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

Guillaume de Mathelin de Papigny, Francesco Gassibe, Vincent Padois

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

Guillaume de Mathelin[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.**

I. INTRODUCTION

While advances in robotics have significantly improved path planning techniques [1], 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 problem (SCEEPP).

The challenge in solving SCEEPP-problems is effectively exploring the robot's valid configuration space, or constrained manifold, defined by the end-effector's constraints. Generally, sampling-based methods, due to their simplicity of use and tuning, are privileged for any kind of path planning problem with a manipulator arm. Furthermore, a common framework exists to consider tasks constraints in path planning problems and their induced constrained manifolds (see [2]). It relies on a differentiation of the constraints to explore the constrained manifolds. However, this framework shows its limits in the face of SCEEPP-problems where the differentiation of the constraints fails to give solutions.

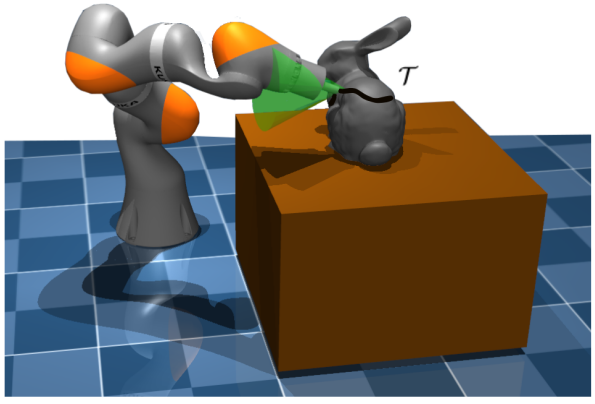


Fig. 1. The gluing task is a classic semi-constrained end-effector task where the end-effector is semi-constrained in orientation but is fully position constrained. The tip of the end-effector must follow the defined Cartesian path $\mathcal{T}$ inside an orientation cone.

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 hyper-parameters 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 and preserves the simplicity of sampling-based methods. 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 statement. Then, the new 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.

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 [3], where different methodologies are categorized. However, this review does not explore

This work was supported by Inria through the program "Plan France Relance" in collaboration with the Aerospline company

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

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

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

the SCEEPP-problems in details. From [3], in the Open Motion Planning Library [4], the authors establish a common framework [2] that incorporates three key methodologies for solving general constrained planning problems: the Projection Methodology [5], the Tangent Space Methodology [6] and the Atlas Methodology [7]. 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 of the cases due to the global research made by their sampling-based planning methods, they are not well-adapted for SCEEPP-problems. These methodologies need differentiable constraints and their methods become much slower when the Jacobian of constraints is numerically calculated rather than analytically derived. As it is more detailed in the next 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 lengthy computations and it is not reliable for producing solutions.

[8] and [9] propose a dual sampling-based planner strategy for solving SCEEPP-problems with redundant manipulators. The first planner, referred 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. Then, once the first planner is blocked, a second "soft" task planner locally explores the tolerances axes to bypass the obstacle. 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 [10] and [11], 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. But, it suffers from local minima and its success depends on many non-trivial hyperparameters.

A brute-force algorithm to SCEEPP-problems, named Descartes, is introduced in [12] and in [13]. However, this approach is strongly impacted by the dimensionality. It is worth noting that two-step solutions, such as those in [14] and [15], 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.

In this paper, we propose a novel 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 task. This exploration inherently induces an exploration of the constrained manifold. Our approach can be classified under the Reparametrization Methodology as outlined in [3]. Reparametrization 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, [16] finds a free-collision path in the configuration space by exploring a reachable space made with a Minkowski sum space. Note that [17] introduced a concept similar to our method by considering an alternate space associated to the end-effector (the task space). However, with our proposed alternate space, the extension of an edge always fits on the constrained manifold, eliminating the need for any projection or reconfiguration.

It is shown further than our reparametrization approach effectively solves complex SCEEPP-problems using traditional sampling-based planners within reasonable time frames. This approach achieves these results without increasing the dimensionality of the problem and in reliable way. Additionally, it avoids local minima and requires no more design hyperparameters than a standard sampling-based planner.

