebra [29], defined as $d(T_1, T_2) = \log\left(T_1^{-1}T_2\right) \in \mathbb{R}^6$, and $||.||$ is the norm operator.

Consequently, Eq. (5) defines the first constrained function candidate for SCEEPP-problems. When $F(\mathbf{q}) = \mathbf{0}$, it implies that the end-effector lies in a valid task pose and vice versa. However, due to the min term, the equality function Eq. (5) is not differentiable, which is a critical problem for applying the common sampling-based framework presented in [7].

One possible way to transform this equality function into a differentiable constraint would be to increase the dimension of the configuration space $\mathcal{C}$ by adding one axis for the path parametrization σ and several axes for the tolerances Δ inside the configuration space. Instead of finding only robot configurations $\mathbf{q} \in \mathcal{C}$ whose end-effector pose lies on the valid task poses set (as defined in Eq. (5)), the new equality constraint would find the different triplets $(\mathbf{q}, \sigma, \delta)$ from the augmented configuration space $\tilde{\mathcal{C}}$, where the end-effector pose $f(\mathbf{q})$ and the task poses $\mathcal{T}(\sigma)T(\delta)$ match together. With the new augmented configuration space $\tilde{\mathcal{C}}$, the constraint function F would become:

$$(\mathbf{q}, \sigma, \delta) \in \tilde{\mathcal{C}}, \quad F(\mathbf{q}, \sigma, \delta) = d\Big(f(\mathbf{q}), \mathcal{T}(\sigma)T(\delta)\Big) \tag{6}$$

Even though the constraint function is now differentiable, it increases the size of the configuration space with numerous

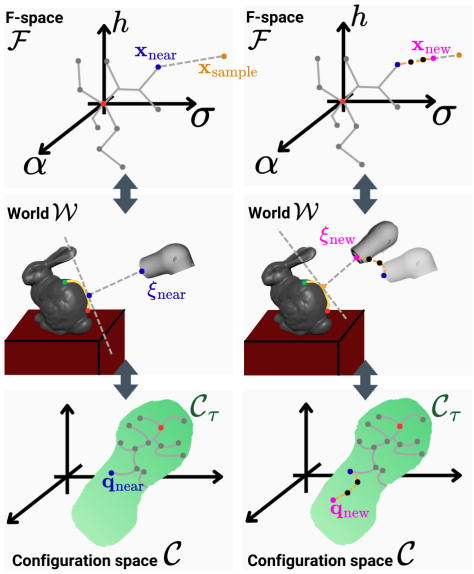

Fig. 3. Connection between the tolerance space $\mathcal{F}$, the world $\mathcal{W}$ and the configuration space $\mathcal{C}$. After finding from the graph G the closest node $\mathbf{x}_{\text{near}}$, with the associated end-effector pose ξ_{near} and joint configuration $\mathbf{q}_{\text{near}}$ (**Left**), a step is generated towards the sampling node $\mathbf{x}_{\text{sample}}$ to reach the new node $\mathbf{x}_{\text{new}}$ with the new end-effector pose ξ_{new} and joint configuration $\mathbf{q}_{\text{new}}$ (**Right**).

irrelevant triplets.

Consequently, Eq. (5) and Eq. (6) are not efficient for modeling the SCEEPP-manifolds.

IV. Reparameterization With the F-Space

This section introduces a novel reparameterization approach for solving SCEEPP-problems. The reparameterization methodology consists of indirectly exploring the manifold $\mathcal{C}_\mathcal{T}$ of the configuration space $\mathcal{C}$ by applying a sampling-based planner on an alternate space. In our case, the alternate space is called the tolerance space $\mathcal{F}$ (or F-space). As illustrated in Fig. 3, the F-space is defined as the Cartesian product of the path parameterization σ and the tolerances Δ, i.e., $\mathcal{F} = [0,1] \times \Delta$. Each element in the manifold $\mathcal{C}_\mathcal{T}$ corresponds to an element in the F-space. More specifically, as presented in Eq. (1), an element $\mathbf{q}$ within $\mathcal{C}_\mathcal{T}$ defines an end-effector pose $f(\mathbf{q}) = \mathcal{T}(\sigma)T(\delta)$ where $(\sigma, \delta) \in \mathcal{F}$. Then, exploring the set $[0,1] \times \mathbb{R}^n$ inherently explores the manifold $\mathcal{C}_\mathcal{T}$ and solves the associated SCEEPP-problem.

