2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Time-Optimal Path Parameterization for Cooperative Multi-Arm Robotic Systems with Third-Order Constraints

Maximilian Dio, Knut Graichen, Andreas Völz

***Abstract*— This paper presents a time-optimal path parameterization (TOPP) method for cooperative multi-arm robotic systems (MARS) manipulating heavy objects with third-order constraints that include jerk, torque rate and wrench rate limits. The method is based on a problem reformulation as a sequential linear program and provides a unified planning approach that is faster than previous convex optimization techniques. The equivalence to a reachability-based TOPP is shown and simulation results for a cooperative MARS consisting of two 7 degree of freedom (DOF) robots and a tightly grasped object with 6 DOFs are provided.**

I. Introduction

Collaborative manipulators operate safely alongside humans, eliminating the need for safety fences. However, they have lower payload capacities compared to larger industrial manipulators [1]. To address this limitation, MARS have gained interest for their compactness, collaboration, and increased payload capacity. Nonetheless, a unified control and motion planning methodology is still missing that addresses the kinematic constraints and redundant actuation. While improper control can lead to stress-induced damage to the workpiece or robots [2], well-planned motion is crucial to enhance the overall performance.

Motion planning is typically divided into direct and decoupled trajectory planning [3]. The former calculates both the path and its timing simultaneously for optimal motion. The latter optimizes the path initially and then computes the timing afterward. This simplifies the problem because kinematics and dynamics are formulated as a function of a scalar path parameter, leading to faster computation. In this work, the traversal time is minimized resulting in a TOPP. In addition to redundant actuation inherent to MARS, constraints up to third order have to be considered, i.e. jerk, torque rate, and wrench rate limits. Ignoring them results in discontinuous joint torques, which can lead to oscillations on real systems [4].

In numerical integration (NI) methods the trajectory is computed by integrating with the maximum and minimum path acceleration from the boundary states [5], [6]. While this method is fast as it exploits the bang-bang structure of the problem, it is challenging to implement and prone to failure as switching points have to be identified [7]. In [8] redundantly actuated systems are incorporated by solving additional linear programs (LPs) for each integration step as the maximal accelerations have to be found numerically. The authors of [9] use polygon projection techniques proposed by [10] instead. In [11] saturation patterns are exploited, as they reduce the necessity to optimize the path accelerations in each integration step. Adapting the NI framework to third-order constraints is proposed in [12] for non-redundantly actuated systems. The problem structure becomes more complex as the phase plane expands to three dimensions, which complicates the bridging of maximized jerk profiles.

By reformulating the problem in the path domain, a convex optimal control problem (OCP) is obtained [13]. Direct transcription and a reformulation as a second-order cone program (SOCP) allow to solve the resulting convex optimization (CO) problem with existing solvers in a reasonable time frame. Instead of solving the SOCP, [10] sequentially solves a linearized version of the problem with a continuously decreasing trust region. This method is faster than the SOCP but still requires multiple iterations to obtain the same solution. In [14], a different constraint discretization compared to [13] is proposed that should solve the original problem with only one LP. However, second-order constraints can not always be enforced as noted by [15]. In [16] redundant actuation is added in the CO formulation and jerk constraints are incorporated in [17] for single-arm robots. Because third-order constraints are nonconvex, the problem is convexified and solved sequentially.

An approach based on reachability analysis (RA) is introduced by [7] which can be implemented easily and is more robust than NI as no switching point search is required. It is extendable to redundantly actuated systems but so far limited to second-order constraints only. Compared to CO methods, TOPP-RA is superior in terms of computational speed and is proven to converge asymptotically to the same solution [18].

In [19], the authors propose a dynamic programming (DP) method, discretizing both the path and velocity domains. This approach is used by [20] for cooperative MARS with jerk limits. Despite coarse discretization, computational times remain very long compared to other approaches.

The literature review shows that efficient methods for solving the TOPP problem for cooperative MARS with third-order constraints including torque rate are still missing. This paper proposes a new LP-based approach with a tailored constraint discretization that solves the problem and ensures smooth trajectories. The presented method is proven to be correct and is faster than previous CO or DP methods. The remaining paper is organized as follows. Section II formulates the general problem. The solution method and its correctness are presented in Section III. Simulation results for a cooperative MARS consisting of two 7-DOF robots and a tightly grasped 6-DOF object are provided in Section IV. Section V concludes the paper with remarks on future work.

Chair of Automatic Control, Friedrich-Alexander-Universität Erlangen-Nürnberg, Email: {maximilian.dio, knut.graichen, andreas.voelz}@fau.de
Acknowledgements This work was funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) – 493539176

II. Problem description

This work builds on the decoupled motion planning approach. Similar to [21], a smooth object path

$$\boldsymbol{T}(s) = \exp([\boldsymbol{\Psi}(s)]_\times) \in SE(3) \tag{1}$$

expressed in the inertial frame is parameterized over $s \in [0,1]$ with the Lie algebra $[\boldsymbol{\Psi}(s)]_\times \in \mathbb{R}^{4\times4}$ of the pose $\boldsymbol{T}(s)$ and its isomorphism $\boldsymbol{\Psi}(s) \in \mathbb{R}^6$. The right invariant twist

$$[\boldsymbol{\psi}(s)]_\times = \frac{\mathrm{d}\boldsymbol{T}(s)}{\mathrm{d}s}\boldsymbol{T}^{-1} \tag{2}$$

is used as the path derivative. From hereon after, all Cartesian quantities are expressed in the inertial frame if not defined differently.