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. 2(a), 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. 2(a), 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 classical 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:

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

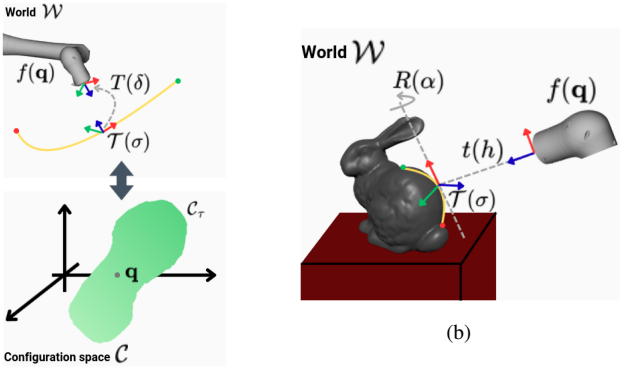
Fig. 2(b) illustrates an example of Δ for a blasting task. It is characterized by two key tolerances:

- The incidence angle α of the end-effector, ranging from $\alpha_{\min}$ to $\alpha_{\max}$.
- 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:

$$\Delta = [h_{\min}, h_{\max}] \times [\alpha_{\min}, \alpha_{\max}]$$

and the relative motion $T(\delta) = T(h, \alpha) \in \mathrm{SE3}$ is the combination of a rotation $R(\alpha) \in \mathrm{SE3}$ along the tangential axis of the Cartesian path and a translation $t(h) \in \mathrm{SE3}$ along the z-axis of the end-effector. Thus, during the task execution,



(a) (b)

Fig. 2. (a) A SCEEPP constrained manifold $\mathcal{C}_\mathcal{T}$ in the configuration space $\mathcal{C}$ denoting all valid joint configurations along the task. (b) The available tolerances for a blasting task.

for every configuration $\mathbf{q}$ taken by the manipulator arm, there should exist $\delta = (h, \alpha) \in \Delta$ and $\sigma \in [0, 1]$ such as:

$$f(\mathbf{q}) = T(\delta)\mathcal{T}(\sigma) = t(h)R(\alpha)\mathcal{T}(\sigma)$$

B. General Approach

The general sampling-based framework for constrained manifold path planning problems, as proposed in [2], is sub-optimal for a SCEEPP-problem. The concept of constrained manifold path planning problem is well-defined in the scientific literature. 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 $\mathcal{C}$ such that the path's start and end points correspond to the start node $\mathbf{x}_{\text{start}}$ and a node within the goal region $\mathcal{X}_{\text{goal}}$, i.e.:

$$\mathcal{P}(0) = \mathbf{x}_{\text{start}} \text{ and } \mathcal{P}(1) \in \mathcal{X}_{\text{goal}} \tag{2}$$

where the path is free of collision:

$$\forall s \in [0, 1] \quad \mathcal{P}(s) \in \mathcal{C} \backslash \mathcal{C}_{\text{obs}} \tag{3}$$

and where the path belongs to a constrained manifold $\mathcal{C}_\mathcal{T} \subset \mathcal{C}$ of the configuration space, defined as follows:

$$\mathbf{x} \in \mathcal{C}_\mathcal{T} \iff F(\mathbf{x}) = \mathbf{0} \tag{4}$$

where $F(\mathbf{x}) \in \mathbb{R}^k (k \in \mathbb{N})$ is known as the constraint function. Generally, in path planning problems, the configuration space $\mathcal{C}$ denotes the joint space $\mathbb{R}^{n_q}$ of the manipulator arm where each element $\mathbf{x} \in \mathcal{C}$ is a configuration $\mathbf{q}$ of $\mathbb{R}^{n_q}$. That is the notation retained in the following sections.