For illustration purposes, we adapt the well-known RRT planner for the exploration of the F-space to connect the nodes $\mathbf{x}_{\text{init}}$ and $\mathbf{x}_{\text{end}}$ with a graph $G(V, E)$ where V and E are the set of nodes and edges, respectively. As described in Alg. 1 (based on [30]), at each iteration, the RRT planner starts by sampling a node $\mathbf{x}_{\text{rand}}$ of the search space (`SampleFree`), then, generates a step (`Steer`) from the closest node $\mathbf{x}_{\text{near}}$ belonging to the initial graph G (found with `Nearest`), in direction of the sampling node, and finally, checks the validity of the step (`CheckValidRRT`) from $\mathbf{x}_{\text{near}}$ to the new node $\mathbf{x}_{\text{new}}$. If the step is valid, the edge and $\mathbf{x}_{\text{new}}$ are added to the graph G. To apply the planner in the F-space, different points need to be adapted. These adaptations, presented below, lead

Algorithm 1 `RRT`

1: **procedure** $(\mathbf{x}_{\text{init}}, \mathbf{x}_{\text{end}})$
2: $V \leftarrow \mathbf{x}_{\text{init}}$; $E \leftarrow \emptyset$; $\mathbf{x}_{\text{new}} \leftarrow \mathbf{x}_{\text{init}}$;
3: **while not** `CheckValidRRT`$(\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{end}})$ **do**
4: $\mathbf{x}_{\text{rand}} \leftarrow$ `SampleFree`$()$;
5: $\mathbf{x}_{\text{near}} \leftarrow$ `Nearest`$(V, E, \mathbf{x}_{\text{rand}})$;
6: $\mathbf{x}_{\text{new}} \leftarrow$ `Steer`$(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{rand}})$;
7: isValid $\leftarrow$ `CheckValidRRT`$(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{new}})$;
8: **if** isValid **then**
9: $V \leftarrow V \cup \{\mathbf{x}_{\text{new}}\}$;
10: $E \leftarrow E \cup \{(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{new}})\}$;
11: **return** $G = (V, E)$;

to a suitable RRT planner to operate in the F-space, named hereafter F-RRT and available in Alg. 2.

a) Bounds of the F-space: As explained previously, each iteration starts by sampling a node (`SampleFree`) inside the search space. In most path planning problems, the search space is the joint space $\mathbb{R}^{n_q}$ bounded by the kinematic limits. In a SCEEPP-problem, as explained in the previous section, the search space is the F-space $\mathcal{F}$. Thus, the function `SampleFree` samples a node $\mathbf{x}_{\text{rand}}$ inside the set $\mathcal{F} = [0,1] \times \prod_{i=1}^{n} [\delta_{i,\min}, \delta_{i,\max}] \subset \mathbb{R}^{n+1}$.

b) The Checking Function: After sampling $\mathbf{x}_{\text{rand}}$, the planner builds a new edge for the graph G in $\mathcal{F}$ by steering a small step from the nearest node $\mathbf{x}_{\text{near}}$ of G toward $\mathbf{x}_{\text{rand}}$, producing the new node $\mathbf{x}_{\text{new}}$ (`Steer`). For a RRT-planner, the planner checks if the step is collision-free at each point with a checking function. Depending on the result of the function, it is either added to the graph or deleted. For F-RRT, the checking algorithm `CheckValidFRRT` is available in Alg. 3. As illustrated in Fig. 3, the step ($\mathbf{x}_{\text{near}}$, $\mathbf{x}_{\text{new}}$) is discretized with a given resolution, producing the sequence $\{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(p)}\} = \{(\sigma^{(1)}, \delta^{(1)}), \ldots, (\sigma^{(p)}, \delta^{(p)})\}$. Each element $(\sigma^{(i)}, \delta^{(i)})$ corresponds, in the workspace SE3, to an end-effector pose obtained as $\xi^{(i)} = \mathcal{T}(\sigma^{(i)})\,T(\delta^{(i)})$, yielding the list of poses $\{\xi^{(1)}, \ldots, \xi^{(p)}\}$ (Alg. 3, line 3). The step ($\mathbf{x}_{\text{near}}$, $\mathbf{x}_{\text{new}}$) is considered valid if $\{\xi^{(1)}, \ldots, \xi^{(p)}\}$ are reachable with collision-free joint configurations $\{\mathbf{q}^{(1)}, \ldots, \mathbf{q}^{(p)}\}$. Starting from the initial pair $(\xi^{(1)}, \mathbf{q}_{\text{near}})$ (Alg. 3, line 4), the checking function iterates over the sequence $\{\xi^{(1)}, \ldots, \xi^{(p)}\}$ and, for each pose $\xi^{(i)}$, solves the constrained differential inverse-kinematics problem—formulated as a Quadratic Program (QP) [31]—to move the end-effector from ξ to $\xi^{(i)}$ (Alg. 3 line 6). The QP constraints enforce joint-motion continuity and joint-limit compliance to validate the reachability of each pose $\xi^{(i)}$, while the regularization term of the solver ensures robustness to singularities and solves the multiple inverse kinematics solutions problem for 6- and 7-DoF manipulator arms. If the pose $\xi^{(i)}$ is reachable with the given joint configuration $\mathbf{q}^{(i)}$, i.e., $f(\mathbf{q}^{(i)}) = \xi^{(i)}$ (Alg. 3 line 7), collision checking (Alg. 3 line 9) is performed on this configuration. If the configuration is valid, ξ and $\mathbf{q}$ are updated with the new values (Alg. 3 line 12), and the loop continues. At the end of the loop, if all poses $\xi^{(i)}$ are reachable and all configurations $\mathbf{q}^{(i)}$ are collision-free, the edge $(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{new}})$ is declared valid and added to the graph G. A sufficiently