The manipulators grasp the object at fixed grasp frames $\boldsymbol{T}_{\mathrm{des}}(s)$. A consistent joint path $\boldsymbol{q}(s)$ is ensured such that the pose of the end effector matches the desired pose of the grasp frames, i.e. $\boldsymbol{T}_{\mathrm{EE}}(\boldsymbol{q}(s)) = \boldsymbol{T}_{\mathrm{des}}(s)\ \forall s \in [0,1]$. The availability of an analytical inverse kinematics (IK) algorithm similar to [22] is assumed. Additionally, the path derivatives $(\cdot)' = \frac{\partial(\cdot)}{\partial s}$ up to third order, i.e.

$$\frac{\partial^k\, {}^s\boldsymbol{\nu}_{\mathrm{EE}}(\boldsymbol{q}(s))}{\partial s^k} = \frac{\partial^k\, {}^s\boldsymbol{\nu}_{\mathrm{des}}(s)}{\partial s^k}\,, k = 0,1,2 \tag{3}$$

for all $s \in [0,1]$ have to match. For simplicity, the pose and joint derivatives are computed by central differences, given the possibility to compute a smooth joint path $\boldsymbol{q}(s)$ analytically. The assumption is made that a planning algorithm has addressed all geometric constraints so that the resulting joint path is smooth enough and completely defined by one scalar path parameter. This work only focuses on the trajectory generation. For this, the kinematics and dynamics are required and are described in the following omitting the explicit dependency on the path parameter s.

A. Kinematics and Dynamics of Serial Link Robots

The joint configuration for a serial link robot is given by $\boldsymbol{q}$. Its time derivatives $\dot{(\cdot)} = \frac{\mathrm{d}(\cdot)}{\mathrm{d}t}$ are computed by

$$\dot{\boldsymbol{q}} = \boldsymbol{q}'\dot{s}\,, \tag{4a}$$
$$\ddot{\boldsymbol{q}} = \boldsymbol{q}'\ddot{s} + \boldsymbol{q}''\dot{s}^2\,, \tag{4b}$$
$$\dddot{\boldsymbol{q}} = \dot{s}\left(\boldsymbol{q}'\dddot{s}' + 3\boldsymbol{q}''\ddot{s} + \boldsymbol{q}'''\dot{s}^2\right)\,, \tag{4c}$$

with $\dddot{s} = \dot{s}\ddot{s}'$ [17]. Joint torques $\boldsymbol{\tau}$ are computed by

$$\boldsymbol{\tau} = \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{g}(\boldsymbol{q}) + \boldsymbol{J}(\boldsymbol{q})^\top\boldsymbol{h} \tag{5}$$

using the rigid body dynamics for a serial link robot with an external wrench vector $\boldsymbol{h}$ acting on the robot. The generalized mass matrix is defined by $\boldsymbol{M}$, Coriolis and centrifugal matrix by $\boldsymbol{C}$ and gravitational forces by $\boldsymbol{g}$. The wrench is transformed via the transposed Jacobian $\boldsymbol{J}$. Inserting the kinematics (4) results in

$$\boldsymbol{\tau} = \underbrace{\boldsymbol{M}\boldsymbol{q}'}_{\boldsymbol{\tau}_m}\ddot{s} + \underbrace{(\boldsymbol{C}\boldsymbol{q}' + \boldsymbol{M}\boldsymbol{q}'')}_{\boldsymbol{\tau}_c}\dot{s}^2 + \underbrace{\boldsymbol{g}}_{\boldsymbol{\tau}_g} + \boldsymbol{J}^\top\boldsymbol{h}\,. \tag{6}$$

Time differentiation of (6) results in the torque rate

$$\begin{aligned}\dot{\boldsymbol{\tau}} = \dot{s}(&\boldsymbol{\tau}_m\dddot{s}' + (\boldsymbol{\tau}_m' + 2\boldsymbol{\tau}_c)\ddot{s} + \boldsymbol{\tau}_c'\dot{s}^2 + \boldsymbol{\tau}_g' \\ &+ \boldsymbol{J}^\top\boldsymbol{h}' + \boldsymbol{J}'^\top\boldsymbol{h})\,.\end{aligned} \tag{7}$$

B. Kinematics and Dynamics of the Rigid Body in $SE(3)$

The time-dependent twist $\boldsymbol{\nu}^\top = [\boldsymbol{v}^\top\ \ \boldsymbol{\omega}^\top] \in \mathbb{R}^6$ as well as its higher order time derivatives can be computed by

$$\boldsymbol{\nu} = \boldsymbol{\psi}\dot{s}\,, \tag{8a}$$
$$\dot{\boldsymbol{\nu}} = \boldsymbol{\psi}\ddot{s} + \boldsymbol{\psi}'\dot{s}^2\,, \tag{8b}$$
$$\ddot{\boldsymbol{\nu}} = \dot{s}\left(\boldsymbol{\psi}\dddot{s}' + 3\boldsymbol{\psi}'\ddot{s} + \boldsymbol{\psi}''\dot{s}^2\right)\,. \tag{8c}$$

The dynamics of the object are given by

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \left[m\boldsymbol{g}^\top \quad (\boldsymbol{\omega}\times\boldsymbol{\Theta}\boldsymbol{\omega})^\top\right]^\top - \boldsymbol{G}\boldsymbol{h} = \boldsymbol{0}\,, \tag{9}$$

where $\boldsymbol{M} = \mathrm{diag}(m\boldsymbol{I}, \boldsymbol{\Theta})$ is the mass matrix with a pose dependent inertia tensor $\boldsymbol{\Theta}$. The grasp-matrix $\boldsymbol{G}$ projects interaction wrenches to the center of mass (COM). In total P rigid grasp frames are considered, where each frame is located at $\boldsymbol{r}_i \in \mathbb{R}^3,\ i = 0,\dots,P$ in relation to the COM. The grasp matrix is computed by

$$\boldsymbol{G} = \begin{bmatrix}\boldsymbol{I} & -[\boldsymbol{r}_0]_\times & \dots & \boldsymbol{I} & -[\boldsymbol{r}_P]_\times \\ \boldsymbol{0} & \boldsymbol{I} & \dots & \boldsymbol{0} & \boldsymbol{I}\end{bmatrix}\,, \tag{10}$$

where $[\boldsymbol{r}_i]_\times \in \mathbb{R}^{3\times3}$ is the skew-symmetric matrix of $\boldsymbol{r}_i$ such that $[\boldsymbol{a}]_\times\boldsymbol{b} = \boldsymbol{a}\times\boldsymbol{b}$, which is the isomorphism in $\mathfrak{so}(3)$.