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 [5], and a tangent space approach, as in [6] and [7]. 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 the constrained function F. For the tangent space approach, tangent spaces are created with the Jacobian to 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 (1). As illustrated in Fig. 2(a), (1) can be expressed as:

$$\mathbf{q} \in \mathcal{C}_\mathcal{T} \iff \exists \delta \in \Delta, \ \exists \sigma \in [0, 1] \ f(\mathbf{q}) = T(\delta)\mathcal{T}(\sigma) \tag{5}$$

From (5), only two equality function candidates $F(\mathbf{x})$ are available. The first equality function is suggested by [18] and can be interpreted as follows: an element $\mathbf{q} \in \mathcal{C}$ belongs to the constrained manifold if and only if the closest valid task pose to the end-effector pose $f(\mathbf{q}) \in \text{SE3}$ is the end-effector pose itself. Mathematically, it is equivalent to:

$$\mathbf{q} \in \mathcal{C}, \quad F(\mathbf{q}) = \min_{\delta,\sigma} |f(\mathbf{q}) - T(\delta)\mathcal{T}(\sigma)| \tag{6}$$

where the operator $||$ is any spatial euclidean distance. The distance used here is inspired from [19]:

$$\forall T_1, T_2 \in \text{SE3}, \ |T_1 - T_2| = \sqrt{||\mathbf{p}_1 - \mathbf{p}_2||^2 + \theta_{1,2}^2}$$

(where $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^3$ are the Cartesian positions of T_1, T_2 and $\theta_{1,2} \in [0, \pi]$ is the absolute rotation angle between both poses).

Therefore, (6) 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 (6) is not differentiable, that is a critical problem for applying the common sampling-based framework presented in [2].

One possible way to transform this equality function in 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 (6)), 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 $T(\delta)\mathcal{T}(\sigma)$ match together. With the new augmented configuration space $\tilde{\mathcal{C}}$, the constraint function F would become:

$$(\mathbf{q}, \sigma, \delta) \in \tilde{\mathcal{C}}, \ F(\mathbf{q}, \sigma, \delta) = |f(\mathbf{q}) - T(\delta)\mathcal{T}(\sigma)| \tag{7}$$

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

Consequently, (6) and (7) are not efficient for modeling the SCEEPP-manifolds.

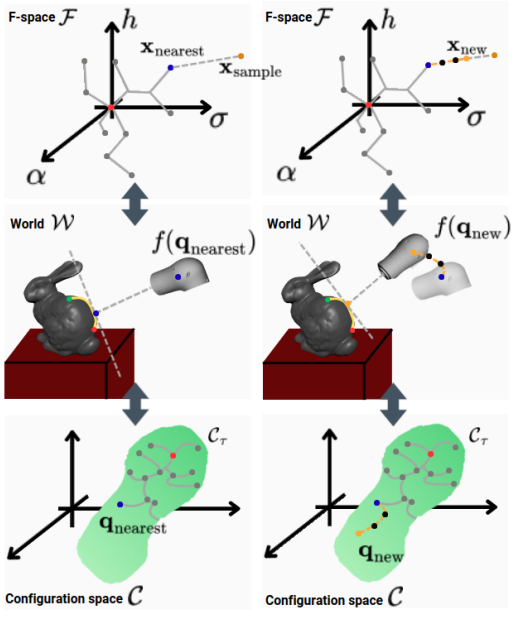


Fig. 3. Connection between the tolerance space $\mathcal{F}$, the world $\mathcal{W}$ and the configuration space $\mathcal{C}$. After finding the closest node x_{nearest} belonging to the initial graph G (left), a step is generated towards the sampling node x_{sample} to reach the new node x_{new} (right).

IV. REPARAMETRIZATION WITH THE F-SPACE

This section introduces our novel reparameterization approach for solving SCEEPP-problems. The reparametrization methodology consists in 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:

$$\mathcal{F} = [0, 1] \times \Delta$$

Each element in the F-space corresponds to an element in the manifold $\mathcal{C}_\mathcal{T}$. More specifically, an element $(\sigma, \delta) \in \mathcal{F}$ defines an end-effector pose $T(\delta)\mathcal{T}(\sigma)$ reachable by different configurations $\mathbf{q}$ within $\mathcal{C}_\mathcal{T}$. 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 purpose, we adapt the well-known RRT planner for the exploration of the F-space with a graph $G = (V, E)$. As described in Algorithm 1 (description based on [20]), at each iteration, the RRT planner starts by sampling a node of the unknown space (`SampleFree`), then, generates a step (`Steer`) from the closest node belonging to the initial graph G (found with `Nearest`), in direction of the sampling node, and finally, checks the validity of the step (`CheckValidityRRT`). If it is valid, it is added to the graph G. To apply the planner in the F-space, different points need to be defined.