Algorithm 2 `F-RRT`

1: **procedure** $(\mathbf{x}_{\text{init}}, \mathbf{x}_{\text{end}}, \mathbf{q}_{\text{init}})$
2: $V \leftarrow (\mathbf{x}_{\text{init}}, \mathbf{q}_{\text{init}})$; $E \leftarrow \emptyset$; $(\mathbf{x}_{\text{new}}, \mathbf{q}_{\text{new}}) \leftarrow (\mathbf{x}_{\text{init}}, \mathbf{q}_{\text{init}})$;
3: **while not** `CheckValidFRRT`$(\mathbf{x}_{\text{new}}, \mathbf{x}_{\text{end}}, \mathbf{q}_{\text{new}})$ **do**
4: $\mathbf{x}_{\text{rand}} \leftarrow$ `SampleFree`$()$;
5: $(\mathbf{x}_{\text{near}}, \mathbf{q}_{\text{near}}) \leftarrow$ `Nearest`$(V, E, \mathbf{x}_{\text{rand}})$;
6: $\mathbf{x}_{\text{new}} \leftarrow$ `Steer`$(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{rand}})$;
7: isValid, $\mathbf{q}_{\text{new}} \leftarrow$ `CheckValidFRRT`$(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{new}}, \mathbf{q}_{\text{near}})$;
8: **if** isValid **then**
9: $V \leftarrow V \cup \{(\mathbf{x}_{\text{new}}, \mathbf{q}_{\text{new}})\}$;
10: $E \leftarrow E \cup \{(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{new}})\}$;
11: **return** $G = (V, E)$;

Algorithm 3 `CheckValidFRRT`

1: **procedure** $(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{new}}, \mathbf{q}_{\text{near}})$
2: $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(p)}\} \leftarrow$ `Discretized`$(\mathbf{x}_{\text{near}}, \mathbf{x}_{\text{new}})$;
3: $\{\xi^{(1)}, \ldots, \xi^{(p)}\} \leftarrow$ `EEPose`$(\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(p)}\})$;
4: $\xi \leftarrow \xi^{(1)}$; $\mathbf{q} \leftarrow \mathbf{q}_{\text{near}}$;
5: **for** $i = 2, \ldots, p$ **do**
6: $\mathbf{q}^{(i)} \leftarrow$ `QP-IK`$(\xi^{(i)}, \xi, \mathbf{q})$;
7: **if** $f(\mathbf{q}^{(i)}) \neq \xi^{(i)}$ **then**
8: **return** False, $\mathbf{q}$;
9: isCollisionFree $\leftarrow$ `CheckCollision`$(\mathbf{q}^{(i)})$;
10: **if not** isCollisionFree **then**
11: **return** False, $\mathbf{q}$;
12: $\xi \leftarrow \xi^{(i)}$; $\mathbf{q} \leftarrow \mathbf{q}^{(i)}$
13: **return** True, $\mathbf{q}$;

fine resolution in `Discretized` function ensures that the whole joint motion is collision free and constraints compliant. For joint consistency, the joint configuration $\mathbf{q}_{\text{new}} = \mathbf{q}^{(p)}$ is saved as an attribute of the node $\mathbf{x}_{\text{new}}$ (Alg. 2 line 9). Hence, each step starting from $\mathbf{x}_{\text{new}}$ begins with its associated joint configuration $\mathbf{q}_{\text{new}}$. In our experiments, the Quadratic Programming solver is implemented with qpOASES [32], combined with Pinocchio [33], and collision detection is performed with MuJoCo [34].