Evaluating the kinematics in (9) results in $\boldsymbol{h}_m\ddot{s} + \boldsymbol{h}_c\dot{s}^2 + \boldsymbol{h}_g - \boldsymbol{G}\boldsymbol{h} = \boldsymbol{0}$, with $\boldsymbol{h}_m = \boldsymbol{M}\boldsymbol{\psi}$, $\boldsymbol{h}_g^\top = \left[m\boldsymbol{g}^\top \quad \boldsymbol{0}^\top\right]$ and $\boldsymbol{h}_c = \boldsymbol{M}\boldsymbol{\psi}' + \left[\boldsymbol{0}^\top \quad (\boldsymbol{\psi}_\omega\times\boldsymbol{\Theta}\boldsymbol{\psi}_\omega)^\top\right]^\top$. The angular part of $\boldsymbol{\psi}$ is $\boldsymbol{\psi}_\omega$. An additional time differentiation results in

$$\dot{s}(\boldsymbol{h}_m\dddot{s}' + (\boldsymbol{h}_m' + 2\boldsymbol{h}_c)\ddot{s} + \boldsymbol{h}_c'\dot{s}^2 - \boldsymbol{G}\boldsymbol{h}' - \boldsymbol{G}'\boldsymbol{h}) = \boldsymbol{0}\,. \tag{11}$$

As noted in [2], the grasp matrix lacks full column rank resulting in a non-empty nullspace that is spanned by $\boldsymbol{\mathcal{N}}$. The nullspace of $\boldsymbol{G}$ does not result in any motion of the object and can therefore be used to constrain the interaction wrenches by limiting $\tilde{\boldsymbol{h}} = \boldsymbol{\mathcal{N}}\boldsymbol{h}$. Here, the projected wrenches are represented in the body frame. Time derivatives of the wrenches are required in the computation of the torque rate. To ensure a bounded problem, the wrench derivatives need to be limited too. One possibility is to constrain the time derivative of the projected wrenches which is given by $\dot{\tilde{\boldsymbol{h}}} = \dot{s}(\boldsymbol{\mathcal{N}}\boldsymbol{h}' + \boldsymbol{\mathcal{N}}'\boldsymbol{h})$.

C. Unified Constraint Structure

The kinematics and dynamics of the robot and the rigid body have the same structure and can be brought into a unified form. First-order constraints, i.e. Cartesian or joint velocity limits can be represented by $\underline{\boldsymbol{x}}_1 \le \boldsymbol{a}_1\dot{s} \le \bar{\boldsymbol{x}}_1$. The constraints can be equivalently expressed as

$$0 \le \dot{s}^2 \le \min_i\left(\max\left(\frac{[\underline{\boldsymbol{x}}_1]_i}{[\boldsymbol{a}_1]_i}, \frac{[\bar{\boldsymbol{x}}_1]_i}{[\boldsymbol{a}_1]_i}\right)\right)^2\,, \tag{12}$$

assuming that the elements of the vector $[\underline{\boldsymbol{x}}_1]_i \le 0\,\forall\, i$ and $[\bar{\boldsymbol{x}}_1]_i \ge 0\,\forall\, i$. Second-order constraints can be represented by $\underline{\boldsymbol{x}}_2 \le \boldsymbol{a}_2\ddot{s} + \boldsymbol{b}_2\dot{s}^2 + \boldsymbol{F}_2\boldsymbol{h} \le \bar{\boldsymbol{x}}_2$ while third-order constraints are given by $\underline{\boldsymbol{x}}_3 \le\ \dot{s}(\boldsymbol{a}_3\dddot{s}' + \boldsymbol{b}_3\ddot{s} + \boldsymbol{c}_3\dot{s}^2 + \boldsymbol{d}_3 + \boldsymbol{F}_3\boldsymbol{h} + \boldsymbol{E}_3\boldsymbol{h}') \le \bar{\boldsymbol{x}}_3$. Combining all c constraints, the admissible set is then defined by

$$\Omega(s,\dot{s}) = \left\{\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{u}\end{bmatrix}\ \middle|\ \boldsymbol{A}(s)\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{u}\end{bmatrix} \le \boldsymbol{b}(s) + \frac{\boldsymbol{c}(s)}{\dot{s}}\right\}\,, \tag{13}$$

with $\boldsymbol{x} = \begin{bmatrix}\dot{s}^2 & \ddot{s} & \boldsymbol{h}^\top\end{bmatrix}^\top \in \mathbb{R}^n$, $\boldsymbol{u} = \begin{bmatrix}\ddot{s}' & \boldsymbol{h}'^\top\end{bmatrix}^\top \in \mathbb{R}^m$ and $\boldsymbol{A}(s) = \begin{bmatrix}\boldsymbol{A}_{\boldsymbol{x}}(s) & \boldsymbol{A}_{\boldsymbol{u}}(s)\end{bmatrix} \in \mathbb{R}^{c\times(n+m)}$, assuming $\dot{s} > 0$.

For first- and second-order constraints, the elements in $\boldsymbol{c}(s)$ are zero resulting in a linear set with respect to $\boldsymbol{x}$ and $\boldsymbol{u}$. For third-order constraints, the elements in $\boldsymbol{c}(s)$ are generally positive and non-zero. With $[\boldsymbol{x}]_1 = \dot{s}^2$, $[\boldsymbol{c}(s)]_i/\dot{s}$ is convex for all i and the set can become nonconvex.

III. TOPP-LP Method

The resulting TOPP problem is obtained by minimizing $T = \int_0^T 1 \,\mathrm{d}t$ over T and $\boldsymbol{u}(\cdot)$ subject to boundary conditions, the admissible set $\Omega(s, \dot{s})$ and the system dynamics of the path and the wrenches. To solve the problem efficiently, a reformulation of the problem as a single LP is proposed. The resulting method is referred to as TOPP-LP.