Algorithm 1: `RRT`

Input: $\mathbf{x}_{\text{init}}$;
$V \leftarrow \mathbf{x}_{\text{init}}$; $E \leftarrow \oslash$;
for $i = 1, \ldots, n$ **do**
  $\mathbf{x}_{\text{rand}} \leftarrow$ `SampleFree`;
  $\mathbf{x}_{\text{nearest}} \leftarrow$ `Nearest`$(G = (V, E), \mathbf{x}_{\text{rand}})$;
  $\mathbf{x}_{\text{new}} \leftarrow$ `Steer`$(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{rand}})$;
  isValid $\leftarrow$ `CheckValidityRRT`$(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{new}})$;
  if *isValid* $==$ *True* **then**
    $V \leftarrow V \cup \{\mathbf{x}_{\text{new}}\}$;
    $E \leftarrow E \cup \{(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{new}})\}$;
return $G = (V, E)$;

A. Bounds of the F-space

As explained previously, each iteration starts by sampling a node (`SampleFree`) inside the unknown space. In most path planning problem, the unknown 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 unknown space is the F-space $\mathcal{F}$. Thus, the function `SampleFree` samples a node $\mathbf{x}_{\text{sample}}$ 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, a new edge of the graph is built inside the F-space $\mathcal{F}$. It starts from the closest node of the graph G to the sample node $\mathbf{x}_{\text{sample}}$ and does a small step towards $\mathbf{x}_{\text{sample}}$. The starting and the ending nodes of the step are named $\mathbf{x}_{\text{nearest}}$ and $\mathbf{x}_{\text{new}}$. As illustrated in Fig. 3, the step ($\mathbf{x}_{\text{nearest}}$, $\mathbf{x}_{\text{new}}$) is discretized in p points. For a RRT-planner, when a new step is constructed, the planner checks if the step is collision-free at each point with a checking function. Depending on the result of the function, either it is added to the graph or deleted. For F-RRT, which checking function `CheckValidityFRRT` is described in Algorithm 3, a step ($\mathbf{x}_{\text{nearest}}$, $\mathbf{x}_{\text{new}}$) is considered valid if all Cartesian poses induced by the points of the step are reachable with collision-free joint configurations $\mathbf{q} \in \mathcal{C}$. Each inverse kinematics between two Cartesian poses is solved using a constrained differential approach formulated as a Quadratic Programming problem (`QP`) [21]. Beyond computational efficiency, a Quadratic Programming based approach can account for joint position limitations and can be made robust to singularities. Overall, it provides an effective way to check if the Cartesian pose is reachable. If the pose is reachable with a joint configuration $\mathbf{q}$, collision checking (`CheckCollision`) is performed on this configuration. In our experiments, we use qpOASES [22] to implement the Quadratic Programming solver and collision detection is performed with the corresponding module in MoveIt! [23]. If the motion reaches the end-effector pose defined by $\mathbf{x}_{\text{new}}$ with a joint configuration $\mathbf{q}_{\text{new}}$ without any collision, the step ($\mathbf{x}_{\text{nearest}}$, $\mathbf{x}_{\text{new}}$) is considered as valid and added to the graph G. For joint consistency, the joint configuration $\mathbf{q}_{\text{new}}$, found during

