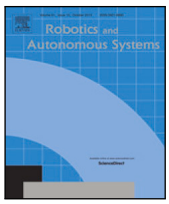
Robotics and Autonomous Systems 170 (2023) 104530

Contents lists available at ScienceDirect

Robotics and Autonomous Systems

journal homepage: www.elsevier.com/locate/robot



Sampling-based time-optimal path parameterization with jerk constraints for robotic manipulation

Huanhuan Huang [a], Houde Liu [a,*], Chongkun Xia [b], Hongwei Mei [a], Xuehai Gao [c], Bin Liang [d]

[a] Center of Intelligent Control and Telescience, Shenzhen International Graduate School, Tsinghua University, 518055, Shenzhen, China
[b] School of Advanced Manufacturing, Sun Yat-sen University, 518055, Shenzhen, China
[c] Research Center of Intelligent Control and Advanced Manufacturing, 518055, Shenzhen, China
[d] Navigation and Control Research Center, Department of Automation, Tsinghua University, 100084, Beijing, China

ARTICLE INFO

Keywords:
Time optimization
Path parameterization
Sampling
Constraint-checking
Jerk-bounded trajectory

ABSTRACT

In this paper, a sampling-based time-optimal path parameterization (S-TOPP) method is proposed to address time-optimal trajectory planning problems with bounded jerks. The key insight of S-TOPP is that a tree of feasible nodes connected by edges is established to find a time-optimal trajectory on the temporal dimension. The tree establishment process includes two major phases at each stage, namely sampling and one-step backtracking. In the sampling phase, a new sampling strategy integrating historical information is proposed to obtain superior samples whereby fewer samples can be controlled automatically, reducing the calculation loss. In one-step backtracking phase, a "lazy" strategy is used to lazily skip constraint-checking when evaluating local connections, enabling S-TOPP to avoid checking the vast majority of nodes that have no chance of being in an optimal trajectory. Simulations and real-world experiments validate the feasibility and practicability of S-TOPP. Results show that S-TOPP is an effective solution to jerk-bounded time-optimal trajectory planning, with features that are more in line with the needs of the practical tasks compared with other methods.

1. Introduction

As a fundamental research topic in the robotics community, trajectory generation plays an important role in controlling robots to achieve well-defined tasks with stability, reliability, and productivity. Trajectory planning represents the geometric path as a function of time, and the goal is to generate optimum reference inputs with constraints of kinematics and dynamics to the robot system so as to complete the manipulation tasks as desired. Time-optimal trajectory planning was first investigated by Bobrow et al. [1] and Shin et al. [2], aiming to find the most efficient way to traverse a specified geometric path in the configuration space under system constraints. Problems like unsmooth velocity and acceleration trajectories, joint errors, vibrations of robots, and motor damage frequently appear in time-optimal trajectory planning. In order to solve or alleviate these issues, jerk, the time derivative of acceleration, gradually gains widespread attention. Nevertheless, the non-linear and non-convex features of the jerk constraints complicate the trajectory optimization problem concerning jerk constraints, resulting in a significant rise in computation time. In the relevant studies, trajectories are primarily constructed by polynomial [3,4], B-spline [5–7], cubic spline [8–10], NURBS [11], and other interpolation functions. Functions like sine and sigmoid are widely adopted in literature [12–14], which utilize piecewise functions to establish trajectories. However, the abovementioned interpolation methods require massive parameters to generate smooth trajectories, causing an increase in computational time complexity. An alternative to constructing trajectories for time-optimal planning problems is time-optimal path parameterization (TOPP). Pham et al. [15] proposed TOPP-RA based on model predictive control (MPC) reachability theory to solve time-optimal problems with second-order constraints. TOPP-RA implicitly identifies switch points between accelerating and decelerating segments, which hugely saves computation time. On the downside, a prominent hurdle of TOPP-RA is that it fails to solve a problem with jerk constraints, hampering its applicability. Methods [16,17] use sequential quadratic programming (SQP) to address non-linear jerk constraints optimization problems, transforming the complex non-linear constrained optimization problem into a relatively simple quadratic programming problem. The computation time and memory consumption considerably increase with the problem size because the computation of the gradient and Hessian of the objective function is costly. Other methods [18–20] use pseudo-jerk to replace non-linear jerk constraints, which are bound to yield local optimum results. Debrouwere et al. [21] decomposed the non-convex jerk constraints into a difference of two convex functions to solve the problem by sequential convex programming (SCP), solving

* Corresponding author.
E-mail addresses: liu.hd@sz.tsinghua.edu.cn (H. Liu), xiachk5@mail.sysu.edu.cn (C. Xia).

https://doi.org/10.1016/j.robot.2023.104530
Received 29 November 2022; Received in revised form 23 August 2023; Accepted 8 September 2023
Available online 28 September 2023

a convex subproblem at each iteration. Consolini et al. [22] developed a line search algorithm based on SCP to solve the non-convex problem. Convex optimization-based methods are simple and robust on account of available convex optimization libraries but significantly slower. Kaserer et al. [23] presented a new dynamic programming (DP) approach named DP3 to address jerk-bounded trajectory optimization problems, resting on interpolation in the phase plane to guarantee continuous joint accelerations and joint torques. However, the DP3 method can only get sub-optimal outcomes with considerable computation time. Methods [24–26] use swarm intelligence algorithms, such as NSGA-II (non-dominated sorting genetic algorithm for two objectives), BAS (beetle antennae search), and PSO (particle swarm optimization) to optimize specific parameters so as to reach the goal of optimization. These methods rely significantly on parameter configuration, and poor accuracy results are often obtained. Pham et al. [27] proposed TOPP3 method to address third-order constraints in the 3D space $(s, \dot{s}, \ddot{s})$ using numerical integration. Nevertheless, it takes much time to identify switch points. Concrete and zero symmetric $\dddot{s}$ limit is given as jerk constraints in [28], which are invalid in practice because $\dddot{s}$ limit changes as any one of $\dot{s}, \ddot{s}$, and $\ddot{\boldsymbol{q}}$ changes. The only practical variable that can be given is $\ddot{\boldsymbol{q}}$. The abovementioned method fails to solve the challenging problem of mapping $\ddot{\boldsymbol{q}}$ into $\dddot{s}$.

We propose a sampling-based time-optimal path parameterization (S-TOPP) method to deal with jerk-bounded time-optimal trajectory planning problems. S-TOPP undergoes two main processes at each stage: sampling and one-step backtracking. In the sampling process, a novel sampling strategy to obtain superior samples is presented (Section 3.1). In the one-step backtracking (Section 3.2) process, constraint-checking is performed to evaluate connections between two adjacent stages incorporating with time-optimal objective. The key idea behind S-TOPP is to avoid the explicit construction of $\{\dot{s}, \ddot{s}, \dddot{s}\}$ values, which can be prohibitive when evaluating concrete values of $\{\dot{s}, \ddot{s}, \dddot{s}\}$ for one single step while ensuring feasibility all along. Instead, S-TOPP conducts a search that probabilistically probes the solution with a sampling strategy. The major contributions of the proposed method are as follows:

- The proposed S-TOPP method uses sampling and local connections to establish a tree of feasible nodes that generates a jerk-bounded time-optimal trajectory, which avoids directly dealing with non-convex and non-linear features of jerk constraints. It offers a new perspective on addressing time-optimal trajectory planning problems with bounded jerks.
- A novel sampling strategy is introduced in this paper, which allows the sampling process using historical information to obtain a small number of samples. Therefore, the proposed sampling strategy can significantly reduce the computational cost of trajectory optimization and yield superior results.
- Experimental results show that the proposed S-TOPP method generates smoother trajectories and performs better than TOPP-RA and DP3 when applied to practical tasks.

Although the proposed method is also based on sampling, S-TOPP is fundamentally different from other sampling-based methods, e.g. RRT*, PRM* [29], SST* [30,31], etc. The abovementioned sampling-based methods aim to seek a collision-free path in cartesian or configuration space and usually do not consider dynamics constraints, e.g. velocity, acceleration, or even higher order constraints. In order to control robots to achieve the desired task, those sampling-based methods need to combine with polynomial interpolation, TOPP, or other planning methods to generate a continuous trajectory with various optimization goals. The proposed method is a variant of TOPP methods, which aims to find a time-optimal trajectory under dynamics constraints given the geometric path. Fig. 1 depicts how the random tree is built in S-TOPP. The sampling space in S-TOPP is within the lower and upper bounds of squared path velocity x, and the validity condition is dynamics constraints. This paper is structured as follows. In Section **2**, we formally describe the time-optimal path parameterization problem. Section **3** presents a high-level description of S-TOPP, describes the main intuition behind its correctness, and defines three sampling strategies. In Section **4**, we compare the results among three sampling strategies, verifying the correctness and efficiency of the novel sampling strategy. Simulations and experiments are conducted to support our statements. Finally, Section **5** draws some conclusions and discusses future work. Notations are summarized in Table 1, and abbreviations are listed in Table 2.

Table 1
Notations for sampling-based time-optimal path parameterization method.

Notation	Definition
$\mathcal{P}$	A given geometric path
N	Segments number of $\mathcal{P}$
$\boldsymbol{q} \in \mathbb{R}^{n\times 1}$	Configuration vector of a n-dof robot system
$\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$	The inertia matrix
$\boldsymbol{C}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$	The matrix of Coriolis and centrifugal terms
$\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^{n\times 1}$	Gravitational terms and Coulomb friction terms
$\boldsymbol{\tau} \in \mathbb{R}^{n\times 1}$	The control torque
$\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \dddot{\boldsymbol{q}} \in \mathbb{R}^{n\times 1}$	Vectors of joint velocity, acceleration, and jerk
$s \in \mathbb{R}^{1\times 1}$	Path position
$\dot{s}, \ddot{s}, \dddot{s}$	Path velocity, acceleration, and jerk
x	Squared path velocity, i.e. $x = \dot{s}^2$
u	Path acceleration, i.e. $u = \ddot{s}$
$\mathcal{X}_i \in \mathbb{R}^{1\times 2}$	Explicit bound of x
$\mathcal{U}_i \in \mathbb{R}^{1\times 2}$	Explicit bound of u
$C_i \in \mathbb{R}^{1\times 2}$	Controllable set at stage i
$\mathcal{L} \in \mathbb{R}^{N\times 2}$	A collection of sampling range generated from TOPP-RA
$\mathcal{T}$	Random tree
r	Searching range for a sample's neighbors
ϵ	Tune coefficient for r
V^{i-1}_{open}	Leaf nodes collection of a random tree at stage $i-1$
$V^{i}_{\text{unvisited}}$	Collection of samples at stage i
V^{i-1}_{near}	Collection of a stage i sample's neighbors
V_{cost}	A dictionary with costs and corresponding nodes
f	The number of redundant samples
k	The number of samples for random and uniform sampling strategies
t_{cpu}	The computation time
t_{e}	The optimized trajectory duration
H_{C}	The vertical height of a cup
H_{LI}	The vertical height of an initial liquid level
H_{LA}	The vertical height of liquid level after execution

Table 2
Abbreviation list.