A. Derivation

1) Convex Reformulation: We extend the convex reformulation proposed by [13], [17] to redundantly actuated systems with third-order constraints. The OCP in the time domain is reformulated in the path domain by replacing $\mathrm{d}t = \frac{1}{\dot{s}}\mathrm{d}s$, $\forall \dot{s} \neq 0$. Note that a singularity in $\dot{s} = 0$ exists. This results in the following convex but nonlinear cost $\int_0^1 \frac{1}{\dot{s}}\mathrm{d}s$. Acknowledging that $\frac{\mathrm{d}\dot{s}^2}{\mathrm{d}s} = 2\ddot{s}$, it becomes possible to construct a linear dynamical system in the path domain for the pseudo path velocity $\dot{s}^2$, the path acceleration $\ddot{s}$, and the wrench $\boldsymbol{h}$, which results in

$$\underbrace{\frac{\mathrm{d}}{\mathrm{d}s}\begin{bmatrix}\dot{s}^2\\ \ddot{s}\\ \boldsymbol{h}\end{bmatrix}}_{\boldsymbol{x}'(s)} = \underbrace{\begin{bmatrix}0 & 2 & 0\\ 0 & 0 & 0\\ \boldsymbol{0} & \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}}_{\boldsymbol{\Phi}_c}\underbrace{\begin{bmatrix}\dot{s}^2\\ \ddot{s}\\ \boldsymbol{h}\end{bmatrix}}_{\boldsymbol{x}(s)} + \underbrace{\begin{bmatrix}0 & 0\\ 1 & 0\\ \boldsymbol{0} & \boldsymbol{I}\end{bmatrix}}_{\boldsymbol{\Gamma}_c}\underbrace{\begin{bmatrix}\ddot{s}'\\ \boldsymbol{h}'\end{bmatrix}}_{\boldsymbol{u}(s)} \tag{14}$$

with $\ddot{s}' = \frac{1}{\dot{s}}\dddot{s}$ and $\boldsymbol{h}' = \frac{1}{\dot{s}}\dot{\boldsymbol{h}}$.

To obtain a convex OCP, the constraints (13) are linearized along a given path velocity $\sqrt{[\boldsymbol{x}]_1} \to z$ similar to [17]. The nonconvex set (13) is then approximated via a first-order Taylor expansion. Because $c/\sqrt{[\boldsymbol{x}]_1}$ is convex, $c/\sqrt{[\boldsymbol{x}]_1} \approx \frac{3}{2}\boldsymbol{c}z^{-1} - \frac{1}{2}\boldsymbol{c}z^{-3}[\boldsymbol{x}]_1 \leq c/\sqrt{[\boldsymbol{x}]_1}$ holds true for all $(\boldsymbol{x}, \boldsymbol{u})$. Consequently, the nonconvex set is underapproximated by the linear set

$$\begin{aligned} T_\Omega(s|z) &= \left\{ \begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{u}\end{bmatrix} \;\middle|\; \tilde{\boldsymbol{A}}(s|z)\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{u}\end{bmatrix} \leq \tilde{\boldsymbol{b}}(s|z) \right\} \\ &= \left\{ \begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{u}\end{bmatrix} \;\middle|\; \left(\boldsymbol{A}(s) + \begin{bmatrix}\frac{1}{2}z^{-3}\boldsymbol{c}(s) & \boldsymbol{0}\end{bmatrix}\right)\begin{bmatrix}\boldsymbol{x}\\ \boldsymbol{u}\end{bmatrix} \right. \\ &\qquad \left. \leq \boldsymbol{b}(s) + \frac{3}{2}z^{-1}\boldsymbol{c}(s) \right\} \subseteq \Omega(s, \dot{s})\,. \end{aligned} \tag{15}$$

The resulting OCP formulated over the path domain is convex and given by

$$\min_{\boldsymbol{u}(\cdot)} \quad \int_0^1 \frac{1}{\sqrt{[\boldsymbol{x}]_1}}\mathrm{d}s \tag{16a}$$

$$\text{s.t.} \quad \boldsymbol{x}(0) = \boldsymbol{x}_0, \quad \boldsymbol{x}(1) = \boldsymbol{x}_T \tag{16b}$$

$$\boldsymbol{x}'(s) = \boldsymbol{\Phi}_c\boldsymbol{x}(s) + \boldsymbol{\Gamma}_c\boldsymbol{u}(s) \tag{16c}$$

$$(\boldsymbol{x}(s), \boldsymbol{u}(s)) \in T_\Omega(s|z) \quad \text{for} \quad s \in [0, 1]\,. \tag{16d}$$

To obtain faster computation times compared to the SOCP by [13], the nonlinear cost function (16a) is replaced by a linear cost function $\int_0^1 -[\boldsymbol{x}(s)]_1\mathrm{d}s$. This is motivated by the fact that the resulting path velocity must be maximized over the complete path to minimize the trajectory duration. Similar to previous CO-based methods, the continuous OCP (16) is discretized over N grid points with step length Δ. The discrete dynamics are computed by exact integration of the linear dynamics (14) resulting in $\boldsymbol{x}_{k+1} = \boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{u}_k$ with $\boldsymbol{\Phi} = e^{\Delta\boldsymbol{\Phi}_c} = \boldsymbol{I} + \Delta\boldsymbol{\Phi}_c$ and $\boldsymbol{\Gamma} = \int_0^\Delta e^{s\boldsymbol{\Phi}_c}\boldsymbol{\Gamma}_c\mathrm{d}s = \Delta\boldsymbol{\Gamma}_c + \frac{1}{2}\Delta^2\boldsymbol{\Phi}_c\boldsymbol{\Gamma}_c$.

As noted by [14], the SOCP solution does not match the LP solution. However, in Section III-B, it is shown that the proposed LP solution converges to the same solution as the SOCP solution with decreasing step size.

2) Constraint Discretization: CO solutions suffer from oscillations close to zero-inertia points, i.e. points where elements of the input constraints $[\boldsymbol{A}_{\boldsymbol{u}}]_{i,1}$ are zero. To mitigate this, regularization terms have to be added in (16a) [13]. To maintain an LP formulation, a special constraint discretization is employed instead. In [7] a constraint discretization was proposed that ensures that the input at a given stage is also valid at a preceding stage. We extend this idea to the case of third-order constraints.