Algorithm 2: `F-RRT`

Input: $\mathbf{x}_{\text{init}}, \mathbf{q}_{\text{init}}$;
$V \leftarrow \mathbf{x}_{\text{init}}, \mathbf{q}_{\text{init}}; E \leftarrow \emptyset$;
for $i = 1, \ldots, n$ **do**
  $\mathbf{x}_{\text{rand}} \leftarrow \texttt{SampleFree}()$;
  $(\mathbf{x}_{\text{nearest}}, \mathbf{q}_{\text{nearest}}) \leftarrow \texttt{Nearest}(G = (V, E), \mathbf{x}_{\text{rand}})$;
  $\mathbf{x}_{\text{new}} \leftarrow \texttt{Steer}(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{rand}})$;
  $(\text{isValid}, \mathbf{q}_{\text{new}}) \leftarrow$ `CheckValidityFRRT`$(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{new}}, \mathbf{q}_{\text{nearest}})$;
  if *isValid* $==$ *True* **then**
    $V \leftarrow V \cup \{(\mathbf{x}_{\text{new}}, \mathbf{q}_{\text{new}})\}$;
    $E \leftarrow E \cup \{(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{new}})\}$;
return $G = (V, E)$;

Algorithm 3: `CheckValidityFRRT`

Input: $\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{new}}, \mathbf{q}_{\text{nearest}}$
$\mathbf{x} \leftarrow \mathbf{x}_{\text{nearest}}; \mathbf{q} \leftarrow \mathbf{q}_{\text{nearest}}$;
$X_{\texttt{list}} \leftarrow \texttt{Discretized}(\mathbf{x}_{\text{nearest}}, \mathbf{x}_{\text{new}})$;
for $i = 1, \ldots, p$ **do**
  $\mathbf{x} \leftarrow X_{\texttt{list}}[i]$;
  $\mathbf{q}, \text{isReachable} \leftarrow \texttt{QP}(X_{\texttt{list}}[i], \mathbf{x}, \mathbf{q})$;
  if *isReachable* $==$ *False* **then**
    return *False;*
  isValid $\leftarrow$ `CheckCollision`$(\mathbf{q})$;
  if *isValid* $==$ *False* **then**
    return *False;*
return *True*;

the validation of the motion, is saved as an attribute of the node $\mathbf{x}_{\text{new}}$. Therefore, each step starting from $\mathbf{x}_{\text{new}}$ begins with its associated joint configuration $\mathbf{q}_{\text{new}}$.

C. Define the Starting and the Ending Regions

When defining the starting and the ending regions in the F-space, the essential part is the first term σ. A manipulator arm starting and finishing its task has necessarily and respectively its end-effector at the beginning of the Cartesian path, i.e., $\sigma = 0$ and at the end of the Cartesian path, i.e., $\sigma = 1$. The definition of tolerance values at the start and the end of the Cartesian path are meaningless. Thus, the sets of starting nodes and ending nodes are both hyperplanes of $\mathcal{F}$ and are respectively defined as follows $\mathcal{F}_0 = \{\mathbf{x} = (\sigma, \delta) \mid \sigma = 0\}$ and $\mathcal{F}_{\text{goal}} = \{\mathbf{x} = (\sigma, \delta) \mid \sigma = 1\}$.

These adaptations get a suitable RRT planner to operate in the F-space, named hereafter F-RRT and available in Algorithm 2. One interest of this adaptation is that we need to tune a similar number of hyper-parameters than RRT, i.e., the size of the step and its resolution.

V. EXPERIMENTS

In order to estimate the benefits and the drawbacks of our F-space approach, the adapted planner F-RRT is experimented in different simulated use cases. All the experiments presented below are made with a processor Intel® Core™ i7-12700K and written in C++. In the first experiment, we compare our adaptation with the general approaches proposed in the Open Motion Planning Library [4]: the Projection, the Tangent Space and the Atlas approaches. These lasts approaches can only work with the differentiable constraint function $F(\mathbf{q}, \sigma, \delta)$ defined in (7) and with the augmented configuration space $\tilde{\mathcal{C}}$. However, the Atlas and the Tangent Space approaches fail due to the large computations needed for the Jacobian of the function F. Therefore, for the first experiment, only the classic Projection approach associated to the RRT planner (PB-RRT) [5] is compared to F-RRT.