Abbreviation	Definition
S-TOPP	Sampling-based Time-Optimal Path Parameterization
TOPP	Time-Optimal Path Parameterization
TOPP-RA	A method proposed in [15]
SQP	Sequential Quadratic Programming
DP	Dynamic Programming
NURBS	Non-Uniform Rational B-Splines
BAS	Beetle Antennae Search
PASF	Path Acceleration Stepping Frozen A sampling strategy proposed in this paper
TOPP3	A method proposed in [27]
MPC	Model Predictive Control
SCP	Sequential Convex Programming
DP3	A method proposed in [23]
NSGA-II	Non-dominated Sorting Genetic Algorithm for two objectives
PSO	Particle Swarm Optimization

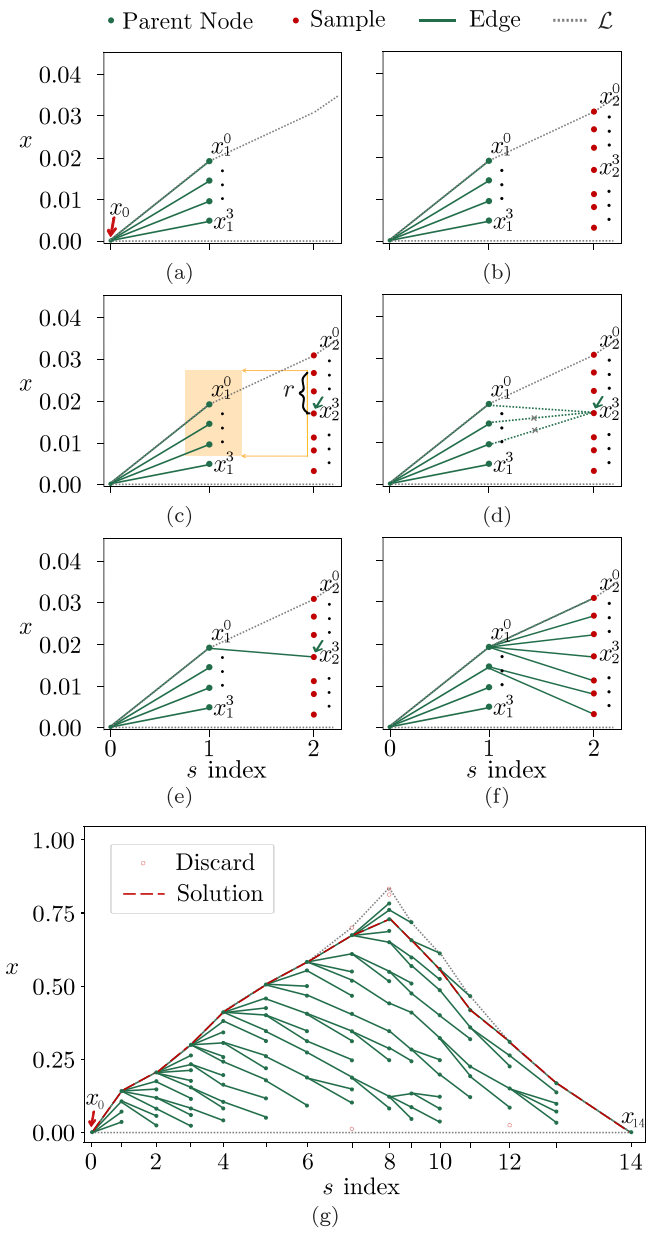


Fig. 1. The illumination of S-TOPP steps. (a) - (f) steps show how the tree extends leaf nodes in one stage, and (g) illustrates the optimal solution after building the tree to N+1 depth (assume N=14). (a) Sampling for stage $i = 1$ on the interval $[\mathcal{L}_1^{\min}, \mathcal{L}_1^{\max}]$, 4 samples are generated. Checking jerk constraints (checking method is described in Section 3.2 *ValidNode*()). If pass constraint-checking, connect all valid samples to x_0 (linked by solid green line). (b) Sampling for stage $i = 2$ on the interval $[\mathcal{L}_2^{\min}, \mathcal{L}_2^{\max}]$, 7 samples are generated. (c) Every sample should be processed in this stage. Taking x_2^3 as an example. Nodes (x_1^0, x_1^1, *and* x_1^2) in orange area are near parent nodes of x_2^3. r is the search range defined in Section 3.2. (d) Calculating costs and checking constraints between x_2^3 and its near parent nodes. It turns out that x_2^3 connects to x_1^0 with lowest cost and passing the constraint-checking. (e) Connecting x_2^3 to x_1^0 by draw a solid green line. (f) Connecting other samples to stage $i = 1$ if pass constraint-checking. Otherwise, discarding the sample when violating jerk constraints. (g) After the tree reaches $N + 1$ depth, connecting from x_{14} to parent node backward step by step until reaching x_0, the final solution is marked by red dash line. Time-optimal trajectory can be obtained through x. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

2. Problem formulation

Given a geometric path $\mathcal{P}$ in the configuration space, which is represented as a trajectory function $\boldsymbol{q}(t)_{t\in[0,t_e]}$. $\boldsymbol{q} \in \mathbb{R}^n$ is the n-dimensional vector representing the configuration of a n-dof robot system. A time-parameterization of the path $\mathcal{P}$ is an increasing scalar function $s : [0, t_e] \to [0, s_{\text{end}}]$. $\boldsymbol{q}(t)$ can be altered as $\boldsymbol{q}(s(t))$. In solving the above problem, we presume that $\mathcal{P}$ is divided into N segments with $N + 1$ points, s_i denotes the i-stage, then $i \in \{0, 1, \dots, N\}$. Based on these definitions, position sequence is obtained, i.e., $\{s_0, s_1, \dots, s_N\}$.

The joint-space dynamics of the robotic system can be formulated as

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{q}}^{\mathrm{T}}\boldsymbol{C}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{G}(\boldsymbol{q}) = \boldsymbol{\tau}. \tag{1}$$

In which, $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ denotes the inertia matrix, $\boldsymbol{C}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ denotes the matrix of Coriolis and centrifugal terms, $\boldsymbol{G}(\boldsymbol{q}) \in \mathbb{R}^{n\times 1}$ stands for gravitational terms and Coulomb friction terms. $\dot{\boldsymbol{q}} \in \mathbb{R}^{n\times 1}$, and $\ddot{\boldsymbol{q}} \in \mathbb{R}^{n\times 1}$ are first-order and second-order derivatives of $\boldsymbol{q}$ with respect to time, which denote joint velocity and joint acceleration, and $\boldsymbol{\tau} \in \mathbb{R}^{n\times 1}$ is the control torque. According to path parameterization formulation with an increasing scalar function $s(t)$, joint velocity and joint acceleration can be described as follows.

$$\begin{aligned} \dot{\boldsymbol{q}} &= \boldsymbol{q}'\dot{s}, \\ \ddot{\boldsymbol{q}} &= \boldsymbol{q}'\ddot{s} + \boldsymbol{q}''\dot{s}^2, \end{aligned} \tag{2}$$

where $\dot{s}$ and $\ddot{s}$ are first-order and second-order derivatives of s with respect to time t, denoting path velocity and path acceleration, respectively. $\boldsymbol{q}'$ and $\boldsymbol{q}''$ are first-order and second-order derivatives of $\boldsymbol{q}$ with respect to scaler s. Substituting Eq. (2) into Eq. (1), the joint-space dynamics of the robotic system with respect to $\boldsymbol{q}$ can be transformed as that with respect to s as

$$\boldsymbol{a}(s)u + \boldsymbol{b}(s)x + \boldsymbol{c}(s) = \boldsymbol{\tau}(s), \tag{3}$$

where $\boldsymbol{a}(s) = \boldsymbol{M}(\boldsymbol{q}(s))\boldsymbol{q}'(s)$, $\boldsymbol{b}(s) = \boldsymbol{M}(\boldsymbol{q}(s))\boldsymbol{q}''(s) + \boldsymbol{q}'(s)^{\mathrm{T}}\boldsymbol{C}(\boldsymbol{q}(s))\boldsymbol{q}'(s)$, $\boldsymbol{c}(s) = \boldsymbol{G}(\boldsymbol{q}(s))$ and $\boldsymbol{\tau}(s) = \boldsymbol{\tau}(\boldsymbol{q}(s))$. $u = \ddot{s}$ denotes the path acceleration, and $x = \dot{s}^2$ denotes the squared path velocity.

To solve an optimization problem with MPC reachability theory, we define a linear system with state x and input control u as

$$x_{i+1} = G(x_i, u_i), \tag{4}$$

$$x_i \in \mathcal{X}_i, u_i \in \mathcal{U}_i, i \in \{1, \dots, N\}, \tag{5}$$

where $\mathcal{X}_i$ and $\mathcal{U}_i$ are explicit constraints of state and input control at stage i. Perform first-order Tylor series expansion at x_i, and we obtain

$$x_{i+1} = x_i + \triangle_i \frac{\mathrm{d}x}{\mathrm{d}s}, \tag{6}$$

where $\triangle_i = s_{i+1} - s_i$. According to the chain rule, we present $\frac{\mathrm{d}x}{\mathrm{d}s}$ by a derivation of t yields

$$\frac{\mathrm{d}x}{\mathrm{d}s} = \frac{\mathrm{d}\dot{s}^2}{\mathrm{d}s} = 2\dot{s}\frac{\mathrm{d}\dot{s}}{\mathrm{d}t}\frac{\mathrm{d}t}{\mathrm{d}s} = 2\dot{s}\ddot{s}\frac{1}{\dot{s}} = 2\ddot{s} = 2u. \tag{7}$$

Then system (4) can be presented as follows.

$$x_{i+1} = x_i + 2\triangle_i u_i. \tag{8}$$

Assuming velocity limits $[\dot{\boldsymbol{q}}_{\min}, \dot{\boldsymbol{q}}_{\max}]$ and acceleration limits $[\ddot{\boldsymbol{q}}_{\min}, \ddot{\boldsymbol{q}}_{\max}]$, we obtain

$$\dot{\boldsymbol{q}}_{\min} \le \boldsymbol{q}'\dot{s} \le \dot{\boldsymbol{q}}_{\max}, \tag{9}$$

$$\ddot{\boldsymbol{q}}_{\min} \le \boldsymbol{q}'\ddot{s} + \boldsymbol{q}''\dot{s}^2 \le \ddot{\boldsymbol{q}}_{\max}, \tag{10}$$

$\mathcal{X}_i = [x_i^{\text{lower}}, x_i^{\text{upper}}]$ can be defined from the in Eq. (9) as

$$\begin{aligned} x_i^{\text{upper}} &= \min_j \left\{ \frac{\dot{q}_{\max,j}}{q'_{ij}} \mid q'_{ij} > 0 \ \text{ or } \ \frac{\dot{q}_{\min,j}}{q'_{ij}} \mid q'_{ij} < 0 \right\}, \\ x_i^{\text{lower}} &= \max_j \left\{ \frac{\dot{q}_{\min,j}}{q'_{ij}} \mid q'_{ij} > 0 \ \text{ or } \ \frac{\dot{q}_{\max,j}}{q'_{ij}} \mid q'_{ij} < 0 \right\}, \end{aligned} \tag{11}$$

where j is the j th column of $\boldsymbol{q}'_i$, $\dot{\boldsymbol{q}}_{\min}$, and $\dot{\boldsymbol{q}}_{\max}$. Note that we neglect the case where some $q'_{ij} = 0$. As for control constraint $\mathcal{U}_i$, we can define $\mathcal{U}_i = [-\infty, \infty]$ since no information can calculate the control constraint

explicitly. The reachability theory deals with the requirement that x_{i+1} of MPC iterative algorithm is bounded. The precursor set and controllable set of reachability theory are defined as follows [32].

Definition 1 (*Precursor Set Pre*($\mathcal{I}$))**.** The precursor set is a set of states, which to the set $\mathcal{I}$ for the system (8) is defined as

$$Pre(\mathcal{I}) = \{x \in \mathbb{R}^n : \exists u \in \mathcal{U},\ s.t.\, G(x,u) \in \mathcal{I}\}. \tag{12}$$

There must exist one input that can drive $Pre(\mathcal{I})$ into the target set $\mathcal{I}$ in one step while obeying state and input constraints.

Definition 2 ($N-$ *step Controllable Set* $C_N(\mathcal{Z})$)**.** At stage N, define a given target set $\mathcal{Z} \subseteq \mathcal{X}_N$, the N-step controllable set $C_N(\mathcal{Z})$ of the system (8) is subject to the constraints (5) is defined recursively as

$$C_i(\mathcal{Z}) = Pre(C_{i+1}(\mathcal{Z})) \cap \mathcal{X}_i,\ C_N(\mathcal{Z}) = \mathcal{Z},\ i \in \{0,\dots,N-1\}. \tag{13}$$

According to Definition 2, when $x_0 \in C_0(\mathcal{Z})$, a control sequence $\{u_0, u_1, \dots, u_{N-1}\}$ must exist that drives state x_0 to the target set $\mathcal{Z}$ in N steps while satisfying state and input constraints. Based on the controllable set theory, we can obtain another constraint for system (8) as

$$x_i + 2\triangle_i u_i \in C_{i+1}, \tag{14}$$

where C_{i+1} is calculated in the backward process of the N-step controllable set.

Considering jerk limits $[\dddot{q}_{\min}, \dddot{q}_{\max}]$, we obtain the inequation as

$$\dddot{q}_{\min} \le q'\ddot{s} + 3q''\ddot{s}\dot{s} + q'''\dot{s}^3 \le \dddot{q}_{\max}. \tag{15}$$

Rewrite the in Eqs. (10) and (15) as

$$Q'u + Q''x \le \ddot{Q}, \tag{16}$$

$$Q'\ddot{s} + 3Q''ux + Q'''x\sqrt{x} \le \dddot{Q}, \tag{17}$$

where $Q' = \begin{bmatrix} q' \\ -q' \end{bmatrix}$, $Q'' = \begin{bmatrix} q'' \\ -q'' \end{bmatrix}$, $Q''' = \begin{bmatrix} q''' \\ -q''' \end{bmatrix}$, $\ddot{Q} = \begin{bmatrix} \ddot{q}_{\max} \\ -\ddot{q}_{\min} \end{bmatrix}$, and $\dddot{Q} = \begin{bmatrix} \dddot{q}_{\max} \\ -\dddot{q}_{\min} \end{bmatrix}$.