Adding third-order constraints increases the dynamics order compared to a TOPP problem with only second-order constraints. Because the pseudo jerk inputs are piecewise constant, the pseudo velocity is once continuously differentiable $\left(\dot{s}^2(s) \in \mathcal{C}^1\right)$ and fully defined by a parabola over one step length Δ. When constraints are enforced at the grid points, as indicated by the hatched area in Fig. 1, different accelerations can yield identical velocities at these points. This leads to a singular arc and oscillations in the acceleration, as depicted by the blue curve. To maintain a smooth trajectory, constraints must also be enforced at intermediate grid points shown by the red curve. This ensures that both acceleration and jerk do not oscillate. The resulting discretization scheme for one stage k is given by

$$\hat{\Omega}_k = \left\{ \boldsymbol{x}_k, \boldsymbol{u}_k \;\middle|\; \hat{\boldsymbol{A}}_k\begin{bmatrix}\boldsymbol{x}_k\\ \boldsymbol{u}_k\end{bmatrix} \leq \hat{\boldsymbol{b}}_k \right\}, \tag{17}$$

with

$$\begin{aligned} \hat{\boldsymbol{A}}_k &= \begin{bmatrix} \tilde{\boldsymbol{A}}_{\boldsymbol{x},k} & \tilde{\boldsymbol{A}}_{\boldsymbol{u},k} \\ \tilde{\boldsymbol{A}}_{\boldsymbol{x},k+\frac{1}{2}}\boldsymbol{\Phi}_{\frac{1}{2}} & \tilde{\boldsymbol{A}}_{\boldsymbol{u},k+\frac{1}{2}} + \tilde{\boldsymbol{A}}_{\boldsymbol{x},k+\frac{1}{2}}\boldsymbol{\Gamma}_{\frac{1}{2}} \\ \tilde{\boldsymbol{A}}_{\boldsymbol{x},k+1}\boldsymbol{\Phi} & \tilde{\boldsymbol{A}}_{\boldsymbol{u},k+1} + \tilde{\boldsymbol{A}}_{\boldsymbol{x},k+1}\boldsymbol{\Gamma} \end{bmatrix}, \\ \hat{\boldsymbol{b}}_k &= \begin{bmatrix}\tilde{\boldsymbol{b}}_k^\top & \tilde{\boldsymbol{b}}_{k+\frac{1}{2}}^\top & \tilde{\boldsymbol{b}}_{k+1}^\top\end{bmatrix}^\top \end{aligned} \tag{18}$$

for all $k = 0, ..., N-1$, where the matrices $\boldsymbol{\Phi}_{1/2}$ and $\boldsymbol{\Gamma}_{1/2}$ are computed with 0.5Δ instead of Δ.

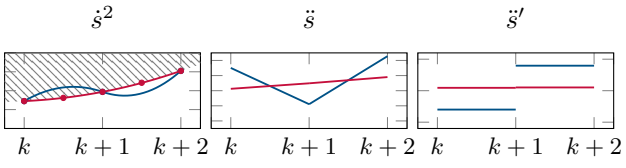

Fig. 1: A singular arc for TOPP with third-order constraints.

3) LP Formulation: The resulting linear optimization problem is given by

$$\max_{\substack{\boldsymbol{x}_{1:N-1},\\ \boldsymbol{u}_{0:N-1}}} \quad \sum_{k=1}^{N-1} [\boldsymbol{x}_k]_1 \tag{19a}$$

$$\text{s.t.} \quad \boldsymbol{x}_0\,, \boldsymbol{x}_N \text{ given} \tag{19b}$$

$$\boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{u}_k = \boldsymbol{x}_{k+1} \tag{19c}$$

$$(\boldsymbol{x}_k, \boldsymbol{u}_k) \in \hat{\Omega}_k \quad \forall\ k = 0, ..., N-1\,. \tag{19d}$$

With the help of efficient LP solvers, the problem can be computed faster than the SOCP formulation proposed by [13] and requires only one LP compared to [10]. Because of the convexification (15), the resulting trajectory will be feasible but not necessarily optimal as the feasible set is an underapproximation of the original set when third-order constraints are considered. To obtain a locally optimal trajectory, a sequential solution process must be utilized while the set is linearized along the previously obtained path velocity. The sequence can be started by initially ignoring the third-order constraints. This increases the overall computation time but in practice, the solution converges after three LP iterations.

It is important to note that the incorporation of redundant actuation generates a significantly more complex problem, as it is not strictly convex. Although the solution for $s(\cdot)$ is unique, the torques and wrenches are not since the same acceleration can be obtained by a set of torques and interaction wrenches. When torques and interaction wrenches can change instantaneously, the redundancy can be resolved in every timestep during control with a small-scale quadratic program (QP) [2]. However, this is not the case when torque or wrench rate limits are considered, as the torques and wrenches cannot change arbitrarily fast. Although the redundancy could be resolved by adding a quadratic cost term in (19), it would increase the trajectory duration and the computation time as multiple QPs have to be solved sequentially. Because $s(\cdot)$ can be uniquely determined by solving (19), the redundancy can be resolved in a post-processing step by solving only one QP omitting kinematic constraints and with the optimal $s^*(\cdot)$ from (19).

B. Comparison to TOPP-RA

In TOPP-RA proposed by [7], the optimal trajectory is obtained by an iterative backward and forward pass for a one-dimensional system $\dot{s}_{k+1}^2 = \dot{s}_k^2 + 2\Delta\ddot{s}_k$. However, only second-order constraints can be enforced. We extend the method to a general linear system to allow for the incorporation of third-order constraints. In the backward pass, the linear controllable set is computed recursively by

$$\mathcal{K}_k = \{\boldsymbol{x}_k \in \mathbb{R}^n \,|\boldsymbol{d}_k\boldsymbol{x}_k \leq \boldsymbol{e}_k\} = \Pi_x \left(\Omega_k \bigcap \mathcal{K}_{k|k+1}\right) \tag{20}$$

until $k = N$. The set

$$\mathcal{K}_{k|k+1} = \{\boldsymbol{x}_k, \boldsymbol{u}_k \in \mathbb{R}^n \,|\boldsymbol{d}_{k+1}(\boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{u}_k) \leq \boldsymbol{e}_{k+1}\} \tag{21}$$

is the controllable set $\mathcal{K}_{k+1}$ backward propagated through the system dynamics. The end state defines $\mathcal{K}_N$ and the linear operator $\Pi_x : \mathbb{R}^{n+m} \to \mathbb{R}^n$ projects a set in $(\boldsymbol{x}, \boldsymbol{u})$ to a set in $\boldsymbol{x}$.