c) Define the Starting and the Ending Regions: When defining the starting and ending regions in the F-space, the essential value is the term σ. A manipulator arm starting and finishing its task necessarily has its end-effector at the beginning of the Cartesian path, i.e., $\sigma = 0$, and at the end of the Cartesian path, i.e., $\sigma = 1$. As the tolerance values δ at these boundary points are optional—meaning the end-effector may reach the start or end of the path within any permissible tolerance—the sets of starting nodes and ending nodes are both hyperplanes of $\mathcal{F}$ and are respectively defined as follows $\mathcal{F}_{\text{init}} = \{\mathbf{x} = (\sigma, \delta) \mid \sigma = 0\}$ and $\mathcal{F}_{\text{end}} = \{\mathbf{x} = (\sigma, \delta) \mid \sigma = 1\}$. By definition of the goal region $\mathcal{X}_{\text{end}}$ in Eq. (3), the configuration $\mathbf{q}$ associated with any node $\mathbf{x} \in \mathcal{F}_{\text{end}}$ necessarily belongs to $\mathcal{X}_{\text{end}}$. Then, at each call of `CheckValidFRRT` in Alg. 2 at line 3 with $\mathbf{x}_{\text{new}} = (\sigma_{\text{new}}, \delta_{\text{new}})$, we can set $\mathbf{x}_{\text{end}} = (1, \delta_{\text{new}})$, thereby accelerating the algorithm's termination.

One interest of this adaptation is that it retains a planner-level hyperparameter count similar to RRT, limited primarily to step size and resolution. The additional IK-solver parameters (e.g., regularization weight, joint limits) are robot-specific and fixed across tasks, eliminating per-task tuning. Furthermore, in terms of complexity dimensionality, task tolerances are low-dimensional (typically 1–3, and at most 6 to span SE3), so the resulting search space remains modest even when considering a broad range of tolerances.

V. Experiments

To evaluate the performance of our F-space approach, we conducted three distinct experiments: (i) a benchmark against standard OMPL methods, (ii) an evaluation in complex and cluttered environments, and (iii) trajectory post-processing stages in a practical industrial scenario. All simulations were performed on a computer equipped with an AMD Ryzen Threadripper PRO 3995WX CPU, and all planners were implemented in C++. A video presenting the results is available in the media attachment or at www.acin.tuwien.ac.at/d0bb.

1) Benchmark Against Existing Approaches: We benchmark our method, F-RRT, against the RRT-planners combined with the three standard constrained approaches implemented in OMPL: the Projection-space approach (PB-RRT), the Atlas approach (Atlas-RRT), and the Tangent space approach (TS-RRT). Three SCEEPP scenarios were designed for this comparison: UR-1, UR-2, and PANDA. Illustrations of the scenarios are provided in Fig. 4. In the first two scenarios, a UR10e robot's end-effector is free to rotate only around the path axis. The difference between the two scenarios lies in the construction of the Cartesian path: circular in the first case and interpolated in the second. The last scenario employs a Franka Panda robot, which end-effector can move along its z-axis and rotate around the path axis. Each method was executed 25 times per scenario. Simulations were terminated if no solution was found within 10 minutes. Table I reports the computation times (median with 95% confidence intervals obtained via bootstrapping), the success probabilities (mean with 95% confidence intervals based on the binomial method), and the conditioning value κ_F of the Jacobian of the constraint function F (Eq. (6)) at the initial configuration. Across all scenarios, F-RRT demonstrates the best performance: median computation times remain consistently below 1 s, and the method achieves a 100% success rate in every case. Among the baseline methods, when the constraint manifold is well-conditioned (UR-1 and PANDA), both TS-RRT and Atlas-RRT achieve near-perfect success rates (above 95%), whereas PB-RRT succeeds in fewer than 25% of the runs. In the poorly conditioned scenario (UR-2), all three standard projection-based methods fail, given their reliance on Jacobian-based iterative projection applied to the Jacobian of F [7]. In contrast, our planner remains unaffected, as it samples directly in the constrained manifold thanks to the F-space approach.

Furthermore, as the other methods require a predefined end joint configuration in the configuration space defined in Eq. (6), users are forced to select a suitable end configuration for each environment, which can be inconvenient. By exploring the F-space instead, our method overcomes this limitation, offering a more flexible and user-friendly solution.

TABLE I
RESULTS FOR DIFFERENT METHODS ACROSS SCEEPP SCENARIOS.