The first experiment, shown in Fig. 4, is a blasting task setup. It is composed of a FANUC CRX-10iA, a 6-DOF robot, whose end-effector needs to follow a linear Cartesian path for blasting. The F-space of the blasting task is detailed in the example given in Section III-A. In the experiment, an obstacle obstructs the nominal path forcing the manipulator arm to use its tolerances for finding a solution. Snapshots of the simulation of one solution given by our approach are available on Fig. 4. For each comparison, both methods, F-RRT and PB-RRT are executed 40 times. If a run lasts more than 10 minutes, it is stopped. The results are available in Table I. Globally, it can be seen that F-RRT is 10-times faster with a better success score than the projection-based planner. Furthermore, as shown in Fig. 5, our approach also works with redundant manipulators. That is one advantage of formulating inverse kinematics as a constrained differential problem with a Quadratic Programming approach as it provides a generic way to handle redundant robots [24] such as the Franka Panda arm.

TABLE I
EXPERIMENTS WITH SIMPLE AND COMPLEX ENVIRONMENTS

	F-RRT	PB-RRT
Simple Environments		
Success (%)	**100.0**	65.96
Time (s)	**16.16**	272.36
Complex Environments		
Success (%)	**84.76**	NA
Time (s)	**90.8**	NA

To test the robustness of our method in obstructed environment, 10 complex environments are created. Each environment is composed of one nominal path and many small cubes obstructing the path. The tolerances are the three Euler orientations (α, β, γ), only the Cartesian translation of the end-effector is constrained. Each complex environment is reloaded 10 times. The simulation and the path function of one of the solutions found by F-RRT are available in Fig. 6 and Fig. 7, respectively. No result was given by the classic projection-based method PB-RRT due to the high complexity of the problem. Even by rising the number of iterations of the algorithm in the projection process, it mostly failed. Conversely, our method presents significant results with an average success

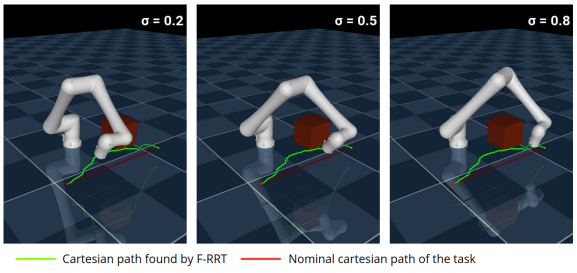


Fig. 4. F-RRT for blasting task along the red line. Three poses along the nominal Cartesian path (different values of σ) are represented here.

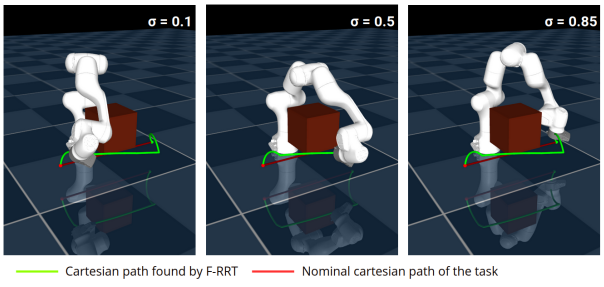


Fig. 5. F-space approach with a redundant robot (Franka Panda) for a blasting task. Three poses along the nominal Cartesian path (different values of σ) are represented here.

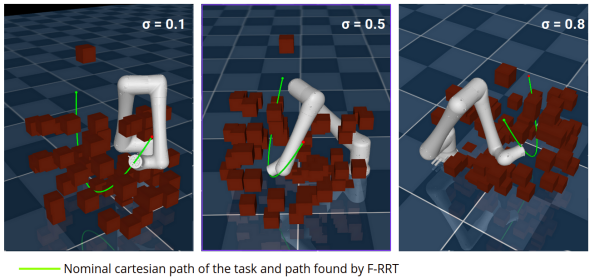


Fig. 6. F-RRT for a complex task obstructed by many small obstacles. Three poses along the nominal Cartesian path (different values of σ) are represented here.