Starting from the initial state $\boldsymbol{x}_0$, the optimal trajectory is iteratively obtained by maximizing the path acceleration $[\boldsymbol{u}_k]_1$ subject to the admissible area Ω_k and $\mathcal{K}_{k|k+1}$. The next optimal state is calculated by $\boldsymbol{x}_{k+1}^* = \boldsymbol{\Phi}\boldsymbol{x}_k^* + \boldsymbol{\Gamma}\boldsymbol{u}_k^*$ until the end state is reached.

The solution of the TOPP-RA formulation converges with decreasing step size to the same solution of correct CO methods [18]. Next, we show that TOPP-RA and TOPP-LP solve the same problem. Thus, the optimality properties of TOPP-RA are inherited. Consider the LP (19). We add constraints $\boldsymbol{d}_k\boldsymbol{x}_k \leq \boldsymbol{e}_k \forall\ k = 1, ..., N$, which define the recursively computed controllable sets $\mathcal{K}_k$ at every stage. By adding these constraints the original problem does not change because they are redundant and map the final state constraint to stage k. This is already given implicitly in (19b). Due to the redundancy, the final state constraint can be removed. By evaluating the system dynamics into the cost (19a) and the newly added constraints, the following equivalent LP is obtained

$$\max_{\boldsymbol{x}_{1:M}, \boldsymbol{u}_{0:M}} \quad \sum_{k=1}^{M} [\boldsymbol{\Phi}\boldsymbol{x}_{k-1} + \boldsymbol{\Gamma}\boldsymbol{u}_{k-1}]_1 \tag{22a}$$

$$\text{s.t.} \quad \boldsymbol{x}_0 \text{ given} \tag{22b}$$

$$\boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{u}_k = \boldsymbol{x}_{k+1} \tag{22c}$$

$$(\boldsymbol{x}_k, \boldsymbol{u}_k) \in \hat{\Omega}_k \tag{22d}$$

$$\boldsymbol{d}_{k+1}(\boldsymbol{\Phi}\boldsymbol{x}_k + \boldsymbol{\Gamma}\boldsymbol{u}_k) \leq \boldsymbol{e}_{k+1} \tag{22e}$$

for all $k = 0, ..., M$ with $M = N - 1$.

A trajectory $\mathcal{X}_M^* = (\boldsymbol{x}_1^*, ..., \boldsymbol{x}_M^*)$ and $\mathcal{U}_M^* = (\boldsymbol{u}_0^*, ..., \boldsymbol{u}_M^*)$ is a subsolution of the original problem when the first M entries equal the optimal solution obtained by (19). By solving problem (22) for $M \leq N - 1$ such a subsolution of the original problem is retrieved because the final state constraint is encoded in (22e). Therefore, (22) can be solved iteratively starting from $\boldsymbol{x}_0$ and setting $M = 1$. The following iterations are initialized with the previously computed optimal state. Using Bellman's principle of optimality, the optimal trajectory must consist of the subsolutions. Therefore, the iteratively obtained solution is optimal too. The iterative solution process is exactly the process proposed by TOPP-RA which shows that the methods can be equivalently transformed into each other. This allows us to inherit the optimality properties of TOPP-RA to TOPP-LP.

Notes on Computational Complexity: The original TOPP-RA method for second-order constraints has time complexity $\mathcal{O}(N)$ because the controllable set $\mathcal{K}_k$ is given by an interval which can be obtained by two small-scale LPs at each grid point. In this case, the TOPP-RA method is more efficient and should be preferred. However, when the state dimension n increases, TOPP-RA becomes impractical. When $n > 1$ the controllable set is no longer an interval but a polytope. Propagating polytopes through the dynamics with constrained inputs results in an increased number of constraints compared to $\mathcal{K}_{k+1}$. As the backward pass requires a projection $\Pi_{\boldsymbol{x}}$ onto the n-dimensional state space, the computational complexity increases significantly. Because the projection of polytopes between halfspace representations is known to be NP-hard [23], the overall method becomes impractical.

In practice, the number of constraints typically does not grow throughout the entire recursion process, as state constraints imposed by Ω_k can reduce the number of constraints in $\mathcal{K}_k$ compared to $\mathcal{K}_{k+1}$. Nevertheless, for higher state dimensions, the explicit computation of the controllable set requires much overhead. During the forward pass, most of this effort is wasted as only a single path on the complex boundary is of interest. The initially appealing linear complexity of the original TOPP-RA method is lost. On the other hand, when exploiting the sparse constraint structure of the large LP problem, the computational time is reduced.

IV. Numerical Results

In this section, the application of the proposed TOPP-LP method is demonstrated for a cooperative MARS with two *KUKA iiwa 7 R800* which tightly grasp a $40\,\mathrm{kg}$ object. Using the proposed method of Section II, the path is planned for the object in Cartesian space by passing through a feasible set of poses. The analytical IK solution [22] is used to generate the consistent joint path. The dynamics and Jacobians of the manipulators are computed with the *Robotics System Toolbox* of *MATLAB*, while the LP is solved by *linprog* of *MATLAB* with the *interior-point* algorithm. The computations are performed on a $1.80\,\mathrm{GHz}$ *Intel Core i5-8250U*. The velocity and torque limits are provided by the manufacturer and are shown in Tab. I. The minimum values are defined by $\dot{\boldsymbol{q}}^- = -\dot{\boldsymbol{q}}^+$ and $\boldsymbol{\tau}^- = -\boldsymbol{\tau}^+$. The acceleration, jerk and torque rate limits are not provided. Here $\ddot{\boldsymbol{q}}^\pm = 10\,\mathrm{s}^{-1}\dot{\boldsymbol{q}}^\pm$, $\dddot{\boldsymbol{q}}^\pm = 15\,\mathrm{s}^{-1}\ddot{\boldsymbol{q}}^\pm$ and $\dot{\boldsymbol{\tau}}^\pm = 10\,\mathrm{s}^{-1}\boldsymbol{\tau}^\pm$ are assumed. The limit on the projected nullspace wrench is set to $\tilde{\boldsymbol{h}}^\pm = \pm 10$ for every element. The wrench rate is limited to $\dot{\tilde{\boldsymbol{h}}}^\pm = \pm 5$. These are very conservative limits to illustrate the effect on the trajectory. The resulting motion and trajectories can be seen in the supplementary video.