Definition 3 (*Time* $-$ *optimal Problem*)**.** TOPP problem with bounded velocities, accelerations, and jerks is generally formulated as

$$\max\ u_i \quad \text{subject to} \begin{cases} x_i \in \mathcal{X}_i \\ u_i \in \mathcal{U}_i \\ x_i + 2\triangle_i u_i \in C_{i+1} \\ Q_i'u_i + Q_i''x_i \le \ddot{Q}_i \\ Q_i'\ddot{s}_i + 3Q_i''u_ix_i + Q_i'''x_i\sqrt{x_i} \le \dddot{Q}_i \end{cases}. \tag{18}$$

According to Eq. (2), maximal x equals maximal velocity, which results in time-optimal. Since the problem is solved iteratively, x_i and $\triangle_i$ are known for system (8). x_{i+1} as a system state can only be determined by input control u_i. Therefore, maximizing path acceleration u can achieve the aim of a time-optimal trajectory.

3. Sampling-based time-optimal path parameterization (S-TOPP)

According to the time-optimal problem (18), an infeasible situation occurs after several execution steps when linear pseudo-jerk is adopted to substitute the non-linear jerk constraints while setting $\frac{d\ddot{s}}{ds}$ as another optimization variable. As large u as possible are obtained while satisfying all constraints at the first several steps. However, when u begins to decrease, a sharp decline is needed to meet second-order constraints, which exactly breaches jerk constraints. Retreating a few steps to make x and u smaller is a palliative to fix the above problem, whereas which stage to address and what values to set are arduous. Even if the program can temporarily proceed a few more steps, an infeasible result would still occur when executing further. A sampling-based time-optimal path parameterization (S-TOPP) trajectory planning method enlightened by simple-based path planning methods is proposed to solve jerk-bounded time-optimal problems. S-TOPP establishes a tree that contains feasible nodes. After reaching the N-stage, a final optimal path will be obtained according to the given objectives. Algorithm 1 is the pseudo-code of S-TOPP, which briefly describes how this method is implemented. To elaborate S-TOPP vividly, Fig. 1 illuminates the pseudo-codes of the proposed S-TOPP method.

Algorithm 1 S-TOPP

Input: $x_0 = 0, q', q'', q'''$.
Output: A sequence of $\{x_0, x_1, ..., x_N\}$.
Initialization: Tree $\mathcal{T} \leftarrow x_0$, $V^0_{\text{open}} \leftarrow x_0$
$\mathcal{L} \leftarrow SampleRange(x_0, q', q'', q''')$
for $i \in [1...N]$ **do**
  $V^i_{\text{unvisited}} \leftarrow \varnothing$
  $V^i_{\text{open}} \leftarrow \varnothing$
  $V^{i-1}_{\text{near}} \leftarrow \varnothing$
  $V^i_{\text{unvisited}} \leftarrow SamplePASF(\mathcal{L})$
  for $x \in V^i_{\text{unvisited}}$ **do**
    $V^{i-1}_{\text{near}} \leftarrow NearParents\ (x, V^{i-1}_{\text{open}}, V^i_{\text{unvisited}})$
    $y \leftarrow FindParent(x, q', q'', q''', V^{i-1}_{\text{near}})$
    if y **then**
      $V^i_{\text{open}} \leftarrow x$
      $Connect(y, x)$
      $\mathcal{T} \leftarrow x$
    end if
  end for
end for
if $i == N$ **then**
  return $Solution(\mathcal{T})$
else
  return Failure
end if

Section 3.1 introduces a novel sampling strategy, path acceleration stepping frozen, enabling the effectiveness and efficiency of S-TOPP. Section 3.2 elaborates on how a sample is validated and connected to a parent node, and a constraint-checking strategy is presented.

3.1. Path acceleration stepping frozen (PASF) sampling strategy

SampleRange() decides in which range to sample. Remove jerk constraints from the time-optimal problem (18) yields

$$\max\ u_i \quad \text{subject to} \begin{cases} x_i \in \mathcal{X}_i \\ u_i \in \mathcal{U}_i \\ x_i + 2\triangle_i u_i \in C_{i+1} \\ Q_i'u_i + Q_i''x_i \le \ddot{Q}_i \end{cases} \tag{19}$$

The lower and upper bound of x_i under velocity and acceleration constraints can be efficiently computed by (19)[15]. Set $\mathcal{L}$ as a collection of the upper and lower bound of x. *SamplePASF*() works to choose samples at stage i in range $\mathcal{L}_i$. Firstly, we define random sampling and uniform sampling strategies, which are the standard ways of sampling in the sampling-based methods.

Definition 4 (*Random Sampling*)**.** Discretizing the range of $[\mathcal{L}_i^{\min}, \mathcal{L}_i^{\max}]$ with uniform segments into m points, where m is a relatively large integer, randomly select k points as the samples.

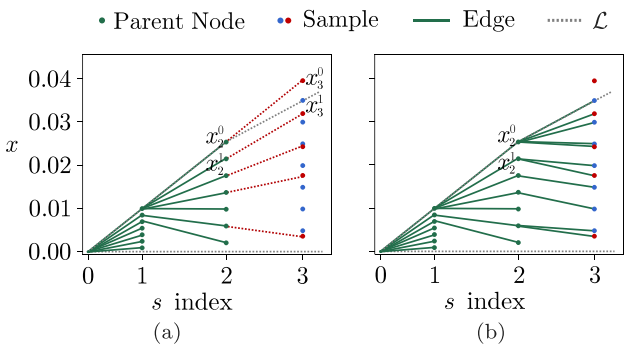


Fig. 2. Example of PASF sampling strategy. Green dots are nodes before stage 3 that have been added to the tree. Green lines represent parent–child relationships between nodes. (a) Additional f samples marked as blue dots are generated by uniform sampling on the interval $\left[\mathcal{L}_i^{\min}, \mathcal{L}_i^{\max}\right]$, which are essential to make S-TOPP robust. The red dash lines are extension lines of edges, which assumes that $u_2^j = u_1^j$. By Eq. (20), samples with historical information are generated, such as x_3^0 and x_3^1 marked as red dots. (b) $x_3^0 > \mathcal{L}_3^{\max}$, so x_3^0 should be discarded. According to Eq. (25), the larger value of x_3, the lower the cost between stages 2 and 3. Therefore, x_3^1 prioritizes connecting to x_2^0 instead of x_2^1 when constraints are met. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Definition 5 (*Uniform Sampling*)**.** Discretizing the range of $\left[\mathcal{L}_i^{\min}, \mathcal{L}_i^{\max}\right]$ with uniform segments into k points, then select all these points as the samples.

A trajectory is a time series function with an intrinsic feature that the future rests on the past. Random sampling and uniform sampling strategies do not consider historical information. Therefore, we propose a new sampling strategy, exploiting the historical information to conduct the sampling.

Definition 6 (*Path Acceleration Stepping Frozen Sampling*)**.** Assigning the value of u_{i-2} to u_{i-1} for every parent node at stage $i-1$ to calculate the corresponding x_i and let the outcomes be the samples.

$$x_i = x_{i-1} + 2 \triangle_{i-1} u_{i-2}. \tag{20}$$

Fig. 2 depicts how to use the PASF sampling strategy to generate a sample with historical information. When growing the tree from the root, S-TOPP samples f nodes from $\mathcal{L}_1$ by uniform sampling strategy to launch. S-TOPP uses $i-1$ stage path acceleration u to predict i stage samples, which fully considers the situation where path acceleration u remains unchanged. Parent nodes at stage $i-1$ are relatively closer to each other (e.g. x_2^0 and x_2^1), then a parent node can have children generated from its neighbors. When a potential parent node connects with several children, there will be some parent nodes coming to an end. Therefore, the distribution of next step samples will exhibit a regional focusing phenomenon as shown in Fig. 5(c), which puts limited samples in the region with a higher hit rate. Thus, the PASF sampling strategy can generate relatively accurate samples and have the number of samples at each stage under control.

An idea of directly applying the sampling-based method in space $(\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \dddot{\boldsymbol{q}})$ is ruled out for the below reasons. TOPP-RA has high efficiency in solving a time-optimal problem with second-order constraints, which is a way to combine velocity and acceleration constraints to obtain a smaller range of sampling. When addressing in space $(\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}, \dddot{\boldsymbol{q}})$, the DOFs of robots can perplex the problems. After sampling at stage i, these samples are put in $V_{\text{unvisited}}^i$, a collection of uncertain nodes waiting to connect with a parent node at stage $i-1$. If a sample in $V_{\text{unvisited}}^i$ is estimated with a positive result, S-TOPP puts it in V_{open}^i, a collection of leaf nodes in tree $\mathcal{T}$ waiting to connect with children at stage $i+1$.

3.2. One-step backtracking

At stage i, *FindParent*() function works to find a parent node from V_{open}^{i-1} for each sample in $V_{\text{unvisited}}^i$. However, the number of parent nodes could be dozens or even hundreds at each stage, and each sample has at most one parent node in V_{open}^{i-1}. If the scope of parent nodes is too large, evaluating connections would be time-consuming. Since the gap between x_{i-1} and x_i should be small, S-TOPP trims the searching range r (Fig. 1(c)), whereby S-TOPP can select fewer parent nodes from V_{open}^{i-1} as candidates for the finding parent process. At stage i, we define

$$\triangle_x = \frac{\mathcal{L}_i^{\max} - \mathcal{L}_i^{\min}}{length(V_{\text{unvisited}}^i)}, \tag{21}$$

where *length*() works to count the number of samples in $V_{\text{unvisited}}^i$. Then r can be presented as

$$r = \triangle_x \cdot \epsilon, \tag{22}$$

where ϵ is a parameter that needs to be set manually, which would determine how many parent nodes will be selected. A constant r is ruled out because the scale is too small to involve decimals, making it hard to find an appropriate value. When (22) is exerted, another parameter ϵ can be set as an integer to determine r. Nevertheless, this paper does not study how to find the optimal ϵ, and we set an experimental value during our test. Based on (22), S-TOPP obtains a collection of pending parent nodes, denoted as V_{near}^{i-1}.

S-TOPP uses trajectory duration between two adjacent nodes to calculate the cost. Performing first-order Taylor expansion for s at t_{i-1} and t_i yields

$$s(t_{i-1}) = s(t_i) + (t_{i-1} - t_i)\dot{s}(t_i), \tag{23}$$

$$s(t_i) = s(t_{i-1}) + (t_i - t_{i-1})\dot{s}(t_{i-1}). \tag{24}$$

Conducting (24)–(23), we get cost between two stages as:

$$t_i - t_{i-1} = \frac{2(s(t_i) - s(t_{i-1}))}{\dot{s}(t_{i-1}) + \dot{s}(t_i)} = \frac{2\triangle_{i-1}}{\dot{s}(t_{i-1}) + \dot{s}(t_i)}. \tag{25}$$

Learning from the lazy collision checking in FMT*[33], S-TOPP adopts the same mindset to find a parent node for every sample, and we named it lazy constraint-checking. Lasy constraint-checking process includes: (1) sorting the costs; (2) checking constraints from the minimum cost parent node until a positive result occurs to return the exact parent node or discard the sample. Lasy constraint-checking allows skipping many constraint-checking processes, which can hugely reduce computation time. The pseudo-code of *FindParent*() can be found in Algorithm 2.