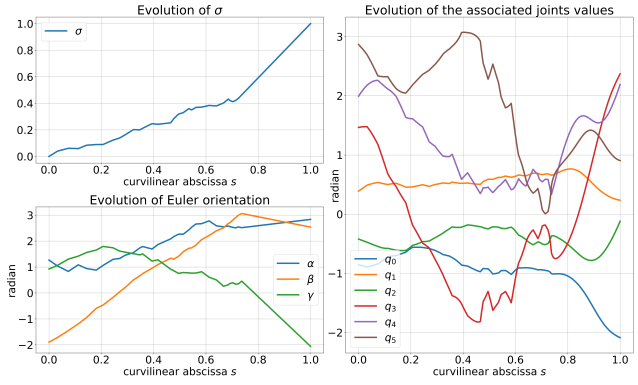


Fig. 7. Evolution of a path solution, along its curvilinear abscissa s, found by F-RRT for a complex task: path in the F-space (Left) and its associated path in the configuration space (Right)

of 84% for a reasonable computation time. Moreover, our implementation of F-RRT uses only around 8% of the CPU. This amount suggests that there is room for improvement in terms of computation time through parallelization.

Finally, with these experiments, we demonstrate that the F-space approach is much more promising for solving SCEEPP-problem than classical semi-constrained approaches. It can take care of numerous tolerances and it is largely faster than the existing methods.

VI. DISCUSSION

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

As explained in Section IV-B, exploring the F-space can be done with continuous sampling-based planners as RRT or RRT* where nodes are evaluated according to neighbouring nodes. The problem induced by the multiplicity of solutions of the inverse kinematics is only solved by the continuity constraint implied by the neighbouring nodes. Therefore, planners which compute isolated node in their process, such as the BIT* planner [25], can not 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 reparametrization process does not fully map the F-space to the entire configuration space. As a result, the reparametrization might not always intersect with the goal region ($\mathcal{X}_{\text{goal}}$) of the configuration space, even if a solution exists. However, since the goal region of the F-space $\mathcal{F}_{\text{goal}}$ is an hyperplane rather than a single point, as presented in Section IV-C, this reduces the possibility of not finding solutions in situations where there are.

By representing the Cartesian path of a task and its path parametrization σ as a single linear axis in F-space, our approach can be applied to any non-linear Cartesian path. The concept could also be extended in more dimensions: a task strictly constraining the end-effector onto a non-linear surface could be represented in the F-space with two axes with the NURBS [26] or T-spline [27] parametrizations .

VII. CONCLUSION

In this paper, we presented a new reparametrization 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 SCEEPP constraint problems 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 Cartesian non-linear surface constraints.

REFERENCES

[1] Steven M. LaValle and James J. Kuffner. Randomized Kinodynamic Planning. *The International Journal of Robotics Research*, 20(5):378–400, May 2001.

[2] Zachary Kingston, Mark Moll, and Lydia E Kavraki. Exploring implicit spaces for constrained sampling-based planning. *The International Journal of Robotics Research*, 38(10-11):1151–1178, 2019.

[3] Zachary Kingston, Mark Moll, and Lydia E Kavraki. Sampling-based methods for motion planning with constraints. *Annual review of Control, Robotics, and Autonomous Systems*, 1:159–185, 2018.

[4] Ioan A. Sucan, Mark Moll, and Lydia E. Kavraki. The Open Motion Planning Library. *IEEE Robotics & Automation Magazine*, 19(4):72–82, December 2012.

[5] Dmitry Berenson, Siddhartha S Srinivasa, Dave Ferguson, and James J Kuffner. Manipulation planning on constraint manifolds. In *Proceedings of IEEE the International Conference on Robotics and Automation*, pages 625–632, 2009.

[6] Beobkyoon Kim, Terry Taewoong Um, Chansu Suh, and Frank C Park. Tangent bundle RRT: A randomized algorithm for constrained motion planning. *Robotica*, 34(1):202–225, 2016.

[7] Léonard Jaillet and Josep M Porta. Path planning under kinematic constraints by rapidly exploring manifolds. *IEEE Transactions on Robotics*, 29(1):105–117, 2012.