In addition to the complete problem formulation (full TOPP3), a softened version (soft TOPP3) is defined. Here, jerk limits are the only third-order constraints, omitting torque rate and wrench rate limits. This simplifies the problem significantly since the state consists of two dimensions and incorporates 13 inputs, including the wrenches. By contrast, the full TOPP3 problem encompasses 14 states and 13 inputs. Since the wrench is a state in the full TOPP3 problem, it is necessary to calculate $\boldsymbol{h}(0)$ and $\boldsymbol{h}(1)$. To address this, a QP is utilized with given boundary path states. Both TOPP3 versions are computed with three LP iterations initialized with a solution that omits third-order constraints (TOPP2). The resulting trajectories are depicted in Fig. 2 along with the maximum velocity curve (MVC) that represents the maximum pseudo velocity within the admissible set Ω.

Restrictive wrench rate limits in the full TOPP3 solution result in slower trajectories compared to TOPP2 and the soft

TABLE I: Velocity and torque limits of *KUKA iiwa 7 R800*.

axis	1	2	3	4	5	6	7
$\dot{\boldsymbol{q}}^+$ [°/s]	98	98	100	130	140	180	180
$\boldsymbol{\tau}^+$ [N m]	176	176	110	110	110	40	40

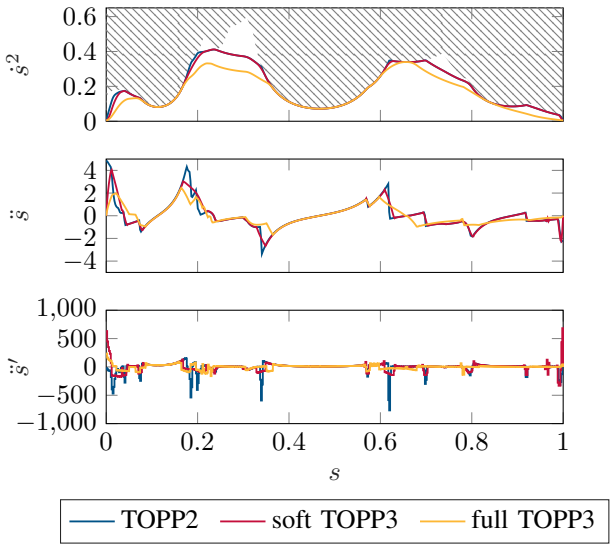

Fig. 2: Solutions for pseudo path velocity, path acceleration, and pseudo path jerk of a cooperative MARS with grid size $N = 500$ and the infeasible velocity area (hatched), which is constrained by the MVC.

TOPP3 solutions which can be seen in Tab. II. While the latter two have similar durations, the full TOPP3 solution is significantly slower. On the other hand, only the full TOPP3 solution ensures that all third-order limits are enforced. The TOPP2 solution violates all third-order constraints, while the soft TOPP3 solution violates only the torque rate and wrench rate limits and the maximum violation is slightly smaller compared to the TOPP2 solution.

Fig. 3 shows the first element of the normalized projected wrench. Because the wrenches for both the TOPP2 and soft TOPP3 versions can change instantaneously, high violations in the wrench and torque rates can be expected. For TOPP2 and the soft TOPP3 formulations the actuation redundancy is resolved using a QP at every stage similar to [2]. However, the full TOPP3 solution is not post-processed as the instantaneous optimization is not possible. This can lead to jitter in the wrenches and torques. Due to the restrictive limit on the wrench derivative, this effect is not very apparent.

The total computation time in Tab. II shows that the full TOPP3 version is significantly slower compared to the other two solutions. This is due to the increased complexity of the problem formulation. The TOPP2 solution is the fastest, as

TABLE II: Numerical results of TOPP for the cooperative MARS with grid size $N = 500$.

	TOPP2	soft TOPP3	full TOPP3
T	2.658 s	2.709 s	3.144 s
$\max \frac{\lvert[\ddot{\boldsymbol{q}}]_i\rvert}{[\ddot{\boldsymbol{q}}]_i^+}$	5.14	1	1
$\max \frac{\lvert[\dot{\boldsymbol{\tau}}]_i\rvert}{[\dot{\boldsymbol{\tau}}]_i^+}$	11.08	7.88	1
$\max \frac{\lvert[\dot{\tilde{\boldsymbol{h}}}]_i\rvert}{[\dot{\tilde{\boldsymbol{h}}}]_i^+}$	639.24	631.31	1
CPU time	1.156 s	12.061 s	29.34 min

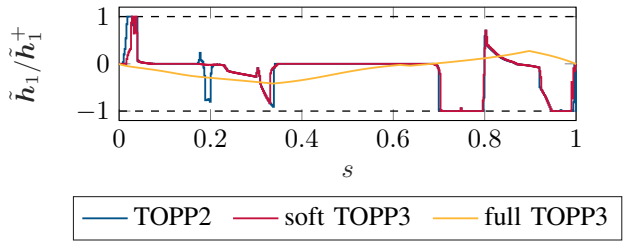

Fig. 3: Projected wrench for different TOPP problem formulations normalized to limit.

it is the least complex problem. A similar picture is drawn in Fig. 4 which shows the computation time over the grid size N. Comparing the computational times with previous works is difficult, as the problems and computational hardware differ. The reported times in [13], [17], [19] are for single-arm robots only. Although the problem in [20] is more relatable, no computation times were provided. A comparison is only done for the TOPP2 and the soft TOPP3 formulations, as none of the previous work has considered MARS with torque and wrench constraints. The here considered problem is more complex as more constraints and inputs are used. Nevertheless, the computational times are lower compared to the ones reported in previous work.