Method	UR-1 [$\kappa_F = 10.76$]		UR-2 [$\kappa_F = 82321$]		PANDA [$\kappa_F = 5.67$]	
	Time (s)	Succ.	Time (s)	Succ.	Time (s)	Succ.
F-RRT (ours)	**0.5 [0.47-0.55]**	**1.00 [0.86-1.0]**	**0.2 [0.14-0.22]**	**1.00 [0.86-1.0]**	**0.3 [0.27-0.34]**	**1.00 [0.86-1.0]**
PB-RRT	351.5 [297.07-405.96]	0.08 [0.01-0.26]	N/A	0.00 [0.0-0.14]	547.4 [480.43-584.45]	0.16 [0.05-0.36]
Atlas-RRT	1.9 [1.75-8.59]	1.00 [0.86-1.0]	N/A	0.00 [0.0-0.14]	15.6 [14.37-19.73]	1.00 [0.86-1.0]
TS-RRT	0.8 [0.81-7.76]	1.00 [0.86-1.0]	N/A	0.00 [0.0-0.14]	7.9 [7.38-25.54]	0.96 [0.8-1.0]

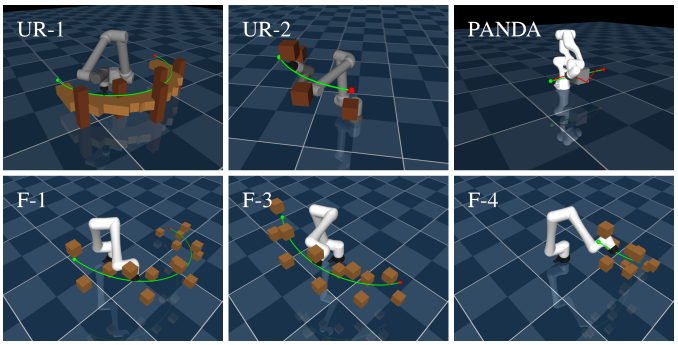

Fig. 4. **Top row**: three SCEEPP setups involving a UR10e robot, free to rotate around the path axis, with circular (UR-1) and interpolated (UR-2) Cartesian paths, and a Franka Panda robot, with rotational freedom around the path axis and translational freedom along its z-axis (PANDA). **Bottom row**: three of the five cluttered FANUC CRX-10iA/L environments (F-1, F-3, F-4), with full rotational freedom, allowing any orientation. The Cartesian path of the task is represented in green with its starting node in green and ending node in red.

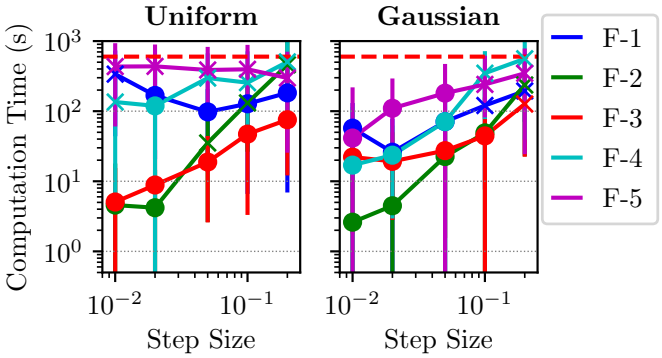

Fig. 5. Computation times for the cluttered environments under different step sizes and sampling schemes in σ. **Left**: Uniform sampling. **Right**: Gaussian sampling. A circular knot indicates that the success ratio is above 90% across runs; the error bars show the 95% confidence intervals. The red dashed line denotes the computation time limit.

2) Evaluation in Cluttered Environments: To further assess the robustness of our method, we designed a family of five cluttered environments involving a FANUC CRX-10iA/L robot and a dense arrangement of obstacles forming a spiral around the Cartesian path. Three of these environments are shown in Fig. 4. In all cases, the end-effector is allowed to rotate freely around Euler angles (α, β, γ). The spiral configuration forces the planner to continuously adjust the feasible orientations to avoid collisions, thereby providing a challenging test of F-RRT's exploration capabilities.

In these environments, we study the influence of the step size on planning performance, and whether using a Gaussian distribution (mean 1, standard deviation 0.3) instead of a uniform distribution for sampling along the σ axis in `SampleFree` (Alg. 2, line 4) improves the planner. As before, each method is executed 25 times per scenario and per step size, and a run is considered a failure if no solution is found within 10 minutes.

Fig. 5 reports the computation times (median with 95% bootstrap confidence intervals) for the different step sizes. Globally, except in scenario F-1, the smallest step size (0.01) performs best across both sampling distributions. This is consistent with the expectation that navigating narrow corridors benefits from small step progress in F-space. Moreover, at this step size (0.01), although uniform sampling performs better in scenario F-3, the Gaussian distribution yields greater improvements in the other scenarios, with all medians remaining below 100 s and with a success ratio always about 90% (except for F-4). This effect comes from the need to progress toward $\sigma = 1$: once $\sigma > 0.5$, Gaussian sampling centered at 1 offers a higher chance of advancing than uniform sampling. Additionally, because the planners must continuously update the tolerance, they cannot benefit from the algorithm's termination acceleration described in Section IV.c.