Algorithm 2 FindParent

Input: $x, \boldsymbol{q}', \boldsymbol{q}'', \boldsymbol{q}''', V_{\text{near}}^{i-1}$.
Output: A node from V_{near}^{i-1} **or** None
Initialization: $V_{\text{cost}} \leftarrow \varnothing$.

```
for z ∈ V_near^{i-1} do
    V_cost ← Cost(z, x)   ▷ V_cost is a dictionary, cost is saved as a key, z is saved as a value.
end for
for (c, z) ∈ Sort(V_cost) do   ▷ Sort the keys.
    if ValidNode(q'_{i-1}, q''_{i-1}, q'''_{i-1}, z, x) then
        return z
    end if
end for
return None
```

ValidNode() is in charge of checking jerk constraints. The acceleration constraint-checking is involved here to ensure acceleration constraints will not be violated again. When managing $\ddot{s}$ in Eq. (15), a discrete notation for each stage is requisite. u_{i-1} is calculated by x_{i-1} and x_i, which adopts forward differences. If $\ddot{s}_{i-1}$ is calculated through forward differences, we can obtain the below expression.

$$\ddot{s}_{i-1} = \frac{u_i - u_{i-1}}{t_i - t_{i-1}}. \tag{26}$$

When at stage i, we can only calculate u_{i-1} through x_i. It should be noted that u_i requires x_{i+1}, which is unknown. It is inadvisable to

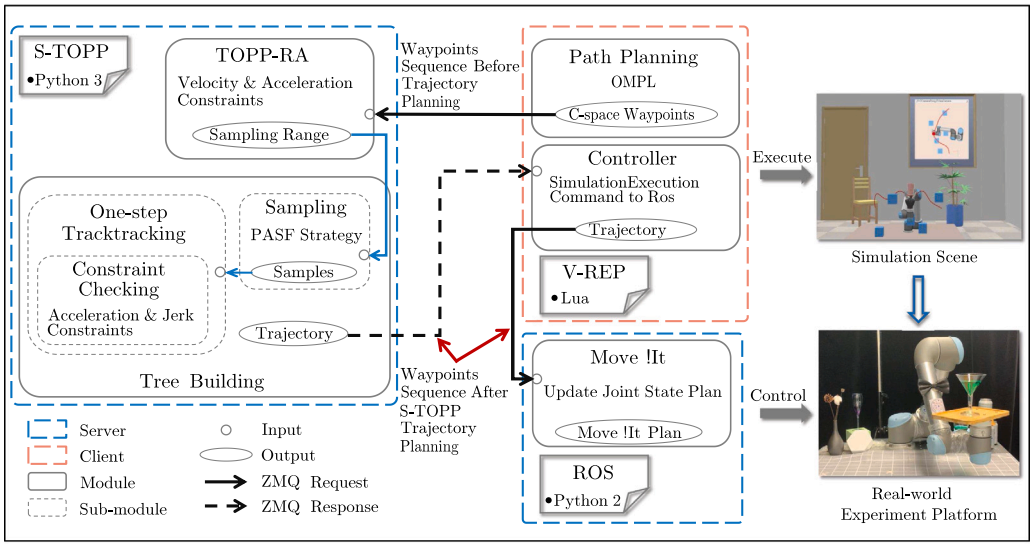


Fig. 3. Architecture of implementation.

predict one x_{i+1} for stage i, which may cause an endless loop. If at stage i, using predicted x_{i+1} to determine that x_i meets jerk constraints. However, when entering stage $i+1$, if the actual x_{i+1} differs from the predicted one, the last stage of x_i cannot be guaranteed to meet jerk constraints, leading to re-predict a proper x_{i+1}. Backward differences help approximate the derivatives if data in the future are not yet available. Primarily when the data in the future rests on the derivatives approximated from the data in the past. Based on the above analysis, S-TOPP utilizes backward differences to deal with $\dddot{s}$ discretization as

$$\dddot{s}_{i-1} = \frac{u_{i-1} - u_{i-2}}{t_{i-1} - t_{i-2}}. \tag{27}$$

Presenting t by scalar s yields

$$\dddot{s}_{i-1} = \frac{u_{i-1} - u_{i-2}}{\triangle_{i-2}} \sqrt{x_{i-1}}. \tag{28}$$

According to Eq. (28), u_{-1} and $\triangle_{-1}$ do not exist when $i = 1$, but x_0 equals zero. Therefore, $\dddot{s}_0$=0. When x_{i-1}, u_{i-1}, and $\dddot{s}_{i-1}$ are obtained, S-TOPP can start to judge if $\dddot{q}_{i-1}$ satisfies jerk constraints.

After finding the parent node, add the sample to V^i_{open}, then connect the sample to the parent node. *Connect*() addresses the storing of cost, the values of u_{i-1}, and $\dddot{s}_{i-1}$. Since a node only has one parent node, the trajectory duration between two nodes is not used directly, but the accumulative cost is used when storing the cost.

$$co(x_i) = \min_{x_{i-1} \in V^{i-1}_{\text{near}}} \{co(x_{i-1}) + Cost(x_{i-1}, x_i)\}, \tag{29}$$

where $Cost(x_{i-1}, x_i)$ is calculated by Eq. (25). A postprocessing function *Solution*() can readily extract the final trajectory by the last node when employing Eq. (29).

4. Experiments and results

First, the performance of different sampling strategies for the proposed S-TOPP method is analyzed. We compare S-TOPP accounting for the constraints $\dot{q}$, $\ddot{q}$, and $\dddot{q}$ against TOPP-RA [15] and DP3 [23] to verify the effectiveness and efficiency of S-TOPP. The proposed S-TOPP method is implemented and tested on 4 predefined geometric paths shown in Fig. 4 (the end-effector cartesian geometric path is demonstrated) subjecting to velocity, acceleration, and jerk constraints. Real-world experiments of tray-carrying tasks are conducted to endow practical significance for S-TOPP. The experiments were performed on Ubuntu with an Intel i7-10700 @ 2.90 GHz CPU and 16-Gb RAM. The optimization results are experimentally validated on Universal Robots 5. We implement S-TOPP with Python 3, and the communications among systems are shown in Fig. 3. Numerical values of joint constraints used for optimization are shown in Table 3, enabling the enforceability of real-world tasks.

Table 3
Numerical values of joint constraints used for optimization. $\dddot{q}$ 1 and $\dddot{q}$ 2 are different joint jerk constraints applied in experiments.

Limits	$\dot{q}$ (rad/s)	$\ddot{q}$ (rad/s^2)	$\dddot{q}$ 1 (rad/s^3)	$\dddot{q}$ 2 (rad/s^3)
All joints	[−1, 1]	[−10, 10]	[−200, 200]	[−50, 50]

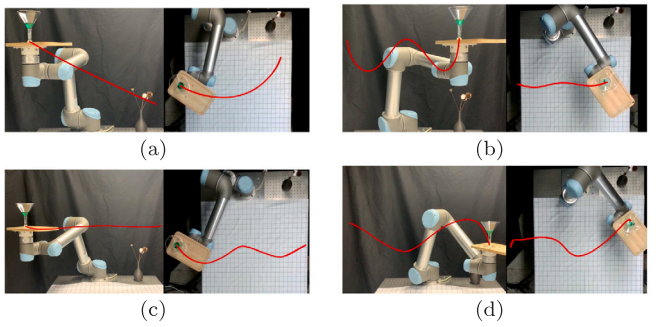


Fig. 4. Red lines demonstrate four different types of geometric paths for the end-effector in cartesian space. (a) Path 1 is an arc from the right bottom to the left top. (b) Path 2 is a vertical squiggly line. (c) Path 3 is a horizontal squiggly line. (d) Path 4 combines vertical and horizontal squiggly lines. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

4.1. Sampling strategy experiments and results

The numerical values of constraints used for optimization are displayed in Table 3. We select one geometric path (Path 2) and jerk limits (200 rad/s^3) to perform the comparison. k=50 for random and uniform sampling strategies, and f=5 for PASF sampling strategy. The obtained results for the computation time t_{cpu} and optimized trajectory duration t_{e} of different sampling strategies and various ε values are listed in Table 4. The tree-building processes of partial trees are shown in Fig. 5. The tree fails to grow to where close to $\mathcal{L}^{\max}$ when ε = 5 for random and uniform sampling strategies, resulting in a longer extension of t_{e}. The above phenomenon is not apparent for PASF as shown in Fig. 5(c). Properly increasing ε to 10 enables shorter t_{e},

Table 4
Optimized trajectory duration and computation time of different sampling strategies.

Sampling strategy	Random (k=50)		Uniform (k=50)		PASF (ours) (f=5)	
ε	5	10	5	10	5	10
t_{cpu} (s)	0.1755	0.2014	0.1689	0.2111	0.1037	0.1117
t_{e} (s)	2.0163	1.8410	1.9619	1.8189	1.7984	1.7617

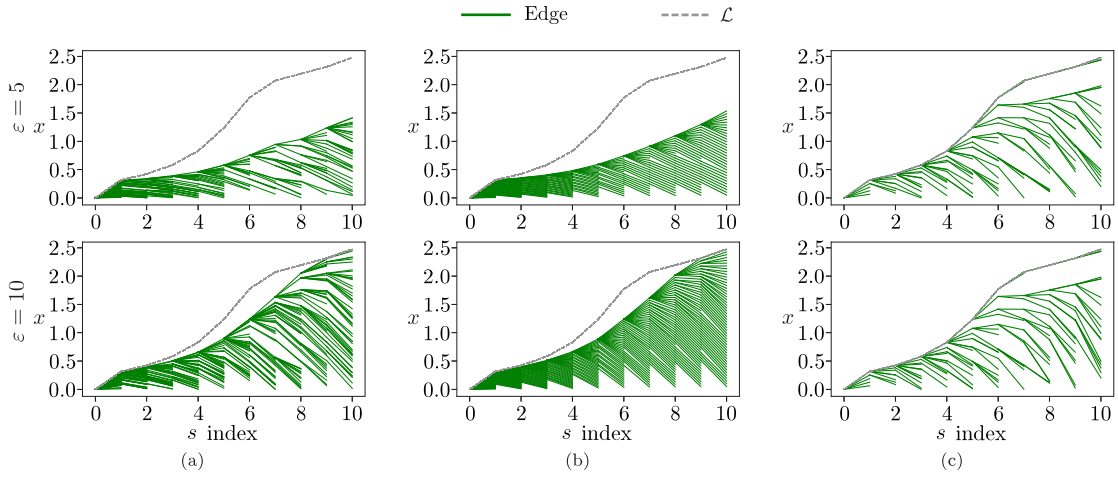


Fig. 5. Insight of tree building at $i = 0, \ldots, 10$ for random, uniform, and PASF sampling strategies. (a) Random sampling strategy. (b) Uniform sampling strategy. (c) PASF sampling strategy.

reducing 8.69% for random, 7.29% for uniform, and 2.04% for PASF compared to $\varepsilon = 5$. However, $\varepsilon = 10$ would cause an increase in the search range for random and uniform sampling strategies, leading to considerable t_{cpu} in the backtracking process (e.g. increasing 14.76% for random and 24.99% for uniform). t_{cpu} of PASF sampling strategy, by contrast, merely increases 7.71% at $\varepsilon = 10$. It is evident that random and uniform sampling strategies are sensitive to the value of ε. PASF sampling strategy grows the tree with sparse nodes, leading to lower t_{cpu} than random and uniform sampling strategies under the same ε. Furthermore, the feature of relying on historical information renders a faster search for the parent node even with different ε. When $\varepsilon = 10$, t_{cpu} of PASF is about 44.54% lower than random and 47.09% lower than uniform. Moreover, the corresponding t_{e} of PASF is approximately 4.31% lower than random and 6.98% lower than uniform. PASF sampling strategy outperforms the other two strategies both in terms of computation time and optimized trajectory duration. Analysis from the view of joint velocity and acceleration trajectories is presented in Appendix A.

Taking Path 2 and Path 4 concerning jerk limits 200 rad/s^3 as examples to analyze the value of f and ε for the PASF sampling strategy. Fig. 6 shows the computation time t_{cpu} and optimized trajectory duration t_{e} for Path 2 and Path 4. The experimental results reveal that both t_{cpu} and t_{e} increase as f increases when other parameters are constant. Fig. 6(b) indicates that increasing ε properly can enhance the performance both in t_{cpu} and t_{e} when f is relatively large. After testing, we select f=5, ε=10, and PASF sampling strategy for the follow-up experiments. We utilize the discretization method in [15] to discrete the geometric path $\mathcal{P}$ into $N + 1$ points. t_{cpu} increases as N becomes large, but t_{e} could also increase. We set experimental N for different paths in the simulations and experiments.