[8] Zhenwang Yao and Kamal Gupta. Self-motion graph in path planning for redundant robots along specified end-effector paths. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2004–2009, 2006.

[9] Massimo Cefalo, Paolo Ferrari, and Giuseppe Oriolo. An opportunistic strategy for motion planning in the presence of soft task constraints. *IEEE Robotics and Automation Letters*, 5(4):6294–6301, 2020.

[10] Andrew Singletary, Karl Klingebiel, Joseph Bourne, Andrew Browning, Phil Tokumaru, and Aaron Ames. Comparative analysis of control barrier functions and artificial potential fields for obstacle avoidance. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 8129–8136, 2021.

[11] Yeping Wang, Pragathi Praveena, Daniel Rakita, and Michael Gleicher. Rangedik: An optimization-based robot motion generation method for ranged-goal tasks. In *Proceedings of the International Conference on Robotics and Automation*, pages 9700–9706, 2023.

[12] Jeroen De Maeyer, Bart Moyaers, and Eric Demeester. Cartesian path planning for arc welding robots: Evaluation of the descartes algorithm. In *Proceedings of th IEEE International Conference on Emerging Technologies and Factory Automation*, pages 1–8, September 2017. ISSN: 1946-0759.

[13] Jeroen De Maeyer, Mark Versteyhe, and Eric Demeester. Sampling-based Tube Following for Redundant, Planar Robotic Manipulators. In *Proceedings of the IEEE International Conference on Emerging Technologies and Factory Automation*, volume 1, pages 752–758, September 2018. ISSN: 1946-0759.

[14] Yu Sun, Xiangqun Meng, Miaocheng Yu, and Houjun Tang. Semi-constrained path planning for industrial robot with external axis. In *Proceedings of the Chinese Automation Congress*, pages 6180–6185. IEEE, 2020.

[15] Rishi K Malhan, Shantanu Thakar, Ariyan M Kabir, Pradeep Rajendran, Prahar M Bhatt, and Satyandra K Gupta. Generation of configuration space trajectories over semi-constrained cartesian paths for robotic manipulators. *IEEE Transactions on Automation Science and Engineering*, 20(1):193–205, 2022.

[16] Troy McMahon, Shawna Thomas, and Nancy M Amato. Sampling-based motion planning with reachable volumes for high-degree-of-freedom manipulators. *The International Journal of Robotics Research*, 37(7):779–817, 2018.

[17] Zhenwang Yao and Kamal Gupta. Path planning with general end-effector constraints: Using task space to guide configuration space search. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1875–1880, 2005.

[18] Mike Stilman. Global manipulation planning in robot joint space with task constraints. *IEEE Transactions on Robotics*, 26(3):576–584, 2010.

[19] Frank C Park and James E Bobrow. Geometric optimization algorithims for robot kinematic design. *Journal of Robotic Systems*, 12(6):453–463, 1995.

[20] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[21] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Fast resolution of hierarchized inverse kinematics with inequality constraints. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3733–3738, 2010.

[22] Hans Joachim Ferreau, Christian Kirches, Andreas Potschka, Hans Georg Bock, and Moritz Diehl. qpoases: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 6:327–363, 2014.

[23] Sachin Chitta, Ioan Sucan, and Steve Cousins. Moveit![ros topics]. *IEEE Robotics & Automation Magazine*, 19(1):18–19, 2012.

[24] Lucas Joseph, Vincent Padois, and Guillaume Morel. Towards x-ray medical imaging with robots in the open: safety without compromising performances. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 6604–6610, 2018.

[25] Jonathan D Gammell, Siddhartha S Srinivasa, and Timothy D Barfoot. Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3067–3074, 2015.

[26] Gerald Farin. From conics to nurbs: A tutorial and survey. *IEEE Computer Graphics and applications*, 12(5):78–86, 1992.

[27] Thomas W Sederberg, David L Cardon, G Thomas Finnigan, Nicholas S North, Jianmin Zheng, and Tom Lyche. T-spline simplification and local refinement. *ACM Transactions On Graphics*, 23(3):276–283, 2004.