Large confidence intervals for computation time are observed in both distributions, mainly due to the exploratory nature of RRT-based methods. Early random samples can bias initial tree growth toward suboptimal regions, causing the planner to spend time in less promising areas before expanding to the rest of the space. The time required to escape these initial basins varies considerably from run to run, which explains the high variance.

In summary, F-RRT with a small step size (0.01) and a Gaussian sampling distribution over σ is best suited for planning in cluttered environments.

3) Real-World Execution and Post-Processing: To demonstrate the applicability of our F-RRT method in a real scenario, we planned a path for a KUKA LBR IIWA 14 robot tasked with drawing a circle on a board located inside a confined structure. Due to joint limits and the restricted workspace, using a direct task-space controller would result in collisions with the structure. To address this challenge, we allowed the robot's end-effector to rotate both around the path axis and the z-axis, providing greater flexibility. The resulting path is then refined by removing nodes whose deletion does not introduce collisions; this removal step also helps to maintain monotonicity along σ. For the dynamic trajectory generation,

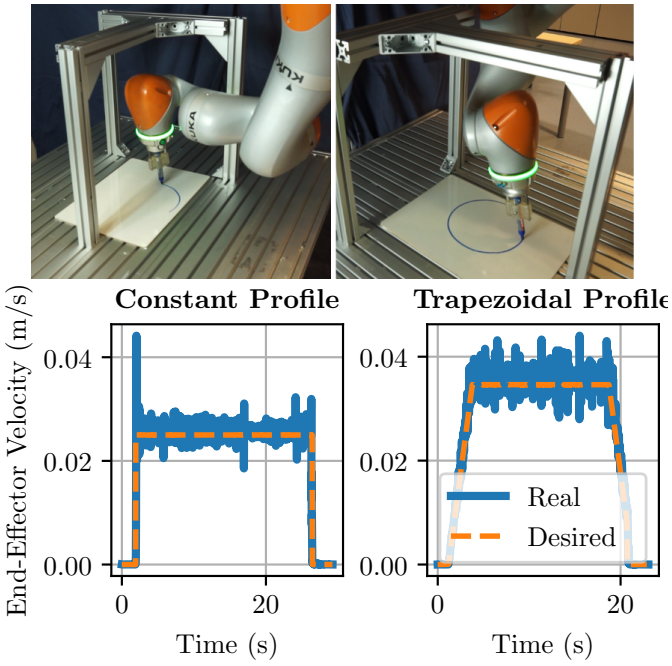

Fig. 6. Application of the F-RRT Method to a drawing task. A KUKA LBR IIWA 14 robot is controlled to draw a circle while being free to rotate around the z-axis and the path axis (**top**). Post-processing of the resulting solution allows enforcing dynamic constraints such as constant or trapezoidal end-effector velocity profiles (**bottom**).

as with any path planning result, our method's generated path can be parameterized in multiple ways, including a time-optimal trajectory using TOPP-RA [35] and alternative trajectories that satisfy dynamic constraints, such as constant or trapezoidal end-effector velocity profiles. Photographs of the experimental setup and the evolution of the end-effector velocity for different profile constraints are shown in Fig. 6. The results confirm that the solution computed by our approach can be effectively executed in the real world with the assistance of path parametrization algorithms.

Overall, our experiments demonstrate that the F-RRT method—and, more generally, the F-space approach—offers a promising solution to the SCEEPP problem. Our approach outperforms existing methods in terms of computation time and success rate. In addition, the results show that our method can handle both redundant robots and cluttered environments, making it a versatile and robust tool for a wide range of robotic applications.

VI. Discussion

The F-space approach seems perfectly fitted for SCEEPP-problems. Yet, several future research directions can be explored to extend its potential.

A key benefit of introducing the F-space is that it enables the direct transfer of standard sampling-based planning tools from configuration space to this alternative representation. For instance, non-fixed tolerances $\Delta(\sigma)$ along the nominal path can be represented as convex subsets of the F-space and sampled using classical techniques such as rejection sampling or projection operators. However, this direct transfer is not for all planners. Due to the multiplicity of the inverse kinematics solutions, planners that compute isolated nodes in their process, such as the BIT* planner [36], cannot yet be considered for the exploration of the F-space.