4.2. Simulations and results

We compare S-TOPP with TOPP-RA in the simulation to verify that the proposed method can effectively address the jerk constraints. Furthermore, the results reveal that S-TOPP is more efficient when compared with DP3 method. Limits of joint velocity, acceleration, and jerk are shown in Table 3. 4 geometric paths are applied in different

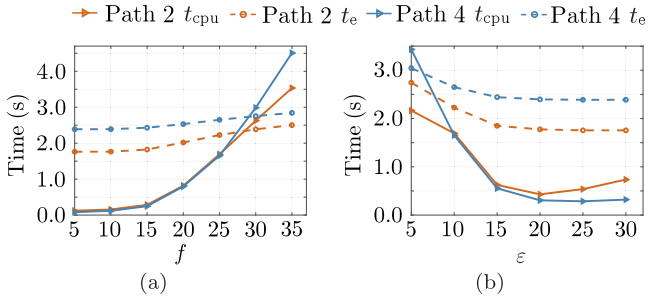


Fig. 6. t_{cpu} and t_{e} for Path 2 and Path 4 concerning different f and ε. (a) ε=10 with various f: t_{cpu} and t_{e} increase for both Path 2 and Path 4 as f becomes large. (b) f=25 with various ε: t_{cpu} and t_{e} decrease for both Path 2 and Path 4 when increasing ε. t_{e} tends to be constant, but t_{cpu} increases when continuously increasing ε. Ultimately, t_{cpu} tends to be stable when ε is large enough to include all parent nodes, which equals to $r = \infty$.

methods and various jerk limits. Table 5 records the computation time t_{cpu} and optimized trajectory duration t_{e} of different methods concerning various jerk limits and paths. TOPP-RA yields a lower optimized trajectory duration, but does not consider jerk limits. Compared to TOPP-RA, t_{cpu} of S-TOPP is increased reasonably due to the addition of jerk constraints. For instance, t_{cpu} of S-TOPP concerning jerk limits 200 rad/s^3 increases about 9.6 times for Path 1, 4.9 times for Path 2, 2.9 times for Path 3, and 2.7 times for Path 4 compared to TOPP-RA. By contrast, t_{cpu} of the DP3 method concerning jerk limits 200 rad/s^3 increases about 403 times for Path 1, 313 times for Path 2, 374 times for Path 3, and 393 times for Path 4 compared to TOPP-RA. t_{cpu} of S-TOPP increases as the jerk limits decrease (50 rad/s^3 jerk limits of t_{cpu} increase about 0.4 times for Path 1, 1.8 times for Path 2, 2.2 times for Path 3, and 1.4 times for Path 4 compared to 200 rad/s^3), which is still in the millisecond level. Obviously, the proposed S-TOPP method has significant advantages in efficiency. From the perspective of optimized trajectory duration t_{e}, S-TOPP yields a slightly longer t_{e} than the TOPP-RA method, which makes sense on the reason that more valid constraints in optimal problems will decrease

Table 5
Computation time and optimized trajectory duration of various trajectory planning methods and jerk limits.

Method	$\dddot{q}$ Limits (rad/s^3)	Path 1		Path 2		Path 3		Path 4	
		t_{cpu} (s)	t_e (s)	t_{cpu} (s)	t_e (s)	t_{cpu} (s)	t_e (s)	t_{cpu} (s)	t_e (s)
TOPP-RA	∞	0.0160	1.7860	0.0190	1.7508	0.0170	1.7880	0.0189	2.3800
DP3	200	6.4628	1.8193	5.9657	1.8020	6.3829	1.8513	7.4565	2.4250
	50	4.7030	2.0284	6.1266	2.6541	4.0800	2.9365	8.3987	3.2963
S-TOPP (ours)	200	0.1695	1.7924	0.1117	1.7617	0.0668	1.7978	0.0698	2.3879
	50	0.2279	1.9424	0.3151	1.9993	0.2114	1.9930	0.1656	2.6072

the feasible range. When considering jerk constraints, accelerates need to be appropriately reduced at some stages to prevent violating jerk constraints at switching points. t_e of S-TOPP is about 0.36% for Path 1, 0.62% for Path 2, 0.55% for Path 3, and 0.33% for Path 4 longer than TOPP-RA concerning jerk limits 200 rad/s^3. However, DP3 generates inferior t_e that is about 1.86% for Path 1, 2.92% for Path 2, 3.54% for Path 3, and 1.89% for Path 4 longer than TOPP-RA concerning jerk limits 200 rad/s^3. As analyzed in Section 3.1 about the PASF sampling strategy, S-TOPP adopts PASF sampling strategy can obtain relatively accurate samples and accelerate calculation. We conclude that S-TOPP is superior in yielding optimal trajectories. As the jerk limits decrease, t_e increases. Nevertheless, the extension of t_e for S-TOPP is obviously less than DP3 as shown in Table 6. The above analysis of computation time and optimized trajectory duration elucidates that the proposed S-TOPP method not only has a huge advantage in computation time but also can generate more optimal trajectories than the DP3 method. For demonstration purposes, 4 geometric paths are subsequently stitched together. Fig. 7 shows the trajectories of joint velocity, acceleration, and jerk. Fig. 7(a) displays trajectories generated from TOPP-RA without jerk constraints. The maximum joint jerk of each path is also marked, showing that joint jerk can reach more than 240 rad/s^3 without concerning jerk constraints. Fig. 7(b) and Fig. 7(c) show DP3 and S-TOPP trajectories for 50 rad/s^3, respectively. Compared with TOPP-RA, DP3 and S-TOPP effectively restrict the jerk, which is shown in jerk trajectories. In acceleration trajectories, the slopes of acceleration for S-TOPP and DP3 are relatively smaller than TOPP-RA, resulting small jerk. In particular, the joint velocity curves of S-TOPP become smoother at switch points than TOPP-RA, which is more in line with the actual requirement of a robotic system. Moreover, S-TOPP generates relatively smoother joint velocity trajectories than DP3 while having a shorter optimized trajectory duration. Table 7 shows the degree of decline in maximum joint jerk for DP3 and the proposed S-TOPP method concerning jerk limits 50 rad/s^3 compared to TOPP-RA. Under the assumption that the jerk of all joints is limited to 50 rad/s^3, the maximum value of joint jerk reduces by averaged 86.61% for DP3 and 84.88% for S-TOPP compared to the ones from TOPP-RA. Thereupon, the joint acceleration trajectories transform more smoothly between the acceleration and deceleration segments at the switch points, where the problem of acceleration mutation is effectively restrained. Although the averaged degree of decline in joint jerk from S-TOPP is smaller than the ones from DP3, the performance of S-TOPP can still excel DP3. Conversely, S-TOPP generates more optimal and smoother trajectories. As shown in Table 6, the optimized trajectory duration becomes longer when jerk limits are smaller, indicating that a lower degree of decline in joint jerk can yield more optimal trajectories when all jerk of joints is successfully bounded. Before applying the proposed method to real-world verification, we conduct a simulation on V-REP with a full cup of particles carrying scene, which is shown in Appendix B.

Table 6
Extension percentage of optimized trajectory duration with jerk limits 50 rad/s^3 compared to jerk limits 200 rad/s^3.

Method	Path 1	Path 2	Path 3	Path 4
DP3	11.49%	47.28%	58.62%	35.93%
S-TOPP (ours)	8.37%	13.49%	10.86%	9.18%

Table 7
Degree of decline in joint jerk for DP3 and S-TOPP concerning jerk limits 50 rad/s^3 compared to TOPP-RA.

Method	Joint 1	Joint 2	Joint 3	Joint 4	Joint 5	Joint 6
DP3	88.48%	83.12%	86.02%	85.37%	87.96%	88.72%
S-TOPP (ours)	87.40%	79.41%	87.36%	85.32%	83.95%	85.82%

4.3. Real-world experiments and results

In addition to demonstrating the effectiveness of the proposed S-TOPP method by trajectories figures, we also verify the effectiveness of S-TOPP through physical verification. The liquid sloshing of tray-carrying tasks can reveal the jerk intensity, which can be observed intuitively. Different types of cups (Fig. 8) are utilized to perform the experiments. We define that H_{LI} denotes the initial vertical height of liquid level, H_C denotes the vertical height of a cup, and H_{LA} denotes the vertical height of liquid level after execution. The result of "Fail" is defined to represent that liquid spills out of the cup, while "Success" is the opposite. Section 4.3.1 conducts tray-carrying experiments under joint constraints and Section 4.3.2 is under end-effector constraints. The experimental video records experiments of these two sections.

4.3.1. Constraints on joints

This section applies the results of the above simulation in practice. After testing, jerk limits at 50 rad/s^3 are suitable for real-world tray-carrying tasks execution. We set other real-world experiments in Appendix D, focusing on executing tasks with approximately equal optimized trajectory durations for different methods.

Table 8 presents the case scenarios and experimental results systematically. The vertical height data in the table are measured manually, which exists measurement errors. Three types of cups are employed in each path. In order to present results concisely, only the results of one cup for each path are recorded in Table 8. TOPP-RA fails to accomplish all the tasks that liquid spills out of the cups, and $H_{LA} < H_{LI}$ as recorded in Table 8. Both DP3 and the proposed S-TOPP method execute the tasks successfully without spilling any drop of liquid out of the cups in the cases of Path 1 and Path 2. However, DP3 fails to accomplish the tasks concerning Path 3 and Path 4. Though both DP3 and S-TOPP limit the joint jerk under 50 rad/s^3, DP3 method can still fail the tasks. It is evident that when there is no jerk bounded, liquid can slosh violently and spill out of the cup a lot as shown in Fig. 9(a), Fig. 9(b), Fig. 9(c), and Fig. 9(d) for TOPP-RA. Compared to DP3, S-TOPP handles the jerk constraints effectively and even performs better in real-world experiments as shown in Fig. 9(e) and Fig. 9(f).

4.3.2. Constraints on the end-effector

As liquid sloshing analysis stated in Appendix C, the end-effector velocity, acceleration, and jerk can greatly affect the liquid sloshing intensity. Though Section 4.3.1 states that the proposed S-TOPP method outperforms other methods in practice by constraining on joints, the end-effector velocity, acceleration, and jerk are not limited to the same conditions among comparison methods. Therefore, this section conducts tray-carrying tasks by applying velocity, acceleration, and

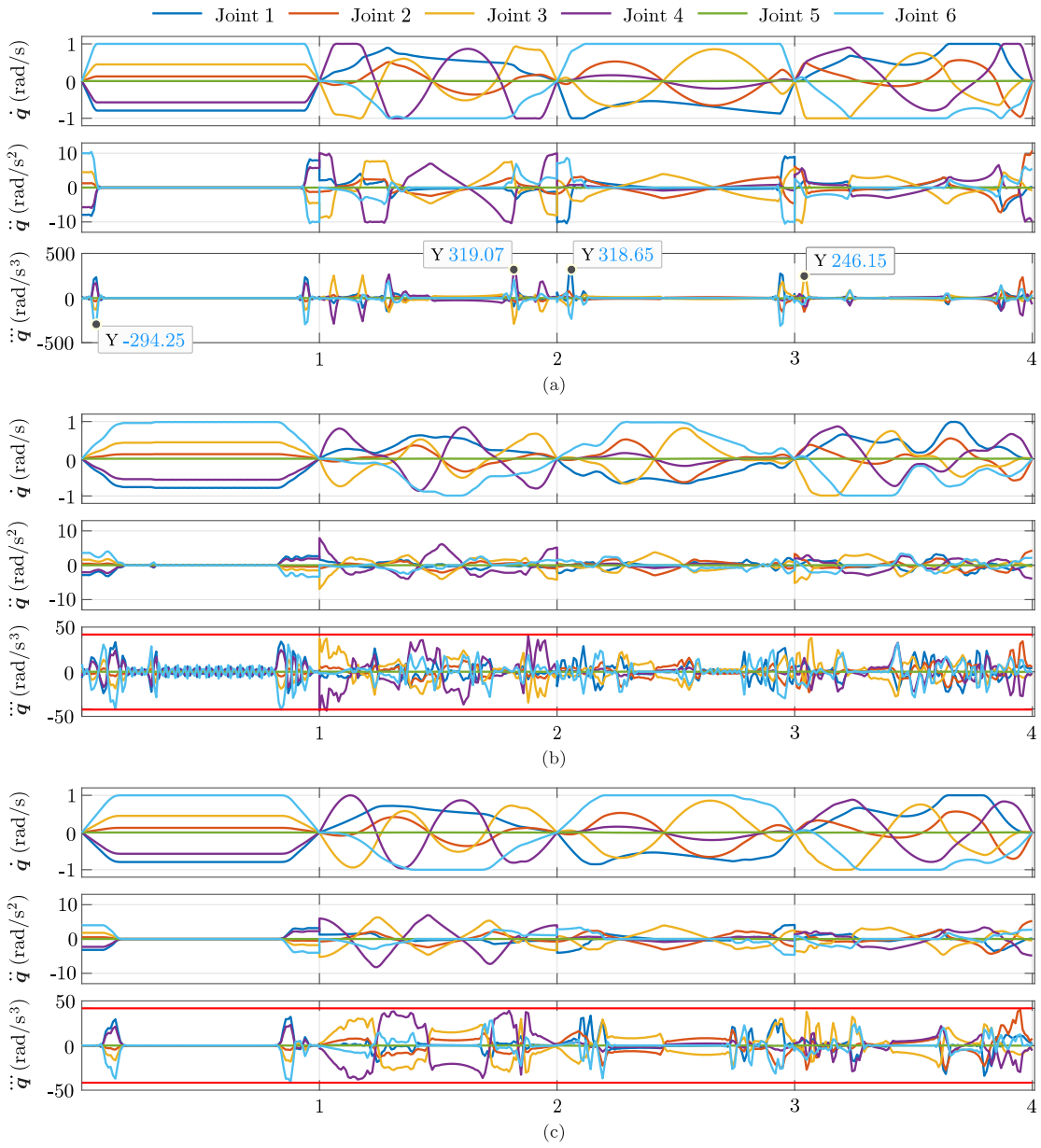


Fig. 7. Trajectories of velocity, acceleration, and jerk for different methods. (a) TOPP-RA: No jerk limits. (b) DP3: Jerk limits 50 rad/s^3. (c) S-TOPP: Jerk limits 50 rad/s^3.