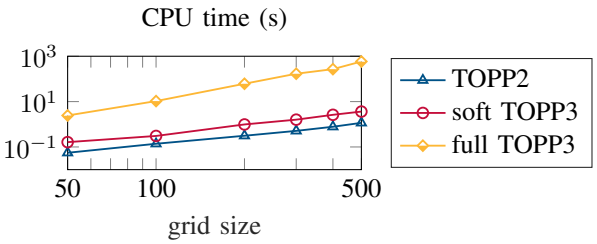

Fig. 4: Computation time per iteration.

V. Conclusion

This paper introduces TOPP for cooperative MARS with up to third-order constraints, such as jerk, torque rate, and wrench rate limits. To solve the underlying OCP, a method based on sequential LP is presented. The correctness of the method is derived from its equivalence with a generalized TOPP-RA approach. The effectiveness of the method is demonstrated for a cooperative MARS comprised of two 7-DOF robots and a 6-DOF object, while the importance of third-order constraints for the trajectory smoothness is highlighted. Adding torque rate limits for cooperative MARS enlarges the problem size significantly, resulting in longer computation times. While the actuation redundancy can be resolved instantaneously when only jerk limits are considered, handling torque- and wrench rate limits is more challenging.

Although incorporating third-order constraints enhances trajectory continuity, the system always operates at its limits. In case of model inaccuracies or disturbances, the underlying controller can no longer follow the path. To address this, future research will focus on predictive path-following control for cooperative MARS.

References

[1] F. Sherwani, M. M. Asad, and B. Ibrahim, "Collaborative Robots and Industrial Revolution 4.0 (IR 4.0)," in *Proc. of ICETEST*, March 2020, pp. 168–172.

[2] M. Dio, A. Völz, and K. Graichen, "Cooperative Dual-Arm Control for Heavy Object Manipulation based on Hierarchical Quadratic Programming," in *Proc. of IROS*, Detroit, USA, 2023, pp. 643–648.

[3] H. Choset, K. M. Lynch *et al.*, *Principles of Robot Motion: Theory, Algorithms, and Implementations*. Cambridge: MIT Press, 2014.

[4] A. Reiter, A. Müller, and H. Gattringer, "On Higher Order Inverse Kinematics Methods in Time-Optimal Trajectory Planning for Kinematically Redundant Manipulators," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 4, pp. 1681–1690, 2018.

[5] J. Bobrow, S. Dubowsky, and J. Gibson, "Time-Optimal Control of Robotic Manipulators Along Specified Paths," *The International Journal of Robotics Research*, vol. 4, no. 3, pp. 3–17, 1985.

[6] Q.-C. Pham, "A General, Fast, and Robust Implementation of the Time-Optimal Path Parameterization Algorithm," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1533–1540, 2014.

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

[8] L. Yun-Hui and S. Arimoto, "Minimum-Time Trajectory Planning for Multiple Manipulators Handling an Object with Geometric Path Constraints," in *Proc. of IROS*, Osaka, Japan, 1991.

[9] Q.-C. Pham and O. Stasse, "Time-Optimal Path Parameterization for Redundantly Actuated Robots: A Numerical Integration Approach," *Transactions on Mechatronics*, vol. 20, no. 6, pp. 3257–3263, 2015.

[10] K. Hauser, "Fast Interpolation and Time-Optimization with Contact," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1231–1250, Aug. 2014.

[11] S. Mansouri, M. J. Sadigh, and M. Fazeli, "A computationally efficient algorithm to find time-optimal trajectory of redundantly actuated robots moving on a specified path," *Robotica*, vol. 37, no. 1, pp. 62–79, 2019.

[12] H. Pham and Q.-C. Pham, "On the Structure of the Time-Optimal Path Parameterization Problem with Third-Order Constraints," in *Proc. of ICRA*, Singapore, 2017.

[13] D. Verscheure, B. Demeulenaere *et al.*, "Practical Time-Optimal Trajectory Planning for Robots: A Convex Optimization Approach," *IEEE Transactions on Automatic Control*, vol. 14, 2008.

[14] A. Nagy and I. Vajk, "LP-based Velocity Profile Generation for Robotic Manipulators," *International Journal of Control*, vol. 91, no. 3, pp. 582–592, 2018.

[15] E. Barnett and C. Gosselin, "A Bisection Algorithm for Time-Optimal Trajectory Planning Along Fully Specified Paths," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 131–145, 2021.

[16] H. Haghshenas, M. Norrlöf, and A. Hansson, "A convex optimization approach to time-optimal path tracking problem for cooperative manipulators," *IFAC-PapersOnLine*, vol. 52, no. 10, pp. 400–405, 2019.

[17] F. Debrouwere, W. Van Loock *et al.*, "Time-Optimal Path Following for Robots with Trajectory Jerk Constraints using Sequential Convex Programming," in *Proc. of ICRA*, 2017, pp. 1916–1921.

[18] I. Spasojevic, V. Murali, and S. Karaman, "Asymptotic Optimality of a Time Optimal Path Parametrization Algorithm," *IEEE Control Systems Letters*, vol. 3, no. 4, pp. 835–840, Oct. 2019.

[19] D. Kaserer, H. Gattringer, and A. Müller, "Nearly Optimal Path Following With Jerk and Torque Rate Limits Using Dynamic Programming," *IEEE Transactions on Robotics*, vol. 35, no. 2, pp. 521–528, 2019.

[20] D. Kaserer, H. Gattringer, and A. Müller, "Time Optimal Motion Planning and Admittance Control for Cooperative Grasping," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2216–2223, Apr. 2020.

[21] M. Žefran and V. Kumar, "Interpolation schemes for rigid body motions," *Computer-Aided Design*, vol. 30, no. 3, pp. 179–189, 1998.

[22] C. Faria, F. Ferreira *et al.*, "Position-based kinematics for 7-dof serial manipulators with global configuration control, joint limit and singularity avoidance," *Mechanism and Machine Theory*, vol. 121, pp. 317–334, 2018.

[23] H. R. Tiwary, "On computing the shadows and slices of polytopes," *CoRR*, vol. abs/0804.4150, 2008. [Online]. Available: http://arxiv.org/abs/0804.4150