Currently, the F-RRT adaptation looses the property of probabilistic completeness. This is because the inverse kinematics method used in the reparameterization process does not fully map the F-space to the entire configuration space. As a result, the reparameterization might not always intersect with the goal region ($\mathcal{X}_{\text{end}}$) of the configuration space, even if a solution exists. Nevertheless, the goal region in F-space, $\mathcal{F}_{\text{end}}$, is a hyperplane rather than a single point (see Section IV.c), which increases the probability of discovering valid solutions in practice. The coverage can also be expanded through practical strategies, including initiating F-RRT from diverse initial joint configurations or encoding the null-space of 7-DoF manipulators along an axis of $\mathcal{F}$.

Our approach represents the task path as a linear axis σ in F-space, enabling any interpolated path. This extends to higher dimensions—e.g., constraining the end-effector to a surface via two F-space axes using NURBS/T-spline parameterizations [37], [38].

VII. Conclusion

In this paper, we presented a new reparameterization approach for solving SCEEPP-problems. From this approach, we constructed an adapted planner of RRT, named F-RRT, which provides a faster and more robust alternative to traditional solutions while maintaining a limited number of hyperparameters for design. By mapping the tolerances to a product of intervals, this approach effectively addresses complex and obstructed task environments. Many further topics can be investigated. One possibility is to consider the redundancy of manipulator arms as an F-space axis. Another area of investigation could involve enhancing the inverse kinematics method to achieve probabilistic completeness. Our approach could also be extended to different sampling-based planners or could be applied to any surface constraints.

Acknowledgment

This work has been funded by the Plan de Relance AUCTUS - Aerospline project.

References

[1] S. M. LaValle and J. J. Kuffner, "Randomized Kinodynamic Planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[2] L. E. Kavraki and S. M. LaValle, "Motion planning," in *Springer handbook of robotics*. Springer, 2016, pp. 139–162.

[3] R. K. Malhan, S. Thakar, A. M. Kabir, P. Rajendran, P. M. Bhatt, and S. K. Gupta, "Generation of Configuration Space Trajectories Over Semi-Constrained Cartesian Paths for Robotic Manipulators," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 1, pp. 193–205, 2022.

[4] Y. Wang and M. Gleicher, "Anytime Planning for End-Effector Trajectory Tracking," *IEEE Robotics and Automation Letters*, vol. 10, no. 4, pp. 3246–3253, 2025.

[5] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-Based Methods for Motion Planning with Constraints," *Annual review of Control, Robotics, and Autonomous Systems*, vol. 1, pp. 159–185, 2018.

[6] I. A. Sucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.

[7] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *The International Journal of Robotics Research*, vol. 38, no. 10-11, pp. 1151–1178, 2019.

[8] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner, "Manipulation Planning on Constraint Manifolds," in *Proc. IEEE International Conference on Robotics and Automation*, 2009, pp. 625–632.

[9] B. Kim, T. T. Um, C. Suh, and F. C. Park, "Tangent bundle RRT: A randomized algorithm for constrained motion planning," *Robotica*, vol. 34, no. 1, pp. 202–225, 2016.

[10] L. Jaillet and J. M. Porta, "Path Planning Under Kinematic Constraints by Rapidly Exploring Manifolds," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 105–117, 2012.

[11] Z. Yao and K. Gupta, "Self-motion graph in path planning for redundant robots along specified end-effector paths," in *Proc. IEEE International Conference on Robotics and Automation*, 2006, pp. 2004–2009.

[12] M. Cefalo, P. Ferrari, and G. Oriolo, "An Opportunistic Strategy for Motion Planning in the Presence of Soft Task Constraints," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6294–6301, 2020.

[13] M. Cefalo, G. Oriolo, and M. Vendittelli, "Task-constrained motion planning with moving obstacles," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, pp. 5758–5763.

[14] G. Oriolo, M. Cefalo, and M. Vendittelli, "Repeatable Motion Planning for Redundant Robots Over Cyclic Tasks," *IEEE Transactions on Robotics*, vol. 33, no. 5, pp. 1170–1183, 2017.

[15] Y. Wang, P. Praveena, D. Rakita, and M. Gleicher, "RangedIK: An Optimization-based Robot Motion Generation Method for Ranged-Goal Tasks," in *Proc. International Conference on Robotics and Automation*, 2023, pp. 9700–9706.

[16] J. De Maeyer, B. Moyaers, and E. Demeester, "Cartesian path planning for arc welding robots: Evaluation of the descartes algorithm," in *Proc. IEEE International Conference on Emerging Technologies and Factory Automation*, 2017, pp. 1–8.