Table 8
Case scenarios and results when constraining on joints.

Path	Cup	H_{LI} vs. H_{C} (cm)	Method	t_{cpu} (s)	t_{e} (s)	Results	H_{LA} vs. H_{LI} (cm)
1	1	4.9 vs. 7	TOPP-RA	0.0160	1.7860	Fail	4.4 vs. 4.9
			DP3	4.7030	2.0284	Success	4.9 vs. 4.9
			S-TOPP	0.2279	1.9424	Success	4.9 vs. 4.9
2	2	7.8 vs. 9	TOPP-RA	0.0190	1.7508	Fail	7.7 vs. 7.8
			DP3	6.1266	2.6541	Success	7.8 vs. 7.8
			S-TOPP	0.3151	1.9993	Success	7.8 vs. 7.8
3	3	9.8 vs. 12	TOPP-RA	0.0170	1.7880	Fail	8.7 vs. 9.8
			DP3	4.0800	2.9365	Fail	9.75 vs. 9.8
			S-TOPP	0.2114	1.9930	Success	9.8 vs. 9.8
4	2	7.9 vs. 9	TOPP-RA	0.0189	2.3800	Fail	7.1 vs. 7.9
			DP3	8.3987	3.2963	Fail	7.85 vs. 7.9
			S-TOPP	0.1656	2.6072	Success	7.9 vs. 7.9

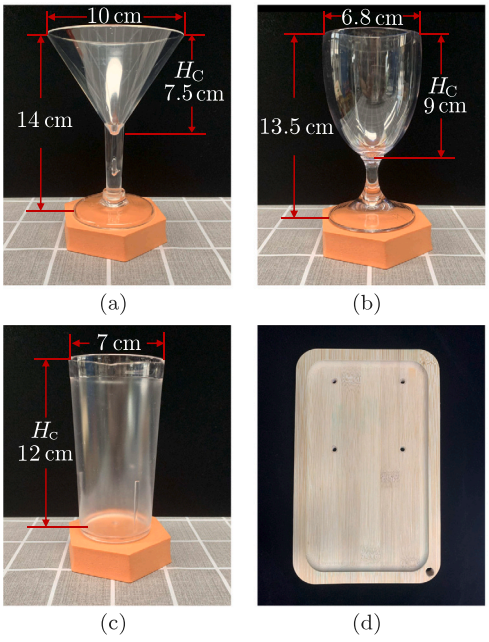


Fig. 8. Three types of cups and one tray. (a) Cup 1. (b) Cup 2. (c) Cup 3. (d) Tray: 26 × 16 cm.

Table 9
Numerical values of constraints on the end-effector. $\dot{q}_{\text{el}}$, $\ddot{q}_{\text{el}}$, and $\dddot{q}_{\text{el}}$ are limits of end-effector velocity, acceleration, and jerk, respectively. x, y, and z directions share the same limit.

Path	$\dot{q}_{\text{el}}$ (m/s)	$\ddot{q}_{\text{el}}$ (m/s^2)	$\dddot{q}_{\text{el}}$ (m/s^3)
1	[−0.35, 0.35]	[−2.0, 2.0]	[−15, 15]
2	[−0.35, 0.35]	[−4.0, 4.0]	[−30, 30]
3	[−0.40, 0.40]	[−2.5, 2.5]	[−15, 15]
4	[−0.30, 0.30]	[−2.0, 2.0]	[−15, 15]

jerk constraints on the end-effector. To remove the effect of the end-effector pose changing, we fix the pose of the end-effector. To enrich experiments, we configure different constraints on the end-effector for different paths, which are recorded in Table 9.

In this section, we use one cup (Cup 2) to perform experiments. Table 10 records experimental results systematically. Trajectories of the end-effector are shown in Fig. 10. The velocities and accelerations are bounded under the same limits for all three methods. Though TOPP-RA with only velocity and acceleration constraints on the end-effector generates slightly shorter execution time, the trajectories of velocity and acceleration are unsmooth, which fails to accomplish all the tasks with $H_{\text{LA}} < H_{\text{LI}}$. By adding jerk constraints on the end-effector, the end-effector jerks of DP3 and the proposed S-TOPP method are bounded effectively. Moreover, S-TOPP takes a shorter execution time than DP3 to execute all tasks successfully without spilling any liquid out of the cups, while DP3 still fails in path 1 and path 3 cases. Either constraining on joints or the end-effector, the proposed S-TOPP method can accomplish the tasks without failure.

5. Conclusions

This paper has introduced a sampling-based time-optimal path parameterization (S-TOPP) method to generate a time-optimal trajectory with jerk constraints. Jerk constraints are intractable and irritating by their natures of non-convex and non-linear. S-TOPP is a novel workaround that achieves good performance while avoiding dealing with these features. An effective sampling strategy path acceleration stepping frozen (PASF) is presented and proved to obtain shorter optimized trajectory duration efficiently when compared with the random sampling and uniform sampling strategies. Based on the trajectory intrinsic temporal feature, PASF strategy utilizes the historical information to predict the samples, automatically picking superior samples and having the number of samples under control. During the one-step backtracking process, a "lazy" constraint-checking is performed to avoid unnecessary checking, thus hugely decreasing the computation time while building a valid tree. Simulation results show that S-TOPP takes about 5 times on average than TOPP-RA to generate trajectories, and the computation time of S-TOPP is still less than 0.4 s for all cases. However, DP3 takes about 370 times than TOPP-RA. The simulation verifies that the proposed S-TOPP method effectively addresses jerk constraints. The execution time of S-TOPP is slightly longer than TOPP-RA by about 0.46% on average. Nevertheless, the execution time of DP3 is longer than TOPP-RA by about 2.55% on average, which indicates that S-TOPP performs better than the DP3 method. Real-world experimental results on tray-carrying tasks show that S-TOPP can be adopted in practice and accomplish tasks safely and efficiently. S-TOPP with smoother velocity trajectory than DP3 results is also verified in real-world experiments, indicating that S-TOPP is more in line with the actual requirement of a robotic system. Future work will be devoted to automatically finding a proper number of redundant samples for different cases and researching how to calculate r to get better neighbors.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article

Acknowledgments

This work was supported by National Natural Science Foundation of China [grant numbers 92248304, 62203260], The Shenzhen Science Fund for Distinguished Young Scholars [grant number RCJC2021070 6091946001], Guangdong Young Talent with Scientific and Technological Innovation [grant number 2019TQ05Z111], Tsinghua SIGS Cross Research and Innovation Fund [grant number JC2021005], Guangdong Basic and Applied Basic Research of China [grant number 2023A151 5011773].

Appendix A. Trajectories of various sampling strategies

We set other limits for $\dot{q}$, $\ddot{q}$, and $\dddot{q}$ as shown in Table A.1 to further analyze $\dot{q}$ and $\ddot{q}$ trajectories for different sampling strategies. The reason to set other limits is that such a setting can produce sharp distinctions in the trajectories. Fig. A.1 displays trajectories of $\dot{q}$ and $\ddot{q}$ generated by different sampling strategies. $\ddot{q}$ trajectory of random sampling shows that the randomness makes the acceleration trajectory zigzag (Fig. A.1(a)). Compared with PASF, the acceleration is relatively small at the time range of $[0.8\,\text{s}, 1\,\text{s}]$ for both random and uniform sampling strategies (Fig. A.1(a) and Fig. A.1(b)), leading to a longer optimized trajectory duration. Moreover, a smoother joint velocity trajectory is generated by PASF sampling strategy compared to random and uniform sampling strategies.

Appendix B. V-Rep simulation

When setting the limits as Table 4, the differences in particle spilling simulation among the results of various $\dddot{q}$ are subtle. Thereupon, we apply $\dot{q}$ and $\ddot{q}$ limits as shown in Table A.1, and different $\dddot{q}$ limits (1000 rad/s^3, 500 rad/s^3, 300 rad/s^3, 100 rad/s^3, and 50 rad/s^3). Fig. B.1 shows spilling out moments of simulation results. When $\dddot{q}$ limits are bounded to 1000 rad/s^3 and 500 rad/s^3, the spilling situation has little difference with TOPP-RA, but the differences become distinct when $\dddot{q}$ limits are smaller, such as 300 rad/s^3, 100 rad/s^3, and 50 rad/s^3.

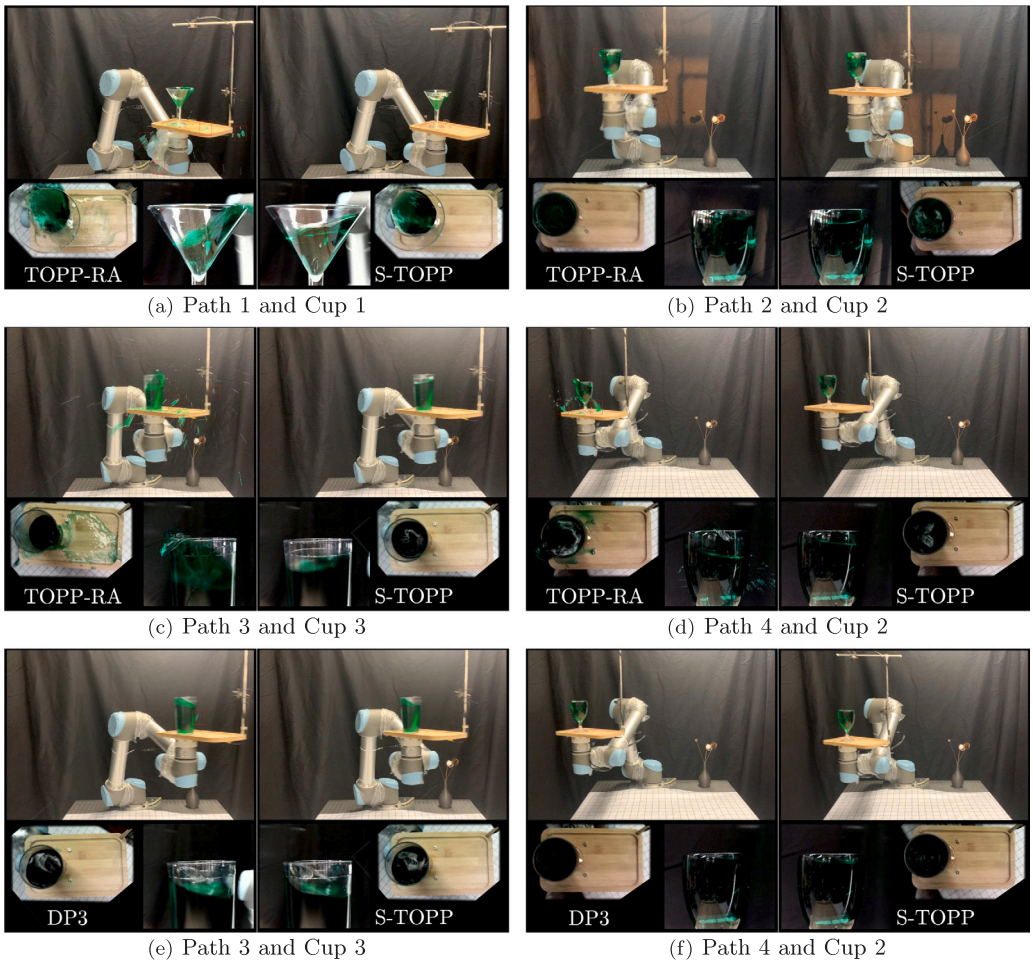


Fig. 9. Real-world experimental moments. Not a drop of liquid spills out of cups for the proposed S-TOPP method. The dropping-out liquid is partially marked with the red block diagram. It should be noted that getting frames of a fast-moving object from a video prevents the picture from being displayed in high definition. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 10
Case scenarios and results when constraining on the end-effector.

Path	H_{LI} vs. H_{C} (cm)	Method	t_{cpu} (s)	t_{e} (s)	Results	H_{LA} vs. H_{LI} (cm)
1	7.1 vs. 9	TOPP-RA	0.0465	1.7060	Fail	6.3 vs. 7.1
		DP3	13.040	1.8084	Success	6.7 vs. 7.1
		S-TOPP	0.2344	1.7911	Success	7.1 vs. 7.1
2	7.9 vs. 9	TOPP-RA	0.0428	1.8398	Fail	7.85 vs. 7.9
		DP3	6.9777	2.1070	Success	7.9 vs. 7.9
		S-TOPP	0.1990	1.9823	Success	7.9 vs. 7.9
3	6.8 vs. 9	TOPP-RA	0.0453	1.3509	Fail	5.8 vs. 6.8
		DP3	8.0684	1.9457	Success	6.7 vs. 6.8
		S-TOPP	0.3131	1.6564	Success	6.8 vs. 6.8
4	6.3 vs. 9	TOPP-RA	0.0499	2.2283	Fail	5.9 vs. 6.3
		DP3	6.9614	2.5018	Success	6.3 vs. 6.3
		S-TOPP	0.1735	2.4283	Success	6.3 vs. 6.3

Table A.1
Velocity, acceleration, and jerk limits of joints.

Limits	Joint1	Joint2	Joint3	Joint4	Joint5	Joint6
$\dot{q}$ (rad/s)	3.92	2.61	2.85	3.92	3.02	6.58
$\ddot{q}$ (rad/s^2)	19.7	16.8	20.7	20.9	23.7	33.5
$\dddot{q}$ (rad/s^3)	1000	1000	1000	1000	1000	1000

Appendix C. Liquid sloshing analysis

When a container undergoes accelerated motions, the liquid in the container will move back and forth, which is termed sloshing. For convenience, we take the ideal liquid (the density of liquid ρ is a constant, and the kinematic viscosity of the liquid $\mu = 0$.) to analyze the fluid mechanics. The surface tension is also neglected for the sake of convenience. Assume the ideal liquid is irrotational, incompressible, and non-viscous. We analyze how the motion would affect the free surface of the liquid. The Navier–Stokes equation and continuity equation describe the motion of a liquid particle in an inertial frame $\sum_I$ of

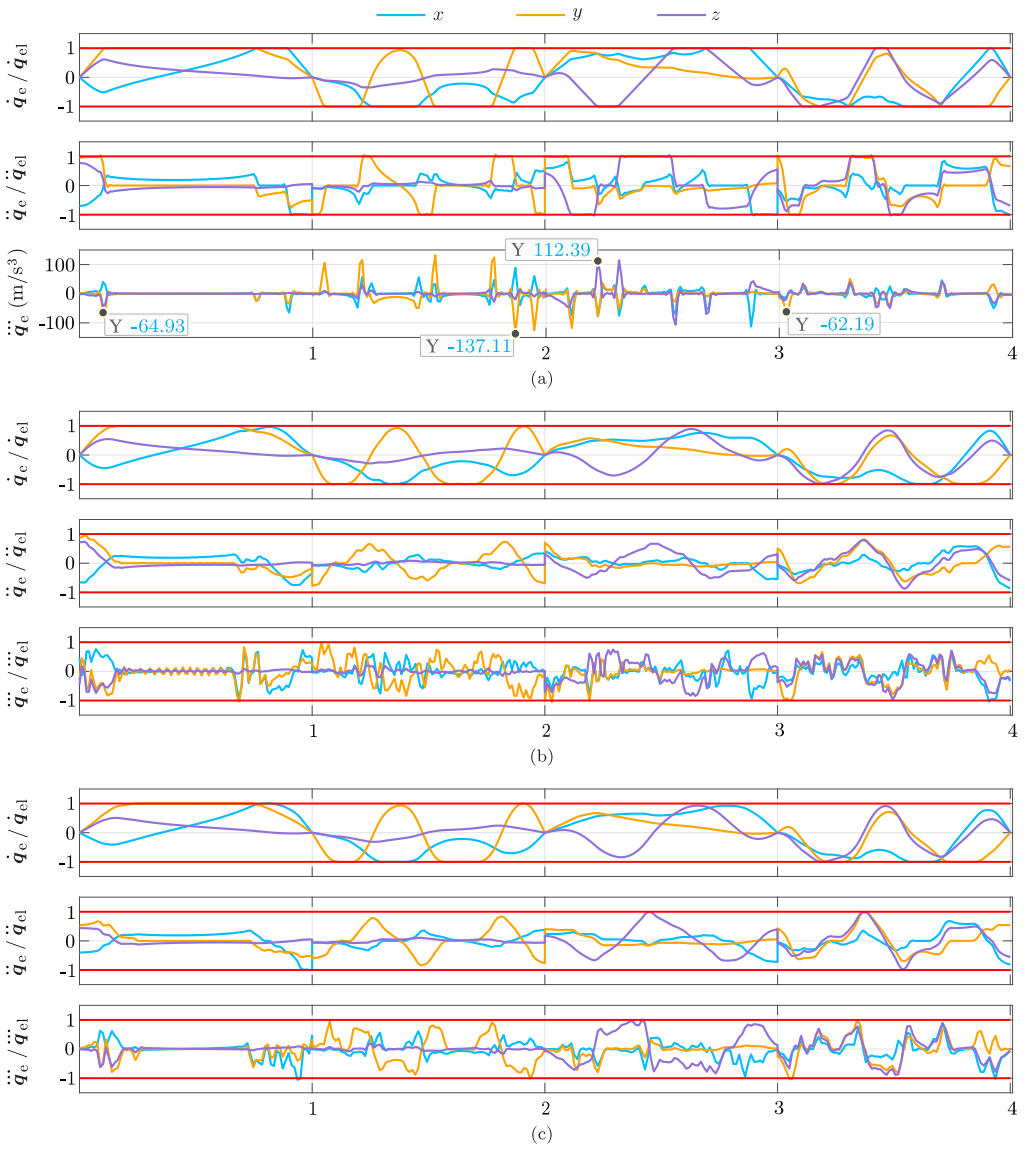


Fig. 10. End-effector trajectories of velocity, acceleration, and jerk for different methods. $\dot{q}_e$ represents the end-effector position. (a) TOPP-RA: No jerk limits on the end-effector. (b) DP3: jerk limits on the end-effector. (c) S-TOPP: jerk limits on the end-effector.

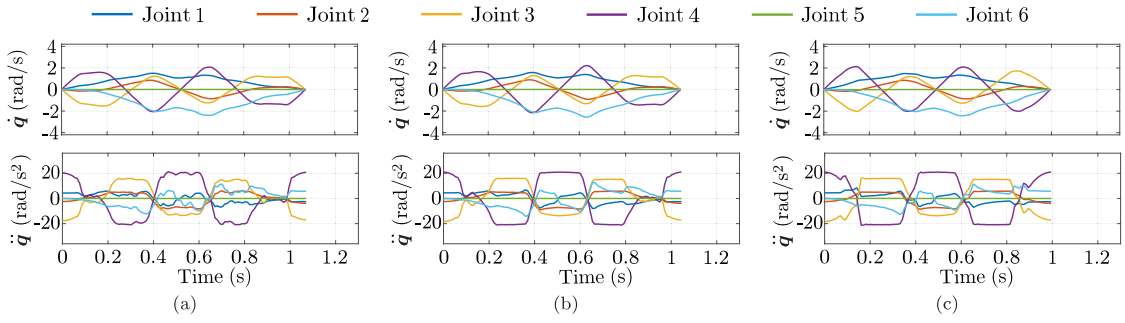


Fig. A.1. The comparison of velocity and acceleration trajectories with different sampling strategies ($\epsilon = 10$), $k = 50$, $f = 5$. (a)Random sampling strategy. (b)Uniform sampling strategy. (c) PASF sampling strategy.

reference.

$$\frac{\partial \boldsymbol{v}}{\partial t} + \boldsymbol{v} \cdot \nabla \boldsymbol{v} + \frac{1}{\rho} \nabla p = \boldsymbol{g}, \qquad \nabla(\rho \boldsymbol{v}) = 0, \tag{C.1}$$

where $\boldsymbol{v} = (v_{x_I}, v_{y_I}, v_{z_I})$ is the absolute velocity of a liquid particle, $\boldsymbol{g}$ is the acceleration due to gravity, p is the static pressure, and ∇ is an operator of a divergence. Let $\sum_c$ denote a coordinate system that is moving along with the cup. Its displacement concerning $\sum_I$ is denoted by $(\delta_x, \delta_y, \delta_z)$. Then the $\sum_c$ coordinate system variables can be displayed

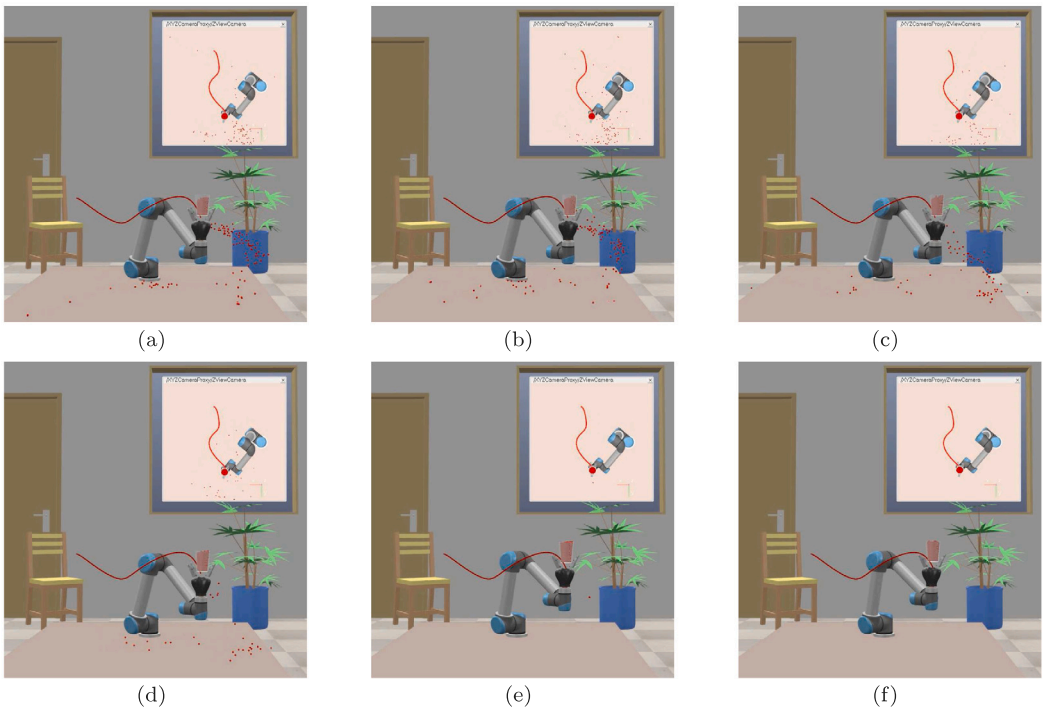


Fig. B.1. Simulation scenes of TOPP-RA and S-TOPP with different jerk limits. (a) TOPP-RA: no jerk limited. The maximum jerk value can reach 1624 rad/s^3, t_e = 1.0325 s. (b) S-TOPP with jerk limits 1000 rad/s^3, t_e = 1.0488 s. (c) S-TOPP with jerk limits 500 rad/s^3, t_e = 1.1041 s. (d) S-TOPP with jerk limits 300 rad/s^3, t_e = 1.1785 s. (e) S-TOPP with jerk limits 100 rad/s^3, t_e = 1.6745 s. (f) S-TOPP with jerk limits 50 rad/s^3, t_e = 2.3708 s. As the jerk limits decrease, the number of particles spilling out of the cup becomes smaller, which can be observed from the density of spilling particles on subfigures. When the jerk is limited to 50 rad/s^3, no any particle spills out of the cup.

concerning $\sum_I$ as below.

$$x_c = x_I + \delta_x, \quad y_c = y_I + \delta_y, \quad z_c = z_I + \delta_z \quad and \quad t_c = t, \tag{C.2}$$

$$v_{x_c} = v_{x_I} + \dot{\delta_x}, \quad v_{y_c} = v_{y_I} + \dot{\delta_y}, \quad v_{z_c} = v_{z_I} + \dot{\delta_z}. \tag{C.3}$$

The velocity vector $\boldsymbol{v_c} = (v_{x_c}, v_{y_c}, v_{z_c})$, is defined as contravariant velocity concerning $\sum_c$. Substituting Eq. (C.3) into Eq. (C.1) leads to the governing equations in $\sum_c$ as

$$\frac{\partial \boldsymbol{v}_c}{\partial t} + \boldsymbol{v}_c \cdot \nabla \boldsymbol{v}_c + \frac{1}{\rho}\nabla p = \boldsymbol{g}_c, \tag{C.4}$$

$$\nabla \cdot \boldsymbol{v}_c = 0, \tag{C.5}$$

where the body force term is given by

$$\boldsymbol{g}_c = \boldsymbol{g} + \ddot{\boldsymbol{\delta}}, \tag{C.6}$$

where $\boldsymbol{g}_c$ represents the net body force due to the cup acceleration plus the acceleration due to gravity. When the cup moves with acceleration $\boldsymbol{a} = \ddot{\boldsymbol{\delta}}$, which would be added to liquid particles with extra force in this term, causing the change of $\boldsymbol{v}_c$. From the analysis, we obtain a conclusion that the velocity of liquid particles is significantly affected by cup moving acceleration. Both smoother velocity curves and smaller jerk values contribute to smoother acceleration, indicating that the smoothness of velocity curves and the value of jerk affect liquid particle movement.