[17] J. De Maeyer, M. Versteyhe, and E. Demeester, "Sampling-based Tube Following for Redundant, Planar Robotic Manipulators," in *Proc. IEEE International Conference on Emerging Technologies and Factory Automation*, vol. 1, 2018, pp. 752–758.

[18] Y. Sun, X. Meng, M. Yu, and H. Tang, "Semi-constrained Path Planning for Industrial Robot with External Axis," in *Proc. Chinese Automation Congress*, 2020, pp. 6180–6185.

[19] T. Weingartshofer, B. Bischof, M. Meiringer, C. Hartl-Nesic, and A. Kugi, "Optimization-based path planning framework for industrial manufacturing processes with complex continuous paths," *Robotics and Computer-Integrated Manufacturing*, vol. 82, p. 102516, 2023.

[20] A. H. Qureshi, J. Dong, A. Baig, and M. C. Yip, "Constrained Motion Planning Networks X," *IEEE Transactions on Robotics*, vol. 38, no. 2, pp. 868–886, 2021.

[21] P. Kicki, *et al.*, "Fast Kinodynamic Planning on the Constraint Manifold with Deep Neural Networks," *IEEE Transactions on Robotics*, vol. 40, pp. 277–297, 2024.

[22] R. Ni and A. H. Qureshi, "Physics-informed Neural Motion Planning on Constraint Manifolds," in *Proc. IEEE International Conference on Robotics and Automation*, 2024, pp. 12 179–12 185.

[23] P. Liu, H. Bou-Ammar, J. Peters, and D. Tateo, "Safe Reinforcement Learning on the Constraint Manifold: Theory and Applications," *IEEE Transactions on Robotics*, vol. 41, pp. 3442–3461, 2025.

[24] T. McMahon, S. Thomas, and N. M. Amato, "Sampling-based motion planning with reachable volumes for high-degree-of-freedom manipulators," *The International Journal of Robotics Research*, vol. 37, no. 7, pp. 779–817, 2018.

[25] T. Cohn, S. Shaw, M. Simchowitz, and R. Tedrake, "Constrained Bimanual Planning with Analytic Inverse Kinematics," in *Proc. IEEE International Conference on Robotics and Automation*, 2024, pp. 6935–6942.

[26] Z. Yao and K. Gupta, "Path planning with general end-effector constraints: Using task space to guide configuration space search," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005, pp. 1875–1880.

[27] A. Kimmel, R. Shome, and K. Bekris, "Anytime motion planning for prehensile manipulation in dense clutter," *Advanced robotics*, vol. 33, no. 22, pp. 1175–1193, 2019.

[28] M. Stilman, "Global Manipulation Planning in Robot Joint Space with Task Constraints," *IEEE Transactions on Robotics*, vol. 26, no. 3, pp. 576–584, 2010.

[29] K. M. Lynch and F. C. Park, *Rigid Body Motions and Twists*. Cambridge University Press, 2017, ch. 3, pp. 57–134. [Online]. Available: http://modernrobotics.org

[30] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *The International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, 2011.

[31] A. Escande, N. Mansard, and P.-B. Wieber, "Fast resolution of hierarchized inverse kinematics with inequality constraints," in *Proc. IEEE International Conference on Robotics and Automation*, 2010, pp. 3733–3738.

[32] H. J. Ferreau, C. Kirches, A. Potschka, H. G. Bock, and M. Diehl, "qpOASES: a parametric active-set algorithm for quadratic programming," *Mathematical Programming Computation*, vol. 6, pp. 327–363, 2014.

[33] J. Carpentier, *et al.*, "The Pinocchio C++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *Proc. IEEE/SICE International Symposium on System Integration*, 2019, pp. 614–619.

[34] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A physics engine for model-based control," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 5026–5033.

[35] H. Pham and Q.-C. Pham, "A New Approach to Time-Optimal Path Parameterization Based on Reachability Analysis," *IEEE Transactions on Robotics*, vol. 34, no. 3, pp. 645–659, 2018.

[36] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Batch Informed Trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in *Proc. IEEE International Conference on Robotics and Automation*, 2015, pp. 3067–3074.

[37] G. Farin, "From conics to NURBS: A tutorial and survey," *IEEE Computer Graphics and applications*, vol. 12, no. 5, pp. 78–86, 1992.

[38] T. W. Sederberg, D. L. Cardon, G. T. Finnigan, N. S. North, J. Zheng, and T. Lyche, "T-spline simplification and local refinement," *ACM Transactions On Graphics*, vol. 23, no. 3, pp. 276–283, 2004.