Appendix D. Additional real-world experiments

The optimized trajectory duration inevitably becomes longer when jerk limits are involved, affecting the intensity of liquid sloshing. To avoid the above concern, we conduct additional experiments. We configure S-TOPP and TOPP-RA velocity limits to find approximately equal optimized trajectory durations on the same path. DP3 method is omitted in this section because the overwhelming computation time hinders the searching process of the approximately equal optimized trajectory duration. Table D.1 presents the limits and optimized trajectory duration for different paths and methods. And Table D.2 records the execution results. Compared with Section 4.3, H_{LA} of TOPP-RA is the same for Path 1 and Path 2, and the volume of liquid lost is smaller for Path 3 and Path 4. Nevertheless, TOPP-RA method still fails to accomplish the tasks. The details of these experiments can also be found in the experimental video.

Table D.1
Limits of different paths for TOPP-RA.

Path	$\dot{q}$ limits (rad/s)	$\ddot{q}$ limits (rad/s^2)	$\dddot{q}$ limits (rad/s^3)	t_e (s)
1	0.910	10	∞	1.9437
2	0.845	10	∞	2.0051
3	0.884	10	∞	1.9979
4	0.905	10	∞	2.6095

Table D.2
Case scenarios and results for TOPP-RA. H_{LA}^1 denotes H_{LA} in Section 4.3.1 and H_{LA}^2 denotes H_{LA} in current Section.

Path	Cup	Results	H_{LA}^1 vs. H_{LA}^2 (cm)
1	1	Fail	4.4 vs. 4.4
2	2	Fail	7.7 vs. 7.7
3	3	Fail	8.7 vs. 8.9
4	2	Fail	7.1 vs. 7.3

Appendix E. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.robot.2023.104530.

References

[1] J. Bobrow, S. Dubowsky, J. Gibson, Time-optimal control of robotic manipulators along specified paths, Int. J. Robot. Res. 4 (3) (1985) 3–17, http://dx.doi.org/10.1177/027836498500400301.

[2] K. Shin, N. McKay, Minimum-time control of robotic manipulators with geometric path constraints, IEEE Trans. Automat. Control 30 (6) (1985) 531–541, http://dx.doi.org/10.1109/TAC.1985.1104009.

[3] S. Macfarlane, E. Croft, Jerk-bounded manipulator trajectory planning: design for real-time applications, IEEE Trans. Robot. Autom. 19 (1) (2003) 42–52, http://dx.doi.org/10.1109/TRA.2002.807548.

[4] M. Arsenault, L.F. Tremblay, M. Zeinali, Optimization of trajectory durations based on flow rate scaling for a 4-DoF semi-automated hydraulic rock-reaker, Mech. Mach. Theory 143 (2020) 103632, http://dx.doi.org/10.1016/j.mechmachtheory.2019.103632.

[5] C.H. Wang, J.G. Horng, Constrained minimum-time path planning for robot manipulators via virtual knots of the cubic B-spline functions, IEEE Trans. Automat. Control 35 (5) (1990) 573–577, http://dx.doi.org/10.1109/9.53526.

[6] X. Du, J. Huang, L.M. Zhu, H. Ding, An error-bounded B-spline curve approximation scheme using dominant points for CNC interpolation of micro-line toolpath, Robot. Comput.-Integr. Manuf. 64 (2020) 101930, http://dx.doi.org/10.1016/j.rcim.2019.101930.

[7] S. Qian, K. Bao, B. Zi, W.D. Zhu, Dynamic trajectory planning for a three degrees-of-freedom cable-driven parallel robot using quintic B-splines, J. Mech. Des. 142 (7) (2020) http://dx.doi.org/10.1115/1.4045723.

[8] C. Lin, P. Chang, J. Luh, Formulation and optimization of cubic polynomial joint trajectories for industrial robots, IEEE Trans. Automat. Control 28 (12) (1983) 1066–1074, http://dx.doi.org/10.1109/TAC.1983.1103181.

[9] A. Gasparetto, V. Zanotto, A technique for time-jerk optimal planning of robot trajectories, Robot. Comput.-Integr. Manuf. 24 (3) (2008) 415–426, http://dx.doi.org/10.1016/j.rcim.2007.04.001.

[10] A. Gasparetto, A. Lanzutti, R. Vidoni, V. Zanotto, Experimental validation and comparative analysis of optimal time-jerk algorithms for trajectory planning, Robot. Comput.-Integr. Manuf. 28 (2) (2012) 164–181, http://dx.doi.org/10.1016/j.rcim.2011.08.003.

[11] Z. Jia, D. Song, J. Ma, G. Hu, W. Su, A NURBS interpolator with constant speed at feedrate-sensitive regions under drive and contour-error constraints, Int. J. Mach. Tools Manuf. 116 (2017) 1–17, http://dx.doi.org/10.1016/j.ijmachtools.2016.12.007.

[12] H. Wang, H. Wang, J. Huang, B. Zhao, L. Quan, Smooth point-to-point trajectory planning for industrial robots with kinematical constraints based on high-order polynomial curve, Mech. Mach. Theory 139 (2019) 284–293, http://dx.doi.org/10.1016/j.mechmachtheory.2019.05.002.

[13] D. Lee, C.W. Ha, Optimization process for polynomial motion profiles to achieve fast movement with low vibration, IEEE Trans. Control Syst. Technol. 28 (5) (2020) 1892–1901, http://dx.doi.org/10.1109/TCST.2020.2998094.

[14] Y. Fang, J. Qi, J. Hu, W. Wang, Y. Peng, An approach for jerk-continuous trajectory generation of robotic manipulators with kinematical constraints, Mech. Mach. Theory 153 (2020) 103957, http://dx.doi.org/10.1016/j.mechmachtheory.2020.103957.

[15] H. Pham, Q.C. Pham, A new approach to time-optimal path parameterization based on reachability analysis, IEEE Trans. Robot. 34 (3) (2018) 645–659, http://dx.doi.org/10.1109/TRO.2018.2819195.

[16] B. Sencer, Y. Altintas, E. Croft, Feed optimization for five-axis CNC machine tools with drive constraints, Int. J. Mach. Tools Manuf. 48 (7) (2008) 733–745, http://dx.doi.org/10.1016/j.ijmachtools.2008.01.002.

[17] F. Yuan, D. Chen, C. Pan, Application of optimal-jerk trajectory planning in gait-balance training robot, Chin. J. Mech. Eng. 35 (2022) http://dx.doi.org/10.1186/s10033-021-00665-1.

[18] D. Kaserer, H. Gattringer, A. Müller, Online robot-object synchronization with geometric constraints and limits on velocity, acceleration, and jerk, IEEE Robot. Autom. Lett. 3 (4) (2018) 3169–3176, http://dx.doi.org/10.1109/LRA.2018.2849827.

[19] K. Erkorkmaz, Q.G.C. Chen, M.Y. Zhao, X. Beudaert, X.S. Gao, Linear programming and windowing based feedrate optimization for spline toolpaths, CIRP Ann-Manuf. Technol. 66 (1) (2017) 393–396, http://dx.doi.org/10.1016/j.cirp.2017.04.058.

[20] W. Fan, X.S. Gao, C.H. Lee, K. Zhang, Q. Zhang, Time-optimal interpolation for five-axis CNC machining along parametric tool path based on linear programming, Int. J. Adv. Manuf. Technol. 69 (2013) 1373–1388, http://dx.doi.org/10.1007/s00170-013-5083-x.

[21] F. Debrouwere, V.L. Wannes, G. Pipeleers, Q.T. Dinh, M. Diehl, D.S. Joris, J. Swevers, Time-optimal path following for robots with convex–concave constraints using sequential convex programming, IEEE Trans. Robot. 29 (6) (2013) 1485–1495, http://dx.doi.org/10.1109/TRO.2013.2277565.

[22] L. Consolini, M. Locatelli, A. Minari, A sequential algorithm for jerk limited speed planning, IEEE Trans. Autom. Sci. Eng. (2021) 1–18, http://dx.doi.org/10.1109/TASE.2021.3111758.

[23] D. Kaserer, H. Gattringer, A. Müller, Nearly optimal path following with jerk and torque rate limits using dynamic programming, IEEE Trans. Robot. 35 (2) (2019) 521–528, http://dx.doi.org/10.1109/TRO.2018.2880120.

[24] Y. Cheng, C. Li, S. Li, Z. Li, Motion planning of redundant manipulator with variable joint velocity limit based on beetle antennae search algorithm, IEEE Access 8 (2020) 138788–138799, http://dx.doi.org/10.1109/ACCESS.2020.3012564.

[25] Y. Pei, Z. Liu, J. Xu, C. Yang, Minimum-time trajectory planning for a 4-dof manipulator considering motion stability and obstacle avoidance, in: 2020 5th International Conference on Mechanical, Control and Computer Engineering, ICMCCE, 2020, pp. 207–211, http://dx.doi.org/10.1109/ICMCCE51767.2020.00053.

[26] F. Wang, Z. Wu, T. Bao, Time-jerk optimal trajectory planning of industrial robots based on a hybrid WOA-GA algorithm, Processes 10 (5) (2022) http://dx.doi.org/10.3390/pr10051014.

[27] H. Pham, Q.C. Pham, On the structure of the time-optimal path parameterization problem with third-order constraints, in: 2017 IEEE International Conference on Robotics and Automation, ICRA, 2017, pp. 679–686, http://dx.doi.org/10.1109/ICRA.2017.7989084.

[28] J.W. Ma, S. Gao, H.T. Yan, Q. Lv, G. q. Hu, A new approach to time-optimal trajectory planning with torque and jerk limits for robot, Robot. Auton. Syst. 140 (2021) 103744, http://dx.doi.org/10.1016/j.robot.2021.103744.

[29] S. Karaman, E. Frazzoli, Incremental sampling-based algorithms for optimal motion planning, in: Y. Matsuoka, D. H., J. Neira (Eds.), Robotics: Science and Systems VI, 2011, pp. 267–274.

[30] Y. Li, Z. Littlefield, K.E. Bekris, Asymptotically optimal sampling-based kinodynamic planning, Int. J. Robot. Res. 35 (5) (2016) 528–564, http://dx.doi.org/10.1177/0278364915614386.

[31] R. Arteaga, E. Antonio, I. Becerra, R. Murrieta-Cid, On the efficiency of the SST planner to find time optimal trajectories among obstacles with a DDR under second order dynamics, IEEE Robot. Autom. Lett. 7 (2) (2022) 674–681, http://dx.doi.org/10.1109/LRA.2021.3132923.

[32] B. Francesco, B. Alberto, M. Manfred, Predictive Control for Linear and Hybrid Systems, Cambridge University Press, 2017.

[33] L. Janson, E. Schmerling, A. Clark, M. Pavone, Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions, Int. J. Robot. Res. 34 (7) (2015) 883–921, http://dx.doi.org/10.1177/0278364915577958.

Huanhuan Huang is a master candidate at Tsinghua University. Her research interests include issues related to time-optimal trajectory planning and motion planning under dynamic environment for robotic manipulation.

Houde Liu received the Ph.D. degree from Harbin Institute of Technology. He is currently a research associate in Tsinghua University. His research interests include intelligent robot, robot planning and control technology, and autonomous manipulation, intelligent equipment.

Chongkun Xia received the Ph.D. degree from Northeastern University. He is currently working as an assistant professor in School of Advanced Manufacturing, Sun Yat-sen University. His research interests include intelligent robot, motion planning, visual tactile sensor, and autonomous manipulation.

Hongwei Mei received the Ph.D. degree from Harbin Institute of Technology. He is currently an associated professor in Tsinghua University. His research interests include insulation technology and state detection of power transmission and intelligent robot.

Xuehai Gao received the Ph.D. degree from Harbin Institute of Technology. His research interests include computer simulation, vision measurement, guidance, navigation and control of intelligent robot.

Bin Liang received the Ph.D. degree from Tsinghua University. He is currently a professor in TsingHua University. His research interests include intelligent robot and teleoperation, intelligent sensing technology, and space intelligent